LIVRODEATAS

SIMPÓSIO DE
ENGENHARIA INFORMÁTICA

Departamento de Engenharia Informática
Instituto Superior de Engenharia do Porto
9 de dezembro de 2025

SIMPÓSIO ENGENHARIA INFORMÁTICA · ISEP · P.PORTO · SOFTINGAL · nei-isep

TITLE:
Simpósio de Engenharia Informática 2025: Proceedings Book

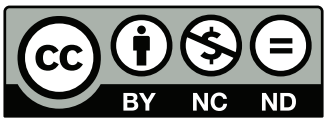
EDITORS:
Carolina Sá, Catarina Oliveira, Emanuel Silva, Marílio Cardoso, Nuno Morgado, Paulo Proença, Piedade Carvalho, Rafael Vieira, Ricardo Meireles e Sérgio Moreira

PUBLISHER:
Instituto Superior de Engenharia do Porto (ISEP) – P.Porto

DESIGN/LAYOUT:
Gabinete de Design e Multimédia do ISEP

VENUE/DATE:
ISEP/December 9th, 2025

ISBN:
978-989-36167-7-2



Instituto Superior de Engenharia do Porto (ISEP)
Rua Dr. António Bernardino de Almeida, 431
4249-015 Porto
phone: +351 22 83 40 500

www.isep.ipp.pt
http://sei.dei.isep.ipp.pt

Acknowledgment

We would like to thank everyone who contributed to the realisation of SEI'25.

First of all, we would like to express our gratitude to both the Presidency of the Engineering School of the Porto Polytechnic (ISEP) and the Management of the Department of Computer Engineering (DEI), for their continuous support and the resources made available that made this event possible.

We also thank the editors, reviewers, and session chairs who dedicated their time and expertise to ensuring the quality of the papers and presentations delivered during the symposium.

Special thanks to the Computer Engineering Assotiation (NEI), students and participants, whose commitment, energy and enthusiasm were fundamental to the dynamics of the symposium.

We thank the guest speakers, who shared their valuable knowledge and experiences, enriching the discussions.

Furthermore, we would like to thank the technical team, the DEI secretariat and the ISEP Communication and Image Office for their essential support and collaboration throughout the event.

Last but not least, we would like to thank the companies and organisations that supported this event, contributing to the promotion of innovation and collaboration in the area of Computer Engineering and related areas.

Contents

Symposium's Vision

The Computer Engineering Symposium (SEI) provides a dynamic platform for showcasing the latest innovations, research discoveries, and emerging solutions in Computer Engineering and related fields. This year's edition highlightts Artificial Intelligence (AI), offering a unique opportunity for students, professionals, and researchers to convene, exchange ideas, and explore how generative AI is poised to transform various aspects of our lives, from education to industry and organizational functions.

SEI'25 emphasized the potential of AI as a tool to facilitate personalized learning and enrich the overall educational experience. Participants delved into a diverse array of topics, ranging from integrating AI technologies for increased accuracy in furniture detection and volume estimation to examining the impact of AI in higher education, including its promises and challenges.

Furthermore, the symposium showcased cutting-edge research and practical applications of AI across multiple domains. Participants shared insights on topics such as:

- Arduino-based prototyping and instrumentation
- Cross-platform development and unified communications
- Business process automation with LLMs/NLP
- Infrastructure inventory, monitoring, and incident response (on-prem and cloud)
- Conversational analytics with RAG for Power BI
- Intelligent systems for health, nutrition, and academic/enterprise management

SEI'25 aimed to inspire a new generation of innovators and leaders, empowering them to harness the potential of emerging technologies in a variety of contexts. By fostering knowledge exchange and showcasing real-world applications, the symposium promotes a well-informed generation capable of making decisions that benefit society as a whole.

Committees

Organization Committee

Carolina Sá	Núcleo de Estudantes de Engenharia Informática do ISEP
Catarina Oliveira	Instituto Superior de Engenharia do Porto
Emanuel Silva	Instituto Superior de Engenharia do Porto
Marílio Cardoso	Instituto Superior de Engenharia do Porto
Nuno Morgado	Instituto Superior de Engenharia do Porto
Paulo Proença	Instituto Superior de Engenharia do Porto
Piedade Carvalho	Instituto Superior de Engenharia do Porto
Rafael Vieira	Núcleo de Estudantes de Engenharia Informática do ISEP
Ricardo Meireles	Núcleo de Estudantes de Engenharia Informática do ISEP
Sérgio Moreira	Instituto Superior de Engenharia do Porto

Technical-Scientific Committee

A. Augusto Sousa	Faculdade de Engenharia da Universidade do Porto
Alexandre Gouveia	Instituto Superior de Engenharia do Porto
Álvaro Rocha	Instituto Superior de Economia e Gestão
Ana Raquel Faria	Instituto Superior de Engenharia do Porto
Ana Rita Teixeira	Instituto Superior de Engenharia do Porto
Anabela Serrano	Instituto Superior de Contabilidade e Administração do Porto
Bertil Marques	Instituto Superior de Engenharia do Porto
Catarina Oliveira	Instituto Superior de Engenharia do Porto
Emanuel Silva	Instituto Superior de Engenharia do Porto
Fátima Leal	Instituto Superior de Engenharia do Porto
Fátima Rodrigues	Instituto Superior de Engenharia do Porto
Fernando Duarte	Instituto Superior de Engenharia do Porto
Helder Pinto	Instituto Superior de Engenharia do Porto
João Pascoal Faria	Faculdade de Engenharia da Universidade do Porto
José Reis Tavares	Instituto Superior de Engenharia do Porto
José Marinho	Instituto Superior de Engenharia do Porto
Luís Alves	Instituto Politécnico de Bragança
Marílio Cardoso	Instituto Superior de Engenharia do Porto
Mário Cruz	Escola Superior de Educação
Mário Pinto	Escola Superior de Media Artes e Design
Nuno Bettencourt	Instituto Superior de Engenharia do Porto

Nuno Morgado	Instituto Superior de Engenharia do Porto
Orlando Sousa	Instituto Superior de Engenharia do Porto
Paula Flores	Escola Superior de Educação
Paula Tavares	Instituto Superior de Engenharia do Porto
Paulo Baltarejo Sousa	Instituto Superior de Engenharia do Porto
Paulo Maio	Instituto Superior de Engenharia do Porto
Paulo Matos	Instituto Superior de Engenharia do Porto
Paulo Proença	Instituto Superior de Engenharia do Porto
Pedro Pinto	Instituto Superior de Engenharia do Porto
Piedade Carvalho	Instituto Superior de Engenharia do Porto
Rosa Reis	Instituto Superior de Engenharia do Porto
Rui Marques	Instituto Superior de Engenharia do Porto
Sérgio Moreira	Instituto Superior de Engenharia do Porto
Tânia Rocha	Universidade de Trás-os-Montes e Alto Douro
Telmo Matos	Faculdade de Engenharia da Universidade do Porto
Teresa Terroso	Escola Superior de Media Artes e Design

Technical Support and Secretariat

Carla Cunha	Instituto Superior de Engenharia do Porto
Lurdes Santos	Instituto Superior de Engenharia do Porto
Nuno Fonseca	Instituto Superior de Engenharia do Porto
Pedro Rocha	Instituto Superior de Engenharia do Porto

Invited Speakers

Hugo Ferreira	Ethiac
Pedro Carneiro	Roboyo
Pedro Pinto	ISEP
Rui Moura	Critical Techworks

Keynote Session

Hack or Get Hacked: A Importância do Hacking Ético

Hugo Ferreira
Ethiac

Director of Hacking Operations at Ethiack and professor at ISTEC-Porto, he has over 7 years of experience in IT security. He is an enthusiast of social engineering and ethical hacking, having participated in Live Hacking Events and CVE publications. Lately, he has primarily managed hacking teams with Ethiack, finding vulnerabilities in organizations so that they can mitigate them and strengthen their systems.

Round Table

Cybersecurity Strategic Directions: New Priorities, Skills Gap, and Evolving Career Pathways

An analysis of new strategic directions in cybersecurity and their implications for workforce development: emerging roles, hybrid skill needs, employer expectations, and evolving professional pathways.

Moderator

Pedro Pinto
ISEP

Pedro Pinto is an Assistant Professor at the Instituto Superior de Engenharia do Porto (ISEP), Portugal, and a researcher at the GECAD and INESC TEC research centers. He served as the Data Protection Officer (DPO) and the Director of the Master's in Cybersecurity at the Instituto Politécnico de Viana do Castelo (IPVC) in Portugal. He also served as deputy director of the Applied Digital Transformation Laboratory (ADiT-Lab) Research Center, Portugal.

His main interests include telecommunications, computer networks and systems, cybersecurity, and data protection and privacy. He is also a recipient of honorable mentions and awards in vulnerability reward programs (e.g., Google and Microsoft) and serves as an expert for the Portuguese Attorney General's Office, for scientific councils of foreign countries, and for the European Commission.

Participants

Pedro Carneiro

Roboyo

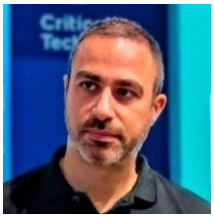
Global Lead of Cybersecurity at ROBOYO, a company that acquired WeMake - Tecnologias de Informação, Lda, where he served as Agile Project Manager & CISO and was a founding partner for 20 years, since 2002.

He is the Director of WeSecure - Cybersecurity & Forensics Analysis, with extensive experience in Information Security and Business Continuity projects for major national institutions, including operators of critical and essential services.

With the acquisition of Roboyo, it manages and integrates international cybersecurity projects, operating directly across all companies in the group.

He has remained a DPO (Data Protection Officer) in public and private companies since 2018.

He has also been part of the Attorney General's Office's pool of forensic experts in the area of secure application development since 2017.



Rui Moura

Critical Techworks

Rui specializes in cybersecurity, advanced solutions, and project management. He has successfully led large-scale global projects related to electronic identification (eID), ePassports (PKIs), and information security across Europe, Asia, Africa, and the Americas.

Currently at Critical TechWorks in Porto, Portugal, Rui serves as Chief Technical Titan for Cyber Security and Infrastructure, providing strategic leadership in cybersecurity initiatives and infrastructure and software protection.

Always focused on enhancing security and efficiency, Rui actively pursues ways to strengthen cybersecurity posture and streamline processes within projects and teams.

Papers

Session	Title	Chair
4A	"Prototipagem e Controlo em Projetos de Engenharia: Arduino e Energias Renováveis"	Paulo Matos
4B	"Soluções Inovadoras em Desenvolvimento de Software e Aplicações Multiplataforma"	Rosa Reis
5A	"Soluções Avançadas para Processos Empresariais e Infraestruturas"	Rui Marques
5B	"Automação Inteligente e Apoio à Decisão com LLMs"	F. Jorge Duarte
6A	"Soluções Avançadas para Monitorização, Controlo e Análise em Azure"	Alexandre Gouveia
6B	"Transformação Digital: Soluções para Gestão Empresarial, Biomédica e Académica"	Paulo Maio

Session 4A

"Prototipagem e Controlo em Projetos de Engenharia: Arduino e Energias Renováveis"

Chair: Paulo Matos **Room:** H202 **Time:** 14:15 - 15:15

Paper	Presenter
Braço Ótico com Controle Arduíno para Medição do Índice de Refração	Ana Rita Moreira
Projeto e Desenvolvimento de um Mini Frigorífico e uma App de registo da sua temperatura com recurso a Microcontrolador Arduino	Christopher Sá
Controlo e Simulação de um Sistema Fotovoltaico a Instalar no Edifício G do ISEP	José Magalhães
Estudo e Simulação de sistema fotovoltaico para produção energética a instalar Edifício E do ISEP	José Magalhães

Braço Ótico com Controle Arduíno para Medição do Índice de Refração

Ana Rita Moreira[1,2], C.A. Ramos [1,2], G. Vilão [1,2]

[1]Dep. de Física, ISEP, Instituto Politécnico do Porto, Rua Dr. António Bernardino de Almeida 431, 4249-015 Porto, Portugal
[2]NBIN/CIETI, ISEP, Instituto Politécnico do Porto, Rua Dr. António Bernardino de Almeida 431, 4249-015 Porto, Portugal
{1200851, CAR, GMR}@isep.ipp.pt

Resumo. A automação em contexto laboratorial tem reforçado a precisão e a reprodutibilidade das medições no domínio biomédico. Neste trabalho descreve-se um sistema robótico automatizado para determinar o índice de refração de tecidos biológicos através do ângulo de Brewster, integrando controlo motorizado, aquisição sensorial e análise computacional. O protótipo, composto por motores de passo controlados por Arduino e por um fotodetetor com processamento em tempo real numa interface Python, realiza varrimentos angulares sincronizados e identifica autonomamente o ponto mínimo de refletância, completando todo o ciclo de medição. Em material de referência obteve-se um ângulo de Brewster de 56,2° e um índice de 1,495 com erro inferior a 0,5%, enquanto nos ensaios com tecido muscular os valores variaram entre 1,375 e 1,393, refletindo a influência da orientação das fibras. Os resultados confirmam a precisão e a repetibilidade do sistema, demonstrando o seu potencial como solução acessível para caracterização ótica de tecidos biológicos.

Palavras-chave: Automação, Arduíno, controlo de motores, medição ótica, ângulo de Brewster, índice de refração.

1 Introdução

A determinação do índice de refração em tecidos biológicos é um tema relevante na ótica biomédica, uma vez que esse parâmetro influencia fenómenos de absorção, dispersão e penetração da luz [1]. O conhecimento exato dessas propriedades é determinante para o desenvolvimento de aplicações clínicas e tecnológicas baseadas na interação luz-tecido, como terapias fotodinâmicas, diagnóstico por imagem e caracterização de lesões [2].

A integração entre robótica e automação de equipamentos tem impulsionado o desenvolvimento de sistemas experimentais capazes de executar tarefas complexas com precisão e reprodutibilidade consideráveis. Em contexto laboratorial, a automatização de processos de medição permite reduzir o erro humano, otimizar o tempo de aquisição de dados e aumentar a consistência dos resultados [3].

A medição do índice de refração de tecidos biológicos é um processo tradicionalmente dependente de equipamentos laboratoriais e do operador. Apesar da elevada precisão dos métodos laboratoriais tradicionais, a sua complexidade, custo e necessidade de condições laboratoriais controladas tornam a sua utilização limitada fora do contexto de investigação [2].

Neste enquadramento, a automatização da medição ótica surge como uma solução, combinando motores, um sensor ótico e uma interface de visualização instantânea de medições efetuadas e cálculo final de índice de refração. Com isto, é possível construir sistemas economicamente acessíveis, capazes de realizar medições de forma autónoma, repetível e adaptável a diferentes materiais.

O presente trabalho consistiu no desenvolvimento de um protótipo de medição automatizada, concebido para determinar o índice de refração de tecidos biológicos através do ângulo de Brewster. O protótipo é composto por um subsistema de controlo de movimento motorizado, um subsistema de aquisição sensorial ótica e um subsistema de monitorização e decisão computacional, implementado em Python.

O controlo é realizado por um microcontrolador Arduino, que coordena o movimento sincronizado de dois motores de passo e garante varrimentos angulares precisos, assegurando o posicionamento ideal para a deteção do ponto mínimo de refletância. Paralelamente, a interface Python processa os dados em tempo real, deteta automaticamente o ângulo de Brewster e calcula o índice de refração, completando um ciclo fechado de medição e análise.

2 A evolução das medições do índice de refração

A medição do índice de refração de tecidos biológicos tem sido amplamente estudada nas últimas décadas, devido à sua importância na caracterização ótica e na diferenciação entre tecidos saudáveis e patológicos [2]. Posto isto, diversas abordagens experimentais foram propostas, variando em complexidade e precisão.

A técnica de reflexão interna total (TIR), introduzida por Li e Xie [4], é uma das mais utilizadas na determinação do índice de refração de tecidos biológicos (Figura 1). O método consiste em medir o ângulo crítico de reflexão entre a amostra e um semicírculo de vidro, permitindo distinguir tecidos normais e patológicos em diferentes comprimentos de onda. Apesar da sua simplicidade, requer um contacto controlado entre amostra e substrato, o que pode comprometer a reprodutibilidade em tecidos irregulares.

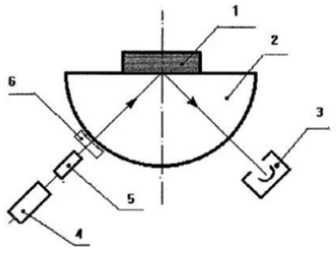


Fig. 1 - Método de medição do índice de refração do tecido biológico por reflexão interna total. 1-Amostra; 2- Lente semicilíndrica; 3- Detetor de fotões; 4- Laser; 5- Polarizadora; 6- Lente plano-côncava [4]

A espectrometria com esferas integradoras, desenvolvida por Shimojo et al. [5], tem sido empregue para medir refletância difusa e transmitância total em tecidos, possibilitando a obtenção simultânea dos coeficientes de absorção e dispersão reduzida (Figura 2). Quando associada a modelos de inversão numérica, permite calcular o índice de refração complexo com alta precisão, mas exige calibração rigorosa e equipamento dispendioso.

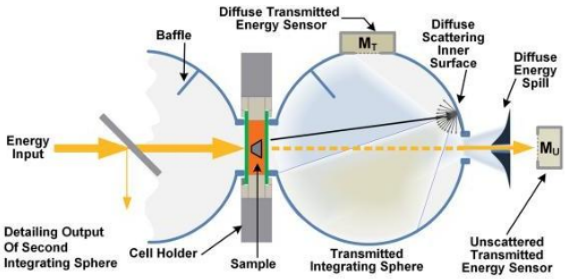


Fig. 2 - Esquema ilustrativo do sistema espectrométrico de esfera de integração dupla [5]

Dirckx et al. [6] desenvolveram uma técnica, que diz respeito a um método baseado num microscópio confocal, permitindo o cálculo do índice de refração, visto que é possível medir o caminho ótico numa amostra de espessura conhecida (Figura 3). No entanto, a medição da espessura dos tecidos é invasiva, pois requer a medição física da espessura do mesmo, pelo que esta é uma das grandes desvantagens desta técnica.

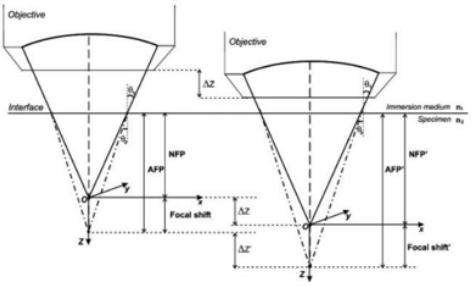


Fig. 3 - Esquema básico de uma lente objetiva de imersão [6]

Complementarmente, os métodos de simulação Monte Carlo (MC), descrito por Torres-García et al. [7], como o MCLTmx ou o FullMonteCUDA, são amplamente utilizados para prever a propagação da luz em meios biológicos complexos. Estas abordagens permitem modelar a distribuição da energia luminosa e avaliar o impacto da anisotropia, mas não são métodos diretos de medição e requerem elevada capacidade computacional [8].

Assim sendo, nenhuma das técnicas tradicionais integram de forma coesa as vertentes de controlo automatizado, aquisição sensorial ótica e tratamento automático dos dados num único sistema dedicado à medição do índice de refração de tecidos biológicos.

Neste contexto, o presente trabalho distingue-se pela implementação de um protótipo controlado por Arduino, capaz de realizar varrimentos angulares automáticos e identificar o ângulo de Brewster, aliando a simplicidade, precisão e portabilidade.

3 Materiais e métodos

O sistema desenvolvido (Figura 4) integra componentes óticos, eletrónicos e mecânicos controlados por microcontroladores Arduino Uno. Dois motores de passo NEMA 17, comandados por drivers DRV8825, asseguram o movimento angular preciso de um prato rotativo (goniómetro) e de um braço móvel que sustenta o fotodetetor. O conjunto é alimentado por uma fonte industrial TDK-Lambda 24 V, estabilizada e regulada a 18 V para compatibilidade com os controladores.
O funcionamento do sistema desenvolvido é composto por três módulos principais:

1. Subsistema de controlo de movimento, responsável pela execução dos varrimentos angulares;
2. Subsistema de aquisição sensorial, encarregado da aquisição do sinal ótico;
3. Subsistema de monitorização dos dados adquiridos, responsável pela análise automática dos dados e posterior cálculo do índice de refração.

Esta divisão permite uma integração entre *hardware* e *software*, garantindo que cada componente execute uma função autónoma, mas sincronizada.

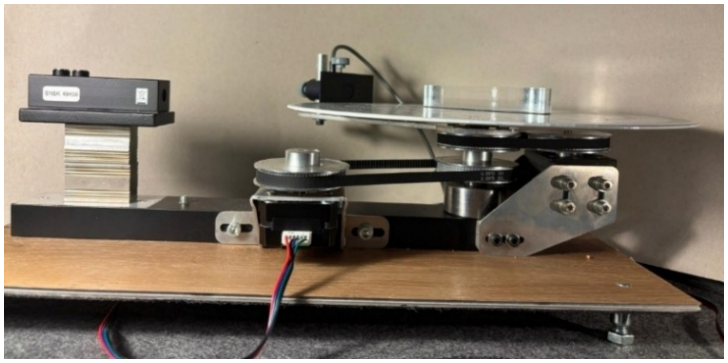


Fig. 4 - Protótipo mecânico do sistema de medição (vista lateral)

3.1 Subsistema de controlo de movimento

O controlo mecânico do sistema é realizado por dois motores de passo NEMA 17, cada um responsável por um componente do sistema (goniómetro e braço que contém o fotodetetor): o primeiro controla o braço móvel que realiza varrimentos angulares entre 45° e 65° e o segundo comanda o goniómetro que avança 2° após cada ciclo de varrimento, simulando o comportamento de um manipulador com dois eixos coordenados.
Ambos os motores são acionados por drivers DRV8825, configurados com um passo de $1{,}8^{0}$, no caso do goniómetro, e $0{,}45^{0}$, no caso do braço. O controlo é gerido por um

microcontrolador Arduino Uno, programado em linguagem C++ onde o sistema executa automaticamente as sequências de varrimento e reposicionamento.
Assim sendo, o código implementa uma sequência de estados que gere automaticamente o ciclo de operação (Figura 5):

Fig. 5 – Diferentes etapas da automação do motor: 1- Inicialização dos motores colocando o braço na posição de 45º e o goniómetro na posição de 48º; 2- Execução do varrimento angular do braço entre 45° e 65°; 3- Deteção e registo de intensidades a cada incremento de 0,45°; 4- Retorno do braço à posição inicial; 5- Avanço do goniómetro em 1,8°; 6- Repetição do ciclo até o utilizador parar o sistema.

Este processo é executado de forma totalmente autónoma, sem necessidade de intervenção manual. O Arduino comunica continuamente com a interface Python através de porta serial, enviando dados de ângulo e intensidade luminosa, e recebendo comandos de controlo (iniciar, parar, reiniciar ou ajustar parâmetros de varrimento).

3.2 Subsistema de aquisição sensorial

A aquisição ótica é assegurada por um fotodetetor acoplado a um amplificador operacional LM358, configurado com ganho de 100.
O fotodetetor está fisicamente acoplado ao braço móvel, garantindo que a variação angular do sensor acompanha o feixe incidente sobre a amostra. O posicionamento preciso entre o laser emissor e o detetor é fundamental para a obtenção do ângulo de Brewster, ponto em que a componente p-polarizada da luz refletida é mínima.
Os sinais adquiridos são digitalizados e transmitidos via porta serial para o computador, onde são processados em tempo real. Durante o varrimento angular, o Arduino envia, em intervalos regulares, pares de dados (ângulo, intensidade), que representam a posição atual do braço e o valor de intensidade medido, Figura 6.

```
  case LER_DADOS: {
// Ângulo absoluto (0 passos = 45°)
float anguloAbs = 45.0 + (braco.currentPosition() * degPerStep);
// Ângulo relativo ao início do varrimento atual (também começa em 45° no gráfico)
float anguloRel = 45.0 + ((braco.currentPosition() - bracoSweepStartSteps) * degPerStep);

float tensao = lerSensor();

// PARA O GRÁFICO: usa o relativo
Serial.print("A:");
Serial.print(anguloRel, 2);
Serial.print(";V:");
Serial.println(tensao, 4);
```

Fig. 6 – Leitura dos dados e representação gráfica

3.3 Subsistema de monitorização dos dados adquiridos

Foi desenvolvida uma interface computacional em Python, responsável pela comunicação serial com o Arduino e pela aquisição e tratamento dos dados experimentais. O *software*, criado com as bibliotecas PySerial, Matplotlib e SQLAlchemy, executando variadas funções. O sistema realiza automaticamente a deteção da porta *serial* e a sincronização com o microcontrolador Arduino, garantindo comunicação contínua e fiável. Durante o varrimento, é efetuada a aquisição em tempo real dos valores de intensidade luminosa e dos respetivos ângulos, sendo o sinal submetido a filtragem digital para reduzir o ruído e melhorar a estabilidade das medições.

```
def save_measurement(peaks, chosen_angle_deg, chosen_intensity_mv, refractive_index, meta,
                     theoretical_ri=None, error_ri=None, error_percent=None):
    """peaks: list[(angle_deg, intensity_mv, sweep_index)]"""
    if not peaks:
        raise ValueError("Sem picos para guardar.")
    if refractive_index is None:
        raise ValueError("Índice de refração inválido.")
    session = SessionLocal()
    try:
        m = Measurement(
            started_at=meta.get("started_at", datetime.utcnow()),
            ended_at=meta.get("ended_at", datetime.utcnow()),
            substance=meta.get("substance"),
            notes=meta.get("notes"),
            operator=meta.get("operator"),
            refractive_index=refractive_index,
            chosen_angle_deg=chosen_angle_deg,
            chosen_intensity_mv=chosen_intensity_mv,
            theoretical_ri=theoretical_ri,
            error_ri=error_ri,
            error_percent=error_percent,
            method_version=meta.get("method_version", "alg-v1.0"),
        )
        for i, (ang, inten, idx) in enumerate(peaks):
            m.peaks.append(Peak(angle_deg=ang, intensity_mv=inten, sweep_index=idx if idx is not None else i))
        session.add(m)
        session.commit()
        return m.id
    finally:
        session.close()
```

Fig. 7 – Tratamento dos dados.

A aplicação identifica de forma automática os picos máximos de cada varrimento, executa um ajuste polinomial de 4.ª ordem e determina o ângulo de Brewster através da derivada da curva obtida, Figura 7. Por fim, o sistema realiza o cálculo imediato do índice de refração e armazena todos os resultados numa base de dados SQLite, assegurando a rastreabilidade e organização das medições.

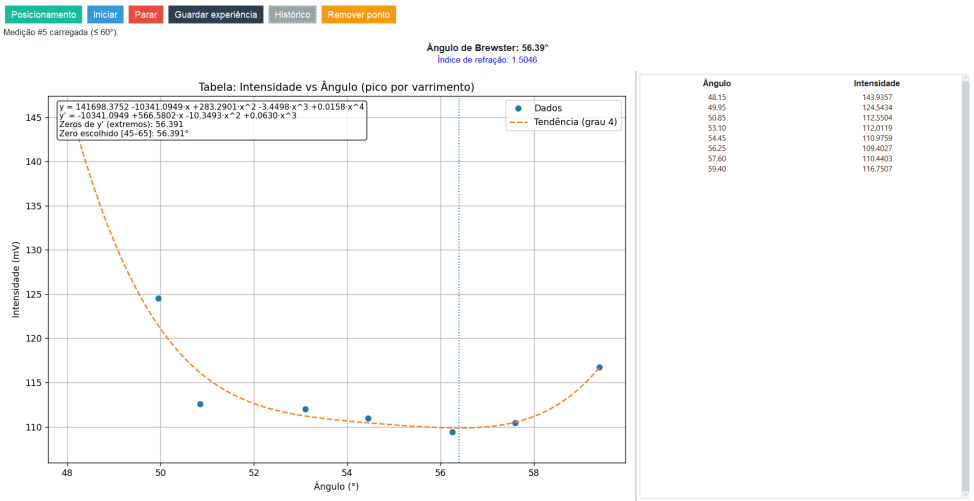


Fig. 8 - Interface desenvolvida para acompanhamento instantâneo das medições e processamento de dados. Gráfico que contém o pico máximo de intensidade de cada varrimento associado ao respetivo ângulo.

A interface (Figura 8) apresenta o gráfico de intensidade versus ângulo em tempo real e disponibiliza funções adicionais como guardar medições, aceder ao histórico (Figura 9), editar resultados e remover pontos do gráfico final desnecessários. Todo o processo, desde o posicionamento inicial até ao cálculo final do índice de refração, é executado de forma automática e interativa (Figura 10).

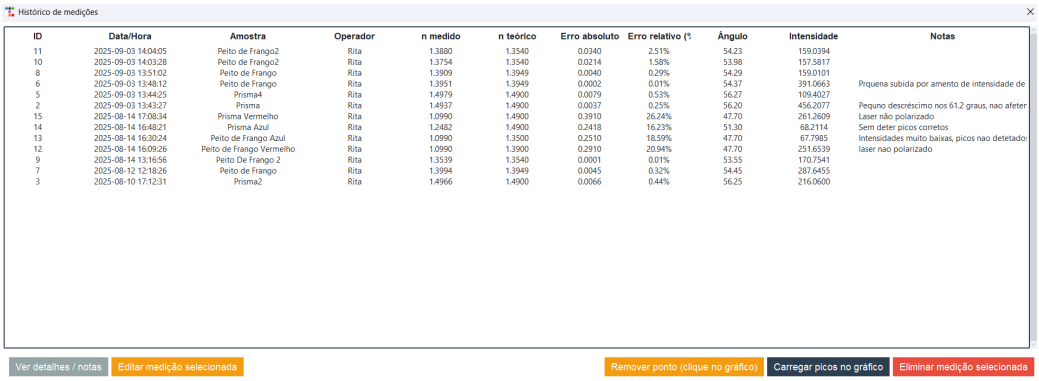


Fig. 9 – Interface do histórico de medições realizadas

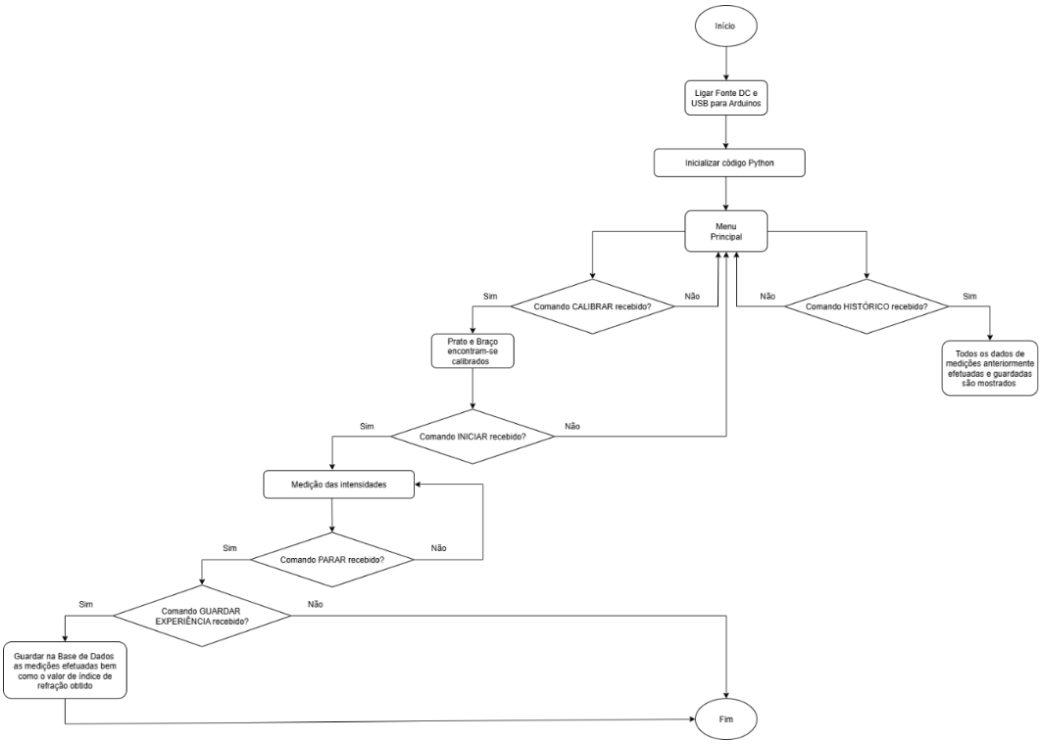


Fig. 10 - Fluxograma do funcionamento do sistema

4 Resultados e discussão

Nesta secção são apresentados e discutidos os resultados obtidos durante o desenvolvimento experimental do protótipo e a aplicação do método de medição automatizada do índice de refração. Inicia-se pela validação do sistema e do protocolo experimental, de modo a confirmar a precisão, estabilidade e reprodutibilidade do controlo robótico e da aquisição sensorial. Seguidamente, são analisados os resultados obtidos em diferentes amostras biológicas, considerando a influência da orientação das fibras musculares e a comparação com valores de referência descritos na literatura. Por fim, é discutido o comportamento espectral do sistema em função do comprimento de onda das fontes laser utilizadas.

4.1 Validação do sistema experimental

A primeira fase experimental teve como objetivo validar o protótipo desenvolvido, utilizando um semicírculo de acrílico como material de referência. A medição foi conduzida com o feixe laser verde (520 nm), e os valores de intensidade refletida foram registados em função do ângulo de incidência.
As curvas obtidas (Figura 10) apresentaram a morfologia característica do fenómeno de Brewster: a intensidade refletida diminuiu progressivamente até atingir um mínimo acentuado em torno de 56,2°, aumentando novamente para ângulos superiores. O ajuste polinomial de 4.ª ordem revelou um ângulo de Brewster de 56,2, correspondendo a um

índice de refração experimental de 1,495 ± 4,9%, em excelente concordância com o valor de referência do acrílico para o mesmo comprimento de onda (n = 1,49).
A diferença relativa inferior a 0,5% demonstra a exatidão e reprodutibilidade do protótipo, confirmando que o sistema ótico e o algoritmo de ajuste são capazes de identificar o ângulo mínimo de refletância com elevada precisão, mesmo na presença de ruído.

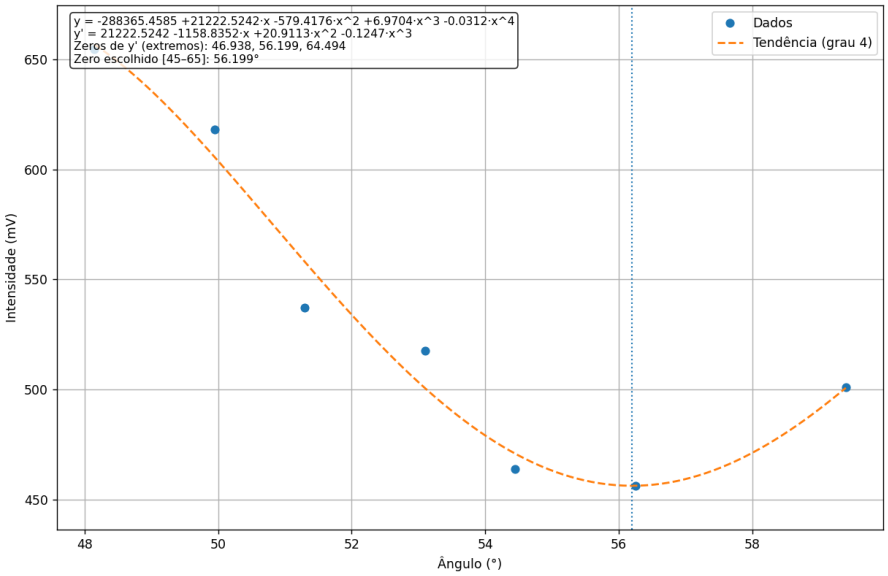


Fig. 11 - Picos de intensidade para o laser verde no semicírculo de acrílico

4.2 Medição em tecidos biológicos

Após a validação, o sistema foi aplicado à medição de amostras de tecido muscular de peito de frango, com o objetivo de analisar a influência da anisotropia das fibras no índice de refração. Foram realizadas medições com fibras orientadas horizontalmente e verticalmente em relação ao feixe incidente, utilizando o laser verde (520 nm).
Nos ensaios com fibras horizontais, as curvas de intensidade apresentaram vales bem definidos próximos de 54,3°, resultando em um índice de refração médio de 1,393 ± 0,03. Já para as fibras verticais, o ângulo mínimo situou-se em 54,1°, correspondendo a $n = 1{,}375 \pm 0{,}002$.
Os resultados confirmam o comportamento anisotrópico do tecido muscular, uma vez que a orientação das fibras afeta a propagação e reflexão da luz polarizada. O valor de índice de refração associado ao peito de frango com as fibras orientadas na horizontal encontra-se em concordância com os reportados por Sun e Sun no seu estudo[9], que determinaram índices de 1,3949. Já o valor de índice de refração associado ao peito de frango com as fibras orientadas na vertical encontra-se ligeiramente acima ao reportado na teoria (1,3540).

5 Conclusão

O sistema desenvolvido demonstrou ser uma solução fiável, acessível e totalmente automatizada para a determinação do índice de refração de tecidos biológicos com base no ângulo de Brewster. A integração entre controlo motorizado, aquisição sensorial e análise computacional permitiu realizar medições precisas sem intervenção manual, assegurando elevada estabilidade e repetibilidade.

A inovação central do protótipo reside na combinação num único dispositivo de baixo custo de todas as etapas da medição, desde o varrimento angular até ao cálculo automático do índice de refração. Esta abordagem contrasta com métodos tradicionais que exigem equipamentos complexos, calibração rigorosa ou análises manuais. O sistema oferece ainda portabilidade e flexibilidade para utilização em contextos laboratoriais ou didáticos, podendo contribuir para técnicas biomédicas que dependem da caracterização ótica não invasiva de tecidos.

Como perspetiva futura, a incorporação de polarização controlada, fontes laser mais estáveis e sensores de maior sensibilidade poderá ampliar o desempenho e reduzir o tempo de aquisição. A aplicação do sistema a diferentes tipos de tecido e ensaios in vivo abre caminho ao desenvolvimento de ferramentas clínicas para diagnóstico ótico e análise de propriedades estruturais dos tecidos.

6 Referências

1. Sliney, D.H.: What is light? The visible spectrum and beyond. Eye. 30, 222–229 (2016).
2. Tuchin, V.V.: Tissue Optics and Photonics: Light-Tissue Interaction. JBPE. 98–134 (2015).
3. Holland, I., Davies, J.A.: Automation in the Life Science Research Laboratory. Front. Bioeng. Biotechnol. 8, 571777 (2020).
4. Li, H., Xie, S.: Measurement method of the refractive index of biotissue by total internal reflection. Appl. Opt., AO. 35, 1793–1795 (1996).
5. Shimojo, Y., Nishimura, T., Hazama, H., Ozawa, T., Awazu, K.: Measurement of absorption and reduced scattering coefficients in Asian human epidermis, dermis, and subcutaneous fat tissues in the 400- to 1100-nm wavelength range for optical penetration depth and energy deposition analysis. J. Biomed. Opt. 25, 1 (2020).
6. Dirckx, J.J.J., Kuypers, L.C., Decraemer, W.F.: Refractive index of tissue measured with confocal microscopy. J. Biomed. Opt. 10, 044014 (2005). https://doi.org/10.1117/1.1993487.
7. Torres-García, E., Oros-Pantoja, R., Aranda-Lara, L., Vieyra-Reyes, P.: A new Monte Carlo code for light transport in biological tissue. Med Biol Eng Comput. 56, 649–655 (2018).
8. Leino, A.A., Pulkkinen, A., Tarvainen, T.: ValoMC: a Monte Carlo software and MATLAB toolbox for simulating light transport in biological tissue. OSA Continuum. 2, 957 (2019).
9. Sun, P., Sun, H.: Determination of the anisotropy complex refractive indices of chicken tissues in vitro at 650 nm. J. Europ. Opt. Soc. Rap. Public. 5, 10030 (2010).

Projeto e Desenvolvimento de um Mini Frigorífico e uma App de registo da sua temperatura com recurso a Microcontrolador Arduino

Beatriz Resende[1], José Magalhães[2], Alberto Peixoto Pinto[2], Carolina Costa[1], Inês Ferreira[1], João Aroso[1], Christopher Sá[2]

[1] Alunos do Instituto Superior de Engenharia, Politécnico do Porto, Porto, Portugal
1241197@isep.ipp.pt, 1241084@isep.ipp.pt, 1241072@isep.ipp.pt, 1240763@isep.ipp.pt
[2] Professores do Instituto Superior de Engenharia, Politécnico do Porto, Porto, Portugal
jdm@isep.ipp.pt, cas@isep.ipp.pt, apo@isep.ipp.pt

Resumo. Este trabalho descreve uma atividade de aprendizagem com a realização de um projeto sobre a criação de um sistema de refrigeração termoelétrica portátil, fundamentado no efeito Peltier, por um grupo de alunos de Engenharia de Sistemas do Instituto Superior de Engenharia do Porto. O estudo visa o desenvolvimento de um protótipo de mini-frigorífico controlado eletronicamente e por telemóvel, capaz de arrefecer bebidas de lata autonomamente. A metodologia adotada iniciou pela modelação 3D do protótipo e simulação dos seus circuitos elétricos em ambiente virtual (Tinkercad), passando depois para a construção do mesmo em XPS incluindo o microcontrolador Arduino e sensores de temperatura e distância. Adicionalmente, foi desenvolvida uma App de monitorização remota no MIT App Inventer para via Bluetooth ser capaz de adquirir e visualizar os valores da temperatura em tempo real. Os ensaios experimentais comprovaram a eficiência do sistema, registando-se um decréscimo de temperatura de 5°C numa bebida de lata de 33 cL em aproximadamente 6 minutos. O trabalho valida a aplicação de módulos termoelétricos em sistemas de refrigeração compactos e demonstra a integração funcional entre princípios termodinâmicos e sistemas embebidos de controlo. A metodologia de aprendizagem utilizada foi a Project Based Learning (PBL) que aplica a componente prática de *hands-on* em princípios de engenharia e o reforça as competências dos alunos para o trabalho em equipa.

Palavras-chave: Refrigeração Termoelétrica; Efeito Peltier; Monitorização Remota; Prototipagem Eletrónica; Arduino; MIT app Inventor.

1 ENQUADRAMENTO e OBJETIVOS

A refrigeração é um processo fundamental em diversas aplicações industriais e domésticas, essencial para a conservação de alimentos, manutenção de ambientes climatizados e funcionamento de equipamentos eletrónicos. Em particular, os sistemas de refrigeração termoelétrica de pequena escala podem ser usados para manter fresco bebidas ou transporte medicamentos. Nesse âmbito, na unidade curricular (UC) de Laboratórios

de Engenharia I (LENG1), curso de licenciatura de Engenharia de Sistemas (LES) do Instituto Superior de Engenharia do Porto (ISEP), ano letivo 2024/25, foi pedido aos alunos do 1º que realizassem um protótipo de um mini frigorifico portátil que fosse capaz de manter bebidas de lata frescas. O objetivo principal do projeto foi desenvolver um protótipo com capacidade entre 2 e 5 litros. Especificamente, pretendeu-se:

- Conceber e simular o sistema em ambiente virtual (Tinkercad);
- Implementar fisicamente o protótipo com integração de componentes eletrónicos na plataforma Arduino;
- Desenvolver um sistema de monitorização remota via Bluetooth;
- Validar experimentalmente a eficiência térmica do sistema.

O protótipo de um mini frigorifico, de design compacto, tem cerca de 3,4 litros e foi desenhado tridimensionalmente à escala, com o auxílio do software Tinkercad, permitindo a sua pré-visualização, seleção dos materiais necessários à sua construção e o dimensionamento dos circuitos elétricos.

Baseado na plataforma eletrónica Arduino construiu-se o protótipo, montaram-se os componentes eletrónicos e criou-se a App de monitorização de todo sistema por telemóvel via Bluetooth. Na App usou-se programação em código C++ na IDE do Arduino e programou-se por blocos na MIT app Inventor para design e comunicação do sistema com o telemóvel. A última fase foi o teste do sistema de refrigeração e a realização de uma prova de funcionamento perante os professores e colegas.

Este trabalho destaca a importância da integração de diversas áreas do conhecimento de engenharia de sistemas para a resolução de problemas práticos e a criação de soluções inovadoras. O projeto apresentou uma abordagem abrangente e multidisciplinar, reforçando o papel da engenharia no desenvolvimento de sistemas tecnológicos modernos e sustentáveis.

2 Referencial Teórico

Efeito Peltier/Seebeck e Refrigeração Termoelétrica.

O efeito Peltier consiste na criação de um gradiente de temperatura entre dois diferentes materiais ou semicondutores, chamados de termopar, quando estes são submetidos a uma tensão elétrica em circuito fechado (Fig. 1 B) [1].

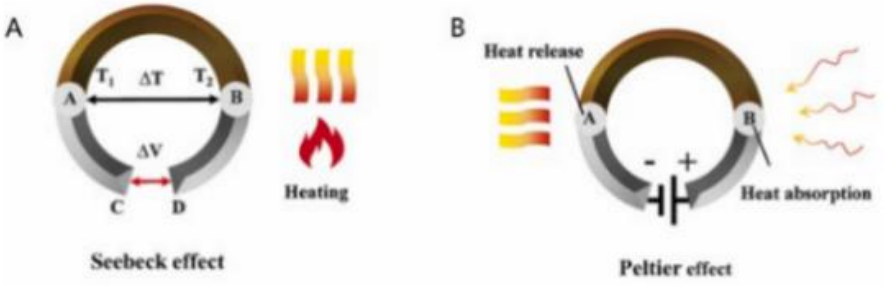


Fig. 1. Efeito Peltier-Seebeck (T_1 e T_2 valores de temperatura)

O efeito Seebeck (Fig. 1 A) é o reverso do Peltier, ou seja, quando se aplica uma diferença de temperatura entre diferentes materiais gera-se uma diferença de potencial (ΔV) que pode ser mensurada. Este efeito é usado em aplicações de geração de energia termoelétrica (TE) (Fig. 2 C) enquanto o efeito de Peltier usa-se na refrigeração TE, ilustrada na Fig. 2 D. Os dois efeitos são considerados como um só e denominado de efeito Peltier-Seebeck ou efeito termoelétrico.

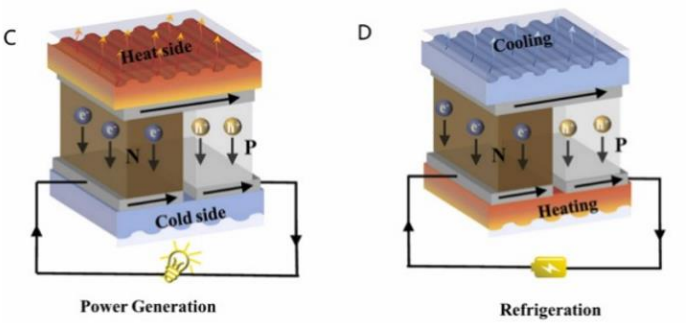


Fig. 2. Geração de energia (C) ou Refrigeração (D) por efeito termoelétrico [1].

Os sistemas de refrigeração convencionais com recurso a bomba de calor (compressor e válvula de expansão) apresentam limitações em aplicações de pequena dimensão, daí que no contexto deste trabalho a refrigeração termoelétrica surgiu como uma alternativa viável. As vantagens da refrigeração termoelétrica incluem a ausência de compressor, controlo preciso de temperatura e maior durabilidade [2].
Na refrigeração de pequena dimensão podem ser usados os módulos Peltier ou pastilhas termoelétricas (Fig.3) e que são pequenas unidades que contém uma série de semicondutores agrupados em pares, para operarem como bombas de calor com espessura de alguns milímetros e de forma quadrada. Esses módulos são pequenos cubos de telureto de bismuto ensanduichados por de placas cerâmicas [3].

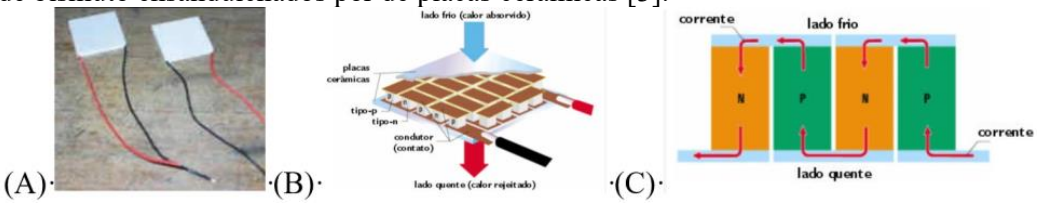


Fig. 3. Módulo Peltier: (A) aspeto comercial; (B) modelo; (C) Esquema [3].

Essa série de semicondutores é soldada entre duas placas cerâmicas, eletricamente em série (Fig. 3 C) e termicamente em paralelo. Quando uma corrente continua passa por um ou mais pares, há uma redução na temperatura da junta ("lado frio") resultando em uma absorção do calor do ambiente, atendendo à 2ª lei da termodinâmica que justifica que o calor flui naturalmente do corpo mais quente para o mais frio. Este calor é transferido pela movimentação de eletrões na pastilha, e é proporcional à corrente e ao número de pares de elementos tipo-n e tipo-p [4].
O rendimento termodinâmico de um módulo Peltier ou eficiência energética de um refrigerador pode ser quantificado pelo Coeficiente de Performance (COP), dada pela

relação COP = Qc/W, onde Qc é o calor removido do lado frio e W é a potência elétrica que lhe é fornecida [2][4]. Em sistemas Peltier, o COP típico varia entre 0,3 e 0,7, podendo atingir valores próximos de 1 em configurações que otimizam a dissipação de calor com dissipadores [4].

Aplicações com Microcontroladores para Monitorização e Controlo.

Arduino é uma plataforma de prototipagem eletrónica de código aberto, baseada em microcontrolador, que permite a criação de projetos interativos de forma acessível e intuitiva. Esta plataforma foi concebida e desenvolvida por Massimo Banzi e alguns colegas com o objetivo de facilitar a todos o acesso à tecnologia de microcontroladores[5].
A integração de microcontroladores em sistemas de refrigeração permite controlo preciso e monitorização em tempo real [6]. A comunicação em Bluetooth oferece uma solução de baixo custo para transmissão de dados sem fios, amplamente utilizada em aplicações IoT (Internet das Coisas). É possível encontrar diversos trabalhos [7] [8] que utilizam Arduino para controle de módulos Peltier com algoritmos PID e integração aplicações para telemóvel via App Inventor ou frameworks IoT.
Plataformas de desenvolvimento visual como MIT App Inventor permitem a criação rápida de interfaces de utilizador para monitorização de sistemas embebidos através de programação por blocos, democratizando o acesso sem fios à tecnologia de prototipagem [9].

3 MATERIAIS e METODOLOGIA

3.1 Componentes do Sistema

De seguida apresentam-se os principais componentes usados na construção do protótipo do mini frigorifico (Fig. 4) e as suas respetivas funcionalidades:

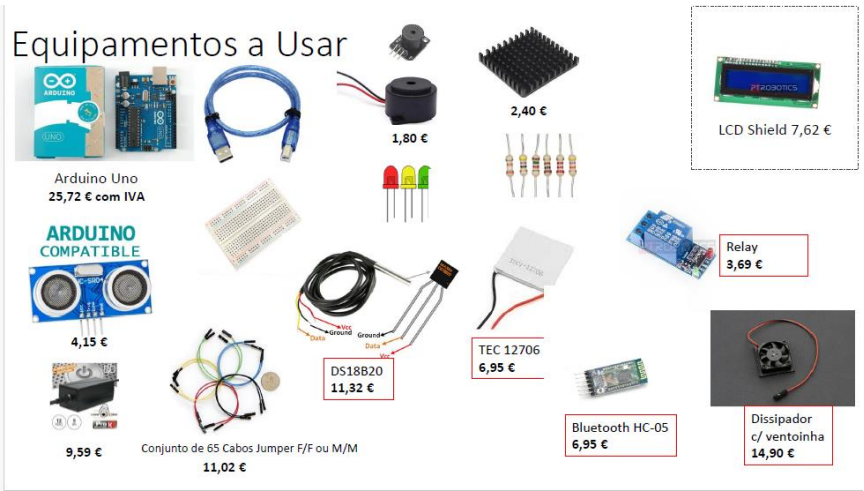


Fig. 4. Componentes principais integrados no protótipo

- O Módulo ou placa de Peltier (TEC-12706) alimentado por uma fonte externa de 12v foi o componente de base para o sistema de refrigeração e é responsável pela geração de um diferencial de temperatura.
- O controlo de todos os componentes e sensores do protótipo foi usado o Arduino Uno.
- O sensor de distância ultrassónico (HC - SR04) permitiu detetar a presença e determinar a altura do objeto no interior do mini frigorifico.
- O sensor de temperatura (DS18B20) serviu para medir a temperatura do líquido no interior da lata de bebida.
- A placa Bluetooth (HC-05) permitiu a transmissão em tempo real de dados captados pelos sensores para a aplicação construída para telemóvel.
- O dispositivo sonoro, Buzzer, foi usado e ativado pelo Arduino para produzir diferentes sons pré-definidos conforme a ação ou alarme pretendido.
- A fonte de alimentação externa de 12V serviu a placa de Peltier,
- O relay (SRD-05VDC-SL-C) usou-se como interruptor *on/off* da placa.
- O dissipador com Ventoinha (Air cooler AMD socket AM2/AM3) esteve em contacto com a parte quente da placa Peltier.
- A pequena ventoinha junto à face da placa virada para o interior do frigorifico permitiu a uniformização da temperatura do ar frio no sistema de refrigeração interior.
- Alguns leds foram usados para controlo visual do sistema.
- As placas de XPS foram usadas na construção das paredes do refrigerador e a cortiça usada como isolante entre o XPS e a face quente do módulo Peltier.
- A madeira foi usada como estrutura para suportar o dissipador de calor e a ventoinha.
- A pasta térmica usou-se entre o dissipador e a placa de Peltier para melhorar a condução térmica entre essas superfícies.
- A fita cola alumínio serviu para evitar perdas de frio e a goma EVA para revestimento e isolamento do frigorífico.
- Os ímanes foram usados como fecho para a abertura de porta do frigorífico.

3.2 Metodologia

Descrevem-se de seguida as fases de implementação e realização do protótipo.

Fase de Conceção Virtual.

O desenvolvimento do protótipo iniciou-se com o desenho e a modelação virtual 3D em Tinkercad, assim como a sua simulação e a criação dos circuitos elétricos (Fig. 5)

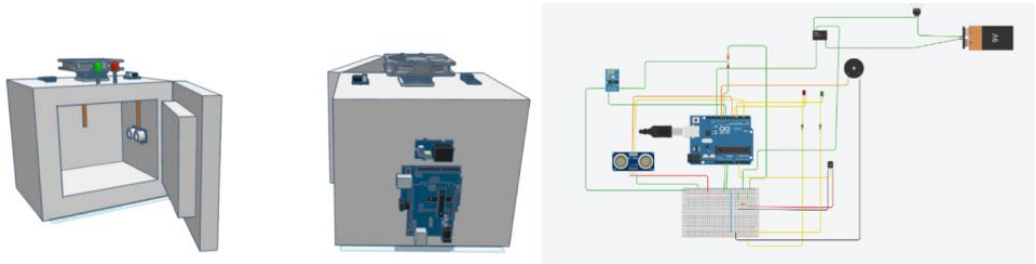


Fig. 5. Modelo 3D do protótipo em Tinkercad

Este Modelo permitiu:
- Visualização prévia da disposição dos componentes eletrónicos internos;
- Validação e simulação do circuito elétrico antes da implementação física;
- Otimização do isolamento térmico e dimensionamento dos compartimentos;
- Identificação de potenciais problemas de compatibilidade.

Fases de Construção Física.

Em seguida identificam-se os passos de construção protótipo físico do mini frigorifico.
Corte e preparação da estrutura: Realização de cortes em placa de XPS (poliestireno expandido) de acordo com as dimensões especificadas (3,4L) (Fig. 6 A);
Testes individuais: Validação de funcionamento de cada componente (Fig. 6) antes da integração no circuito completo (Fig. 7);

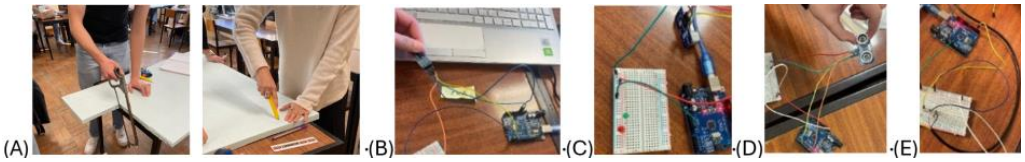


Fig. 6. Cortes (A) e Teste de sensores: Bluetooth (B), Leds(C); Distância (D) e Temperatura (E)

Montagem do circuito: Conexão de todos os componentes (Fig. 7);

Fig. 7. Circuito total

Integração térmica: Aplicação de pasta térmica entre o módulo Peltier e os dissipadores, vedação com fita alumínio e Goma EVA da estrutura do refrigerador.

Desenvolvimento do Firmware.

Para o controlo e operar todos os componentes do protótipo do mini frigorifico de forma integrativa desenvolveu-se código C++ na IDE do Arduino para:
- **Leitura de temperatura no interior frigorífico:** inclusão das bibliotecas OneWire e DallasTemperature para comunicação com o sensor DS18B20 via protocolo 1-Wire (Fig. 8). Foi definido o pino digital 2 do Arduino como porta de comunicação deste sensor;

```
#include <OneWire.h>
#include <DallasTemperature.h>

// Definição do pino de dados para o sensor DS18B20
#define ONE_WIRE_BUS 2 // Sensor conectado ao pino digital 2

// Instâncias para comunicação com o sensor
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
```

Fig. 8. Código de configuração para sensor de temperatura

- **Controlo do relé:** Optou-se por manter o relé continuamente ativo (ligado) para funcionamento contínuo da placa Peltier, pois testes comparativos demonstraram maior eficiência relativamente ao funcionamento intermitente (on/off);
- **Deteção de objetos:** A leitura de distância com o sensor HC-SR04 determina a presença/ausência de objetos (limiar: 12 cm de altura) e liga o led vermelho na ausência de objeto (lata de bebida) ou verde na sua presença (Fig. 9);

```
// Controle dos LEDs baseado na distância
if (distance >= 12) {
  digitalWrite(ledRed, HIGH);
  digitalWrite(ledGreen, LOW);
} else {
  digitalWrite(ledRed, LOW);
  digitalWrite(ledGreen, HIGH);
}
```

Fig. 9. Código de aviso visual presença/ausência de objetos no interior frigorifico

- Indicadores de estado: LED vermelho se frigorífico vazio ou verde se presente;
- Alerta sonoro: Emissão de som a 2000 Hz durante 5 segundos quando detetada variação de temperatura ≥ 5°C no interior do frigorifico (Fig. 10);

```
// Controle do buzzer baseado na temperatura
if (delta_temp == 5) { // Verifica se ΔT = 5°C
  tone(buzzer, 2000);  // Toca com 2000 Hz
  delay(500);          // Aguarda 500 ms
  noTone(buzzer);      // Para o som
} else {
  noTone(buzzer);      // Garante que o som está parado
}
```

Fig. 10. Código de aviso sonoro

Desenvolvimento da Aplicação de Monitorização em Telemóvel.
Desenvolveu-se uma aplicação móvel (App) em MIT App Inventor (Fig. 11) para comunicação via Bluetooth com as seguintes funcionalidades:
- Receção de dados de temperatura em tempo real via Bluetooth;
- Visualização de valores instantâneos de temperatura, hora e rendimento;
- Geração de gráficos temporais de variação de temperatura no interior frigorifico;
- Interface intuitiva para interação do utilizador

Fig. 11. Wireframe da aplicação para telemóvel

Testes experimentais e Protótipo final.

O funcionamento do protótipo final (Fig. 12) foi testado no dia 28 de janeiro de 2025 numa prova prática seguindo o seguinte protocolo:
- Verificação prévia de bom funcionamento de todos os componentes e da App;
- Medição e registo da temperatura inicial do ambiente;
- Colocação do sensor de temperatura no interior da lata de coca cola e ativação do sistema de refrigeração;
- Monitorização contínua da temperatura até atingir variação de 5°C;
- Registo temporal e análise dos dados obtidos.

Fig. 12. Protótipo final do mini frigorifico

4 Resultados

A prova teste de verificação do funcionamento (com cronómetro e termopar) do protótipo demonstrou que o sistema alcança uma redução de temperatura de 5°C em aproximadamente 6 minutos, reduzindo a temperatura do líquido na lata de 24°C para 19°C. Este resultado valida a eficiência operacional do protótipo.
Os valores de temperatura do líquido ao longo do tempo recebidos pelo Arduino estão indicados no gráfico (Fig. 13). O gráfico mostra uma queda progressiva dos valores de temperatura, indicando que o mini frigorífico está a arrefecer. As pequenas variações de temperatura seguidas de estabilidade sugerem um funcionamento intermitente do sistema de refrigeração.

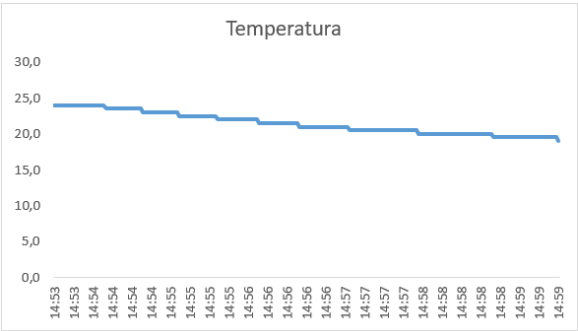


Fig. 13. Gráfico de temperatura líquido ao logo do tempo (Ox horas: minutos; Oy °C)

A figura (Fig.13) mostra que o intervalo de variação (negativa) da temperatura do líquido aumenta gradualmente ao longo do tempo, assim como a diferença entre as temperaturas inicial e final corresponde a 5ºC, evidenciando um padrão de arrefecimento por etapas devido à leitura das amostras de forma discreta.

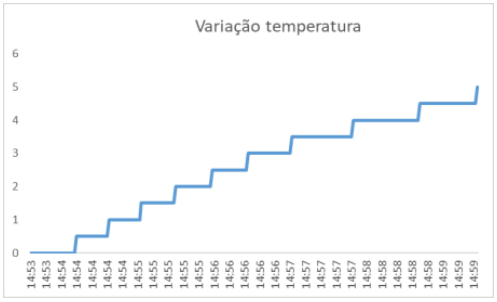


Fig. 14. Variação temperatura do líquido em função do tempo (Ox horas: minutos; Oy °C)

A tendência da evolução temporal da temperatura (Fig. 14) mostra que o sistema não diminui linearmente, sem quedas abruptas, o que demonstra que não houve interferências externas significativas, garantindo um arrefecimento estável e eficiente.
O rendimento do arrefecimento (η) do sistema foi monitorizado ao longo tempo, onde $\eta = P_L/W$, em que P_L é a potência de arrefecimento e W a potência elétrica fornecida ao sistema (máximo de 60 W). Esta potência de arrefecimento é a quantidade de calor que se consegue remover do líquido por unidade de tempo (leituras em intervalos de tempo fixo), expressa em Watts. A quantidade de calor removida (Q) é dada pela expressão $Q = m \cdot c_p \cdot \Delta T$, onde m é a massa do líquido (em kg), c_p o calor específico do líquido (em J/kg·K) e ΔT a variação de temperatura (em °C ou K).
O rendimento do arrefecimento do sistema, conforme se pode visualizar na Fig. 15, cresceu de forma constante ao longo do tempo, mas não aumenta de forma completamente linear.

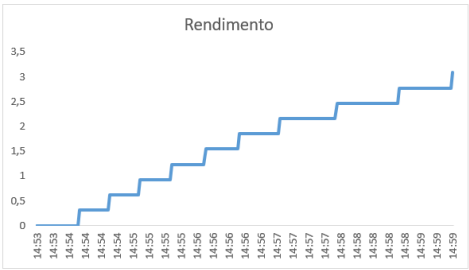


Fig. 15. Rendimento em função do tempo (eixo Ox horas: minutos, eixo Oy rendimento)

Esta figura (Fig. 15) mostra que a eficiência do sistema aumenta de forma progressiva e por fases, atingindo desempenho ideal após aproximadamente 6 minutos de operação. Este comportamento sugere que o sistema requer um período de estabilização inicial para otimizar a dissipação térmica.

5 Conclusões

A integração de conceitos teóricos de termodinâmica com tecnologias práticas de prototipagem eletrónica resultou num protótipo operacional que cumpriu os objetivos estabelecidos.

A placa Peltier foi uma solução tecnicamente viável para refrigerar bebidas de pequena dimensão, a integração de sensores e sistemas de controlo permitiu operação autónoma e a monitorização em tempo real dos valores de temperatura e, por fim, o protótipo diminui a temperatura com uma velocidade de arrefecimento de 0,83°C/min.
Importante será referir que a integração de um dissipador com ventoinha no lado quente da placa Peltier foi determinante para o sucesso do sistema, assim como o uso de um dissipador interno com ventoinha ativa para a homogeneização da temperatura no interior do refrigerador.
A App realizada no MIT App Inventor forneceu uma interface acessível para monitorização da temperatura em tempo real, com visualização gráfica dos seus valores, o que permitiu identificar padrões de funcionamento.
O projeto ilustrou a importância da abordagem multidisciplinar na engenharia, combinando os princípios da física, design estrutural, eletrónica e desenvolvimento de software. A metodologia de aprendizagem adotada, desde a conceção virtual, simulação até à validação experimental, demonstra-se adequada para desenvolvimento de sistemas em contexto educacional. A experiência adquirida pelos alunos reforçou a relevância da aprendizagem prática na consolidação de conhecimentos teóricos e no desenvolvimento de competências de resolução de problemas, essenciais para a formação de engenheiros competentes e inovadores.
Como sugestão final refere-se o aumento da velocidade de refrigeração com o uso de mais placas Peltier e o desenvolvimento da App com mais funcionalidades avançadas de análise de dados.

Referências

1. Q. Sun, C. Du e G. Chen, "Thermoelectric materials and devices Applications in enhancing building," Advanced Nanocomposites, vol. 2, pp. 15-31, 2025
2. D. M. Rowe, *Thermoelectrics Handbook: Macro to Nano*. CRC Press, 2006.
3. Portal de refrigeração, [Online] Available: https://refrigeracao.net/topicos/refrigeracao_eletronica
4. H. A. Mahmood et al., "Improving the Performance of Thermoelectric Systems for Air Conditioning Using Different Configurations," Fluid and Heat Management Technology, 2025. DOI: 10.32604/fhmt.2025.066075
5. Arduino. *About Arduino*. [Online]. Available: https://www.arduino.cc/en/about/, last accessed 2025/10/24
6. S. Monk, *Programming Arduino: Getting Started with Sketches*. McGraw-Hill, 2013.
7. A. Santana et al., "Development of an Electrical circuit with Arduino implementation to automate a Peltier Thermoelectric cooling system," IJASRE, vol. 9, no. 9, 2023. DOI: 10.31695/IJASRE.2023.9.9.3
8. M. M. Martínez Valencia, "Temperature Control and Monitoring with Peltier Cells," IEEE Student Project Report, UPC, 2024. [Online]. Available: https://upcommons.upc.edu
9. Wolber, D., Abelson, H., Spertus, E., & Looney, L. (2011). *App Inventor: Create Your Own Android Apps*. O'Reilly Media

Controlo e Simulação de um Sistema Fotovoltaico a Instalar no Edifício G do ISEP

Marta Santos[1], José M. Magalhães[2], Edvaldo Tavares[1], Thierry Ramos [1] e Alberto Peixoto Pinto[2]

[1] Alunos do Instituto Superior de Engenharia, Politécnico do Porto, Porto, Portugal
1222191@isep.ipp.pt, 1231639@isep.ipp.pt, 1231640@isep.ipp.pt
[2] Professores do Instituto Superior de Engenharia, Politécnico do Porto, Porto, Portugal,
jdm@isep.ipp.pt, apo@isep.ipp.pt

Resumo. O presente artigo descreve o desenvolvimento e simulação de um sistema de produção de energia fotovoltaica e gestão de iluminação do edifício G do ISEP, no âmbito da Licenciatura de Engenharia de Sistemas Elétricos de Energia. A metodologia adotada, fundamentada em Project-Based Learning (PBL), integrou a modelação matemática da trajetória solar com recurso ao GeoGebra, a modelação 3D do sistema em Tinkercad e a prototipagem eletrónica na plataforma Arduino. Foi projetado um sistema de rastreamento solar (solar tracker) controlado por microcontrolador, visando a otimização da captação energética, paralelemente a um sistema de iluminação automatizado baseado em sensores de luminosidade e movimento. As simulações em ambiente virtual (Tinkercad) e a validação experimental através de um protótipo físico demonstraram a funcionalidade dos algoritmos de controlo propostos. Os resultados indicam viabilidade técnica e a sua análise identifica a necessidade de dados de consumo mais detalhados do edifício G para um dimensionamento energético de melhor precisão e análise económica rigorosa.

Palavras-chave: Sistemas Fotovoltaicos; Seguidor Solar; Automação e Controlo; Prototipagem Eletrónica; Arduino.

1 ENQUADRAMENTO e OBJETIVOS

Atualmente existe uma necessidade cada vez maior da utilização da eletricidade e com este aumento existe também uma crescente preocupação com os problemas ambientais devido ao uso excessivo de combustíveis fósseis e não renováveis.
No âmbito das unidades curriculares de Laboratórios de Matemática I (LMAT1) e Métodos de Engenharia (MTENG), do 1º ano do curso de Engenharia de Eletrotécnica – Sistemas Elétricos de Energia do ISEP, ano letivo 2023/24, prepôs-se aos alunos a realização de um estudo e apresentação de propostas de implementação e simulação na plataforma Arduino de um sistema de produção energética fotovoltaico e de controlo de iluminação para o Edifício G do ISEP para melhoria da sua eficiência energética.

Com esse objetivo foi necessário procurar soluções que diminuam ou que atenuam os problemas ambientais. Apresentam-se duas soluções, uma para a utilização de sistemas de produção de energia fotovoltaica em edifícios para autoconsumo e outra para uma aplicação de iluminação mais eficiente, com troca para lâmpadas com menos consumo energético e usando luminárias inteligentes e de controlo automatizado.
Sustentadas nas metodologias PBL (Problem-based Learning) [1] [2] e PJBL (Project-based Learning) [3], com este projeto obtiveram-se conhecimentos de como implementar e simular um sistema de produção energética fotovoltaico, formas de controlo de iluminação e realizou-se a construção de um protótipo que representava proposta de solução para o Edifício G do ISEP. Neste âmbito, como não existem dados dos consumos energéticos do Edifício G, e como o Edifício H é semelhante em estrutura e em consumos, utilizaram-se os dados relativos ao consumo energético do Edifício H e assumindo como sendo do Edifício G. As simulações foram realizadas na aplicação Tinkercad e a construção do protótipo do sistema produção energética fotovoltaico e de controlo de iluminação a implementar teve como recurso a plataforma Arduino.

2 Referencial Teórico

A crescente procura energética mundial, conjugada com a urgência de mitigar as alterações climáticas, tem impulsionado a transição para fontes de energia renovável, destacando-se a energia solar fotovoltaica como uma das tecnologias mais promissoras do século XXI [4]. Nos últimos anos, os custos associados à produção de energia dos sistemas fotovoltaicos registaram uma redução de aproximadamente 89% entre 2010 e 2021, tornando esta tecnologia competitiva ou até mais económica do que as fontes mais convencionais baseadas em combustíveis fósseis em muitas regiões do globo. Esta competitividade económica, conjugada com benefícios ambientais significativos (emissões zero de gases) posiciona a energia fotovoltaica como elemento fundamental na descarbonização do setor energético global [4].
Para se poder gerar energia fotovoltaica é essencial a utilização de painéis fotovoltaicos. Estes painéis são compostos por células fotovoltaicas e constituídas por materiais semicondutores, por exemplo o silício, que absorvem a luz solar e geram energia elétrica (efeito fotovoltaico) [5], como se ilustra na Fig. 1.

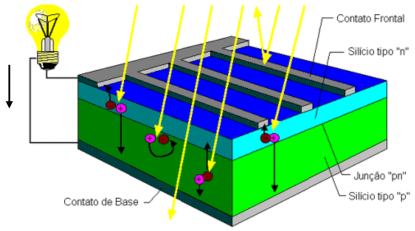


Fig. 1. Conversão de radiação solar em eletricidade.

De acordo com a literatura [6], existem vários tipos de células, silício, silício amorfo (sem forma), micromorfas e híbridas, no entanto neste trabalho vamo-nos focar nas células de silício cristalino. Estas podem ser monocristalinas ou policristalinas. Os painéis monocristalinos apresentam eficiências entre os 16% e os 19% e é normalmente

aplicada em média e elevada potência, mas tem um custo elevado de produção Os painéis policristalinos tem um custo inferior de produção, mas tem uma eficiência também inferior variando entre os 13% e os 16%. (Fig. 2)

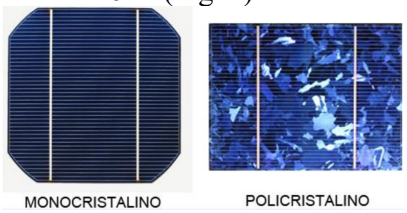


Fig. 2. Aspeto de células monocristalinas e policristalinas.

Segundo estudos [7], o autoconsumo é uma das formas de aplicar a energia elétrica produzida pelos painéis fotovoltaicos. Através de um sistema de autoconsumo fotovoltaico (Fig. 3) consegue-se utilizar parte da energia produzida diretamente nos equipamentos que necessitam de energia, carregar baterias para consumo posterior, injetar na Rede Elétrica de Serviço Público (RESP) ou vender o excedente celebrando um contrato com o comercializador de energia.

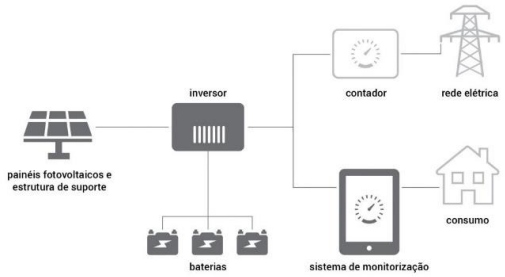


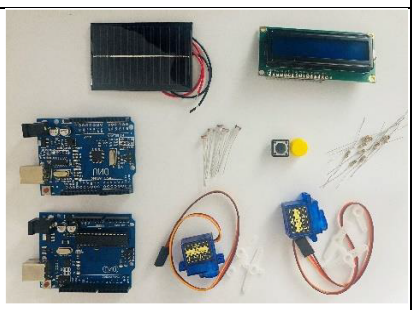
Fig. 3. Esquema de sistema de autoconsumo fotovoltaico [8].

3 METODOLOGIA e MATERIAIS

Para a concretização deste projeto, este grupo de estudantes aplicou diversas metodologias, iniciando pela pesquisa na Web de literatura sobre teorias e aplicações da energia fotovoltaica, bem como contacto com empresas em Portugal que fornecessem painéis fotovoltaicos. Posteriormente, foi feito um contacto com os serviços de manutenção do ISEP no sentido de conhecer os consumos enérgicos do edifício G e, consequentemente, é apresentada uma proposta de um sistema real de autoprodução de energia elétrica para edifício G com instalação de painéis fotovoltaicos e sugestões gerais sobre uma melhor gestão energética do edifício, em particular, a sua iluminação elétrica. Através da aplicação Tinkercad, plataforma online de modelação 3D e simulação de circuitos eletrónicos, realizou-se uma modulação 3D do Edifício G do ISEP com inclusão dos painéis fotovoltaicos e foram realizadas simulações de um circuito para o seguidor solar e de um sistema de controlo de iluminação do edifício de forma a melhorar a sua eficiência energética. Após estas simulações em software, foram executados os protótipos físicos representativos do controlo e monitorização de todo o sistema de autoprodução e gestão energética do edifício G. Para o seu controlo e programação foi

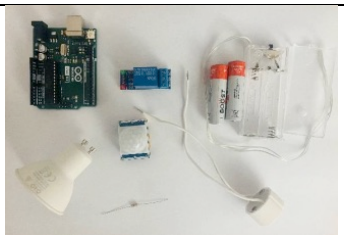
usada a plataforma Arduino por ser um sistema aberto (hardware e software) de prototipagem eletrónica e de baixo custo [9]. Esta plataforma utiliza uma linguagem de programação em código baseada em C++. Para a resolução da Tarefa 2 - simulação do movimento de rotação dos painéis fotovoltaicos formando um plano perpendicular com a direção da posição do sol - e do problema colocado na Tarefa 4 – aplicação de métodos numéricos para determinação da direção da posição do sol ao longo do dia em relação aos painéis fotovoltaicos - foi utilizada a aplicação GeoGebra. Esta aplicação consiste num software de matemática dinâmica e que combina os conceitos de geometria e álgebra.
Para realizar o protótipo foram utilizados os materiais referidos nas Tabelas 1 e 2.

Tabela 1. Materiais utilizados no protótipo do Seguidor Solar.

Seguidor Solar			
Nome	Un.	Descrição	
Motor Servo	2	Motor servo de 4.8V 180º	
Arduino Uno R3	2	Microcontrolador	
Sensor LDR	5	Foto resistência	
Resistências	10	Resistências de 220Ω, 2.2kΩ, 1.5kΩ, 100Ω e 10kΩ	
Painel Solar	1	Painel Solar de 0.5V	
LCD	1	LCD 16x2	
Botão	1	Botão 4 pinos	

De acordo com estudos [10], atualmente a iluminação artificial tem uma grande fatia do consumo de energia, desta forma é necessário o controlo desta iluminação para redução desses mesmos custos. Das várias formas de controlar a iluminação em edifícios, este projeto terá o foco na luminosidade ambiente, fazendo a verificação através de sensores de presença ou ausência de luz exterior para acender ou apagar lâmpadas e em sensores de movimento de pessoas que só acendem a luz quando se deteta movimento. Para isso, usou-se o conjunto de materiais que constam na lista na tabela 2.

Tabela 2. Materiais utilizados no protótipo do controlo de iluminação.

Controlo Iluminação			
Nome	Un.	Descrição	
Arduino Uno R3	1	Microcontrolador	
Pilhas	2	Baterias AA, 1.5V	
Suporte de pilhas	1	Suporte de pilhas 2xAA	
Relé	1	Relé de 5V	
Lâmpada	1	Lâmpada LED	
Casquilho Lâmpada	1	Casquilho para Lâmpada	
Sensor movimento	1	Módulo PIR HC-SR501	
Resistência	1	Resistência 1kΩ	

As escolhas das plataformas de modulação e simulação eletrónica - Tinkercad e Arduino - de cálculo e simulação matemática – GeoGebra - para a realização deste trabalho devem-se ao facto de serem abertas, gratuitas e de livre acesso a todos utilizadores e de serem as estudas na unidade curricular de Laboratórios de Matemática I.

4 Resultados e Discussão

Atendendo aos objetivos do seminário onde se apresenta esta comunicação, far-se-á de seguida uma breve referência aos resultados gerais da tarefa 1 e que consistiu na elaboração da proposta de um sistema real de autoprodução de energia fotovoltaica para edifício G do ISEP e posteriormente uma análise sobre os resultados obtidos na tarefa 5, que consistiu na construção da maquete/protótipo do sistema fotovoltaico e à sua programação do controlo e monitorização incluindo o sistema de iluminação do edifício.

Resultados e análise de resultados da tarefa 1 e 5

Analisando os consumos energéticos fornecido pelos serviços de manutenção do ISEP para o Edifício G, consumo anual do edifício, 84.542,301 kWh, verifica-se que existe uma necessidade energética diária aproximadamente de 234 kWh. Como tal, sugere-se recorrer a um sistema de autoconsumo com recurso a baterias e usar painéis fotovoltaico de marca Aiko Solar monocristalino de potência 0,610 kWp (kW-pico) por painel. Atendendo ao consumo diário do edifício G, seriam necessários aproximadamente 380 painéis solares de capacidade 610 Wp para satisfazer a sua necessidade energética e que daria a capacidade total instalada,

$$Capacidade\ da\ totatlidade\ dos\ painéis = 0{,}610 \times 380 = 230{,}80\ \text{kWp}$$

O próximo passo foi verificar se existia área suficiente no telhado do edifício para poderem serem instalados os 380 painéis solares. Através da aplicação Google Earth, apurou-se que a área bruta do telhado do edifício G que é aproximadamente 1134,41 m^2. Sendo as medidas, em mm, de um painel fotovoltaico de marca Aiko Solar 2278 x 1134 x 35, juntando o espaço necessário para a fixação dos suportes, cerca de 2 cm, calculando a área necessária por painel,

$$Área\ painel\ fixado = b \times h = 2{,}298 \times 1{,}154 = 2{,}65\ m^2$$

O que seria necessário para todos os 380 painéis uma área de telhado,

$$Área\ totalidade\ paineis = nº\ de\ paineis \times área\ de\ 1\ painel = 380 \times 2{,}65 = 1007\ m^2$$

Embora a área bruta do telhado seja suficiente para a instalação da totalidade dos painéis, mas uma vez que existem obstáculos e tendo em conta que deverá existir espaço para efeitos de manutenção, conclui-se que não será possível instalar a totalidade desses painéis. Neste sentido, considerando satisfazer cerca de 75% das necessidades diárias

do edifício (175,5 kWh) e uma área útil de máxima de 800 m^2. Pelos mesmo cálculos anteriores, verifica-se a necessidade 288 painéis, ocupando uma área total de 763,20 m^2. Solução estas que se acha adequada e que ocuparia menos espaço de telhado.
A alteração e controlo de iluminação irá permitir a redução do consumo energético do edifício G, a ser ainda apurada na prática. A solução passa ainda por alteração das luminárias, trocando as lâmpadas existentes, pouco eficientes, por luminárias e lâmpadas mais eficientes tipo LED e o controlo da iluminação por sensores de movimento e luminosidade. Na Fig. 4 está representada a vista real do edifício e a solução gráfica 3D construída no TinkerCad do sistema a implementar e que será a base da construção do protótipo final.

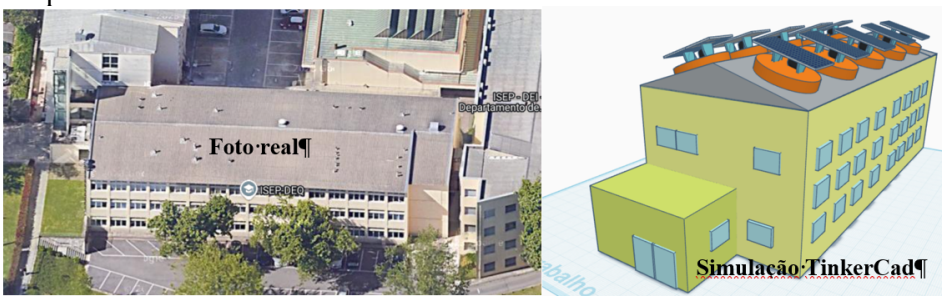


Fig. 4. Vista real do Edifício G e modelo 3D do sistema fotovoltaico no Tinkercad.

O sistema fotovoltaico a instalar no edifício G (Fig. 4) será rotativo e acompanhará o movimento do sol, de nascente a poente. O controlo do movimento dos painéis será feito pelo microcontrolador Arduino Uno R3 e que fará uma análise da incidência da luz solar sobre 4 células LDR (Light Dependent Resistor), colocadas em cruz, rodando-o numa linha perpendicular à posição do sol com o plano que contém os painéis fotovoltaicos. Na Fig. 5 apresenta-se o circuito eletrónico no TinkerCad do esquema elétrico implementado para o controlo de um desses painéis fotovoltaicos.

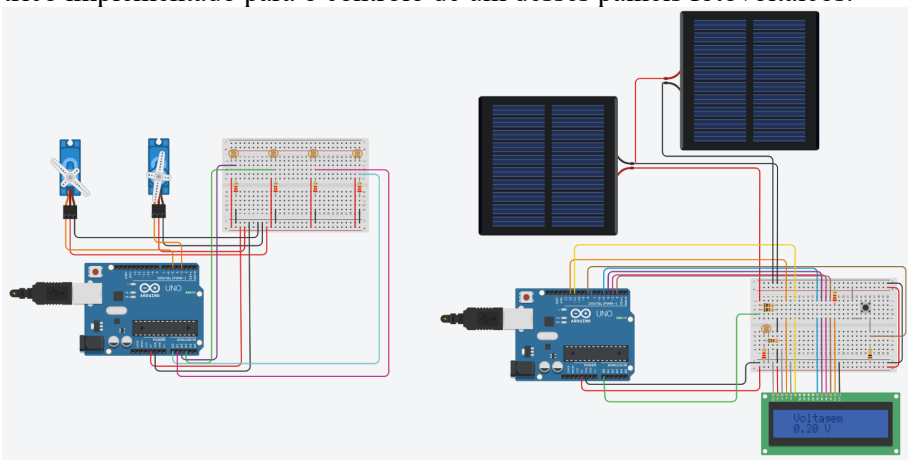


Fig. 5. Circuito eletrónico no TinkerCad para controlo do painel fotovoltaico.

Na tabela 3 pode-se ver um fragmento do código de programação para leitura de valores e cálculo de corrente e luz incidente que consta no programa de controlo do sistema fotovoltaico visualizado na Fig. 5.

Tabela 3. Fragmentos do código para o Arduino no controlo painel fotovoltaico.

Cód. Leitura de botão de Controlo	Cód. leitura do valor de incidência solar

if (digitalRead(8) == 1) { //lê o botão
var1 = 1; // Variavél de controlo = 1
delay (250);
}
while (var1 == 1) { // Corrente
lcd.setCursor(2,0);
lcd.print("Corrente ");
valorSensor=analogRead(sensor);
inten=(((valorSensor*0.00489)/50)*1000);
lcd.setCursor(2, 1);
lcd.print(inten);
lcd.print(" ");
lcd.print("mA ");
 	
if (digitalRead(8) == 1) { // Variavél de controlo = 2
var1 = 2;
delay (250);
}
}
	while (var1 == 3) { // Irradiação
	lcd.setCursor(2,0);
	lcd.print("Irradiacao ");
	lcd.write((byte)0);
	lcd.write("n ");
	valorSensor=analogRead(sensor);
	volt=(valorSensor*0.00489);
	irra=(volt*200);
	lcd.setCursor(2, 1);
	lcd.print(irra);
	lcd.print(" ");
	lcd.print("W/m2 ");
	
	if (digitalRead(8) == 1) { // Variavél de controlo = 4
	var1 = 4;
	delay (250);
	}

Já o controlo interno da luz nos corredores e/ou salas de aula é feito através de um sensor de movimento PIR (Passive Infrared Sensor) e por LDR. Na Fig. 6 apresenta-se o esquema elétrico do circuito eletrónico implementado no TinkerCad para o controlo de iluminação do edifício.

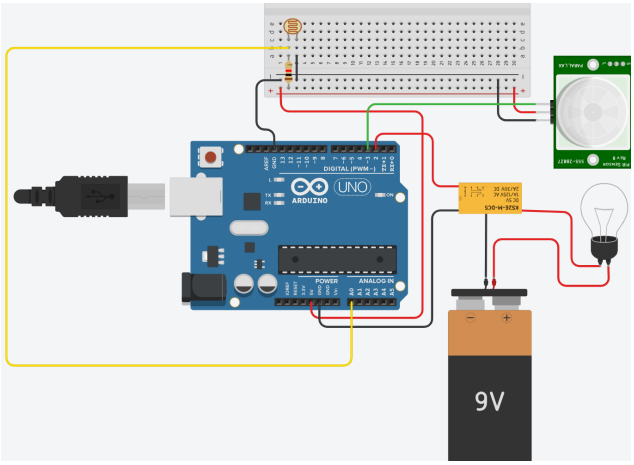


Fig. 6. Circuito eletrónico no TinkerCad para controlo de iluminação do edifício.

Na tabela 4 pode-se ver um fragmento do código para controlo de acionamento da iluminação de parte do edifício G e que consta no programa de controlo do sistema fotovoltaico visualizado na Fig.6.

Tabela 4. Fragmentos do código em C++ para de controlo de iluminação.

Código C++ de controlo de iluminação para Arduino

```
void loop() {
  int luminosidade = analogRead(pinoldr);
  Serial.print("Valor luminosidade = ");//Imprime na serial a mensagem Valor lido pelo LDR
  Serial.println(luminosidade);//Imprime na serial os dados de luminosidade
  int movimento = digitalRead(pinopir);

  if (luminosidade >= 600 && movimento == HIGH)  // Se não houver luz e houver movimento aciona
o relé
  {
    digitalWrite(pinorele, HIGH); // aciona relé
    delay(3000); // Deixa um tempo a luz ligada
  }

  else  // Caso contrário, desliga o relé
  {
    digitalWrite(pinorele, LOW);// desliga o relé
  }
 }
```

O projeto envolvia a construção de um protótipo físico que integrasse as funcionalidades e simulações referidas anteriormente. Para isso, todos os componentes do protótipo foram montados com apoio em *breadboards* e ligados aos Arduinos, como se vê na imagem da Fig. 7.

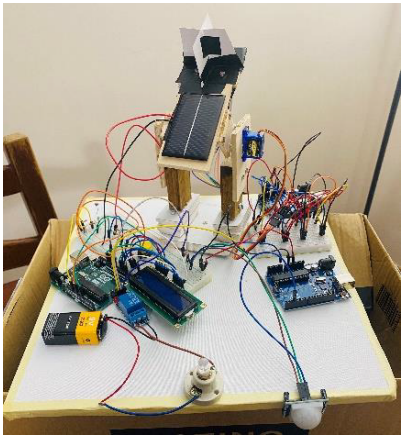


Fig. 7. Protótipo para simular e controlar o sistema fotovoltaico e de iluminação.

De referir que o protótipo apresentado na Fig. 7 apenas apresenta os componentes elétricos/eletrónicos que serão posteriormente integrados numa maquete física 3D, tal como se apresentou e se vê na figura 4. Esta foi a tarefa 5 da PBL e pôde-se verificar, numa escala reduzida, o funcionamento de um seguidor solar analisando o seu movimento em função da posição do foco de luz. Pôde-se também analisar como se comporta a iluminação numa sala com uso de um sensor de iluminação e um sensor de movimento. Verificou-se que a luz permanece desligada quando existe luminosidade e movimento na sala e, também, quando não existe luminosidade e não existe movimento. A luz acende quando existe movimento, mas a luminosidade é insuficiente.

5 Conclusões

O projeto de controlo e simulação de um painel fotovoltaico a instalar num edifício do ISEP que se descreveu neste documento foi lançado como uma atividade de aprendizagem nas unidades curriculares de LMAT1 e de MTENG, inseridas no plano curricular do 1º ano do curso de LESEE do ISEP. A realização das tarefas constante nessa atividade de aprendizagem teve a duração de 6 semanas letivas, com o objetivo de os alunos tomarem contacto com algumas competências exigidas a um engenheiro de Sistemas Elétricos de Energia. Nesse sentido, o projeto seguiu a metodologia de ensino e aprendizagem assentes no PBL [2] [4] [11] e em consonância com as recomendações do modelo CDIO [12] e EUR-ACE [13].

Os resultados obtidos nas tarefas 2, 4 e 5 foram ao encontro dos resultados esperados. Isto permite concluir que as simulações e o protótipo foram contruídos com sucesso, dando desta forma resposta ao que foi pretendido.

A tarefa 1, embora tenha sido concluída, pelo facto de ser um exercício académico, não se pode afirmar que foi realizado com total sucesso, uma vez que não existem dados reais comparativos para se poder dar uma resposta exata relativamente à quantidade de painéis possíveis de colocar no edifício G, não sendo possível desta forma definir as reias necessidades energéticas que poderíamos satisfazer e sua análise económica No caso da iluminação não se pode afirmar quais as luminárias e lâmpadas de deveriam trocar no edifício, uma vez que não se obteve uma relação do que está neste momento instalado e, por isso, também não se consegue obter qual a percentagem de poupança se poderia obter através desta troca. No entanto, atendendo que a direção do ISEP tem um projeto aprovado através do PRR para instalação de um sistema fotovoltaico para todos os edifícios da Instituição, em termos futuros, espera-se conseguir complementar e comparar os resultados obtidos deste trabalho com o que será instalado na realidade.

A atividade de projeto de controlo e simulação de um painel fotovoltaico permitiu aprender conceitos de funcionamento de painéis fotovoltaicos, da plataforma Arduino, de programação por blocos no TinkerCad e por código (C++) na IDE do Arduino, modelar em 3D (TinkerCad), simular e implementar o funcionamento de circuitos eletrónicos no TinkerCad, realizar modelação matemática e simular dinamicamente sistemas físicos em GeoGebra. A realização das tarefas do projeto permitiu ainda aprendizagem de comunicação de resultados em público e da escrita de relatório científicos.

Referências

1. W. Ekawati, "Implementing integrated project based learning to enhance students' writing skill" ELLITE: Journal of English Language, Literature, and Teaching, 3(2), 69, 2019.
2. J. Magalhães, M. Costa, A. Pinto, and C. Sá, "PBL as an Integration Activity for New Engineering Students at ISEP", In Academic Success in Higher Education. Proceedings of the CASHE conference in Porto from 7th to 8th April 2022.
3. R Tiwari, RK Arya, and M. Bansal, "Motivating students for project-based learning for application of research methodology skills", Int J App Basic Med Res,7:S4-7, (2017).
4. IRENA (2022). Renewable Power Generation Costs in 2021. International Renewable Energy Agency, Abu Dhabi. https://www.irena.org/-

/media/Files/IRENA/Agency/Publication/2023/Aug/IRENA_Renewable_power_generation_costs_in_2022.pdf (retirado em 21/11/2025)
5. E. S. Izidoro, H. N. Silva, and R. Soares, "Painel Solar: uma alternativa para a geração de energia". Bolsista de Valor, *1*, 81–84, (2010). https://editoraessentia.iff.edu.br/index.php/BolsistaDeValor/article/view/1796.
6. C. A. Pereira, "Sistemas de autoconsumo fotovoltaico*",* (2016). https://comum.rcaap.pt/handle/10400.26/12577
7. P. M. Batista, "Autoconsumo fotovoltaico, análise de um caso de estudo em termos de poupança e de rentabilidade"*,* (2017). https://repositorio.ul.pt/handle/10451/28282
8. Web Archive, "O que é um sistema solar fotovoltaico" (2021). https://web.archive.org/web/20230921190805/https://energiasmadeira.pt/como-funciona/#o_sistema.
9. J. Magalhães, "Arduino – Ciência Viva", ISEP (2019).
10. D. F. M. Almeida, "Sistema Inteligente para Iluminação Pública"*,* (2016). https://repositorio-aberto.up.pt/handle/10216/85891
11. J. Magalhães, A. Pinto, M. Costa, and C. Sá, "Implementation of a PBL/CDIO methodology at ISEP-P. PORTO Systems Engineering Course", 3rd International Conference of the Portuguese Society for Engineering Education (CISPEE 2018), 27-29 June, Aveiro, Portugal, 2018. ISBN: 978-1-5386-3771-5 (2018).
12. K. Edström, and A. Kolmos, "PBL and CDIO: complementary models for engineering education development", European Journal of Engineering Education, 39:5, pp. 539-555 (2014).
13. J. Malmqvist, "A COMPARISON OF THE CDIO AND EUR-ACE QUALITY ASSURANCE SYSTEMS", in Proceedings of the 5th International CDIO Conference, Singapore Polytechnic, Singapore, June 7 – 10 (2009).

Estudo e Simulação de sistema fotovoltaico para produção energética a instalar Edifício E do ISEP

António Mota[1], José Magalhães[2], Edgar Monteiro[1], Cláudio Cunha[1], Bruno Nunes[1]
Alberto Peixoto Pinto[2]

[1] Alunos do Instituto Superior de Engenharia, Politécnico do Porto, Porto, Portugal
1030050@isep.ipp.pt, 1221761@isep.ipp.pt, 1182105@isep.ipp.pt, 1201961@isep.ipp.pt
[2] Professores do Instituto Superior de Engenharia, Politécnico do Porto, Porto, Portugal
jdm@isep.ipp.pt, apo@isep.ipp.pt

Resumo. Este estudo apresenta uma proposta técnica para a implementação de um sistema de geração de energia fotovoltaica para autoconsumo no Edifício E do Instituto Superior de Engenharia do Porto (ISEP). O objetivo principal consistiu em analisar a viabilidade de um sistema capaz de maximizar a eficiência energética do consumo elétrico do edifício através de geração de energia a partir de uma plataforma de painéis fotovoltaicos rotativa com seguimento solar (solar tracking) automatizada. A metodologia adotada integrou a modelação matemática da trajetória solar e o desenvolvimento de um protótipo experimental automatizado, baseado na plataforma Arduino e com sensores de luminosidade. A investigação abordou a análise comparativa de tecnologias de painéis monocristalinos e policristalinos, bem como a simulação do controlo de movimento dos painéis fotovoltaicos em dois eixos. Os resultados obtidos através da maquete funcional demonstram a viabilidade técnica da aplicação de sistemas dinâmicos de captação solar, evidenciando o potencial de otimização da produção de energia elétrica face a soluções estáticas.

Palavras-chave: Energia Fotovoltaica; Seguidor Solar (Solar Tracker); Arduíno; Prototipagem Eletrónica; Simulação.

1 ENQUADRAMENTO e OBJETIVOS

A implementação de sistemas fotovoltaicos para iluminação em edifícios públicos e escolas representa uma solução sustentável e economicamente viável. Nesse sentido e no âmbito da unidade curricular de Laboratórios de Matemática 1 (Curso de Licenciatura em Sistemas Elétricos de Energia), foi lançado o repto a este grupo de estudantes para a realização de um estudo de melhoria da eficiência com gastos de consumo energético de um determinado edifício E do campus do ISEP. Neste edifício estão localizados os órgãos de administração do ISEP, bem como um conjunto variado de serviços administrativos.

Os objetivos com a realização deste trabalho foram o de apresentar uma proposta de implementação de sistema real de autoprodução de energia fotovoltaico para o edifício E do ISEP e com medidas de melhoria eficiência energética de consumos (tarefa 1), modulação 3D em software do edifício E com inclusão dos respetivos painéis fotovoltaicos (tarefa 2), construção de um protótipo físico automatizado de pequena escala representativo das soluções propostas (tarefa 5). As tarefas 3 e 4 tinham objetivos de índole mais matemático, tais como, modelar e simular em GeoGebra a rotação dos painéis fotovoltaicos em consonância com o movimento solar e aplicação de métodos numéricos de resolução de equações para a determinação da posição do sol em determinadas situações pré-enunciadas matematicamente.
Dos resultados obtidos e sua análise sugere-se a instalação de um sistema rotativo cos 62 painéis fotovoltaicos monocristalinos ocupando uma área de 124 m^2 no telhado do edifício E e com capacidade de fornecer cerca de 40 kWh para as suas necessidades energéticas. A maquete produzida e com simulação do sistema eletrónico automatizado implementada na plataforma Arduíno demonstram a viabilidade técnica do sistema proposto.

2 Referencial Teórico

Os painéis fotovoltaicos são dispositivos que permitem captar energia solar convertendo essa mesma energia em energia elétrica [1]. Sendo considerada uma alternativa energética renovável e alternativa a outros tipos de energia, como os combustíveis fósseis, é uma das mais utilizadas em todo o mundo. A integração de sistemas fotovoltaicos em edifícios escolares tem sido amplamente estudada como estratégia para reduzir custos energéticos e promover sustentabilidade. Estudos demonstram que escolas apresentam condições favoráveis para autoprodução devido ao elevado potencial solar e ao consumo concentrado em horários diurnos. Por exemplo, Alsamamra e Shoqeir [1] analisaram a viabilidade de instalar sistemas fotovoltaicos em escolas públicas, concluindo que a utilização de telhados pode gerar economias significativas e reduzir emissões de CO_2. Atualmente, os painéis fotovoltaicos disponíveis no mercado permitem obter produções de energia de cerca de 600W para uma área útil de captação de 2 m^2.
A utilização de painéis fotovoltaicos para produção de energia elétrica, pode ser equacionada para funcionamento em sistema On-Grid ou Off-Grid [2]. Um sistema On-Grid, permite utilizar energias renováveis em conjunto com a energia da rede de distribuição, de forma que esta última complemente a primeira apenas quando a energia produzida é insuficiente. Já um sistema Off-Grid, apenas permite utilizar energias fotovoltaica sem acesso à rede de distribuição e utiliza baterias para armazenar a energia gerada e garantir o fornecimento mesmo sem sol. Trata-se de um sistema totalmente independente. Os sistemas Híbridos combinam as duas tecnologias, são conectados à rede e possuem baterias, permitindo o uso do excedente armazenado e atuando como backup em caso de falhas na rede. Abdillah et al. [3] compararam diferentes configurações (on-grid, off-grid e híbridas) para escolas, verificando que sistemas híbridos oferecem maior poupança anual e melhor integração com a rede elétrica.

Os painéis solares fotovoltaicos são utilizados para o autoconsumo elétrico. Estes tipos de painéis são compostos por células de silício que reagem à luz solar produzindo eletricidade. Dos diversos tipos de painéis fotovoltaicos (orgânicas, monocristalino, policristalino, filme fino, silício amorfo, etc.) os mais comuns são os monocristalinos e policristalinos (Fig.1). Um painel monocristalino é composto por um conjunto de células de um cristal de silício. Nestes pretende-se que o espaço entre as várias células seja o mínimo possível para maximizar a captação de energia. Estes painéis possuem uma cor mais escura, que os distingue dos painéis policristalinos, mais azulados e o seu rendimento pode variar entre 15% e 20%, dependendo da qualidade [2].

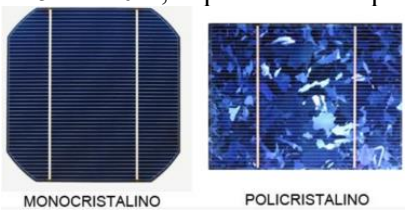


Fig. 1. Aspeto de células monocristalinas e policristalinas.

Os painéis policristalinos possuem conjunto de células de cristais de silício que foram cortados e aquecidos, sendo fragmentado no seu estado final. A produção deste tipo de componentes é mais rápida e barata, mas têm uma eficiência menor que as células feitas com um cristal único. Estes painéis possuem um aspeto mais característico de formato quadrado e cor azulada e o seu rendimento pode variar entre os 13% e os 16% [2]. Podem constituir uma boa alternativa para situações em que a disponibilidade económica é menor ou em que a luz solar seja mais abundante durante o ano, por forma a tornar a diferença de rendimento mais negligenciável.

Além da geração fotovoltaica, a eficiência energética em escolas pode ser ampliada através do controlo inteligente da iluminação. Sistemas baseados em sensores de movimento e luminosidade, integrados com plataformas como Arduino, permitem reduzir desperdícios energéticos em salas e corredores. Dhakla et al. [4] desenvolveram um controlador automático de iluminação com Arduino e sensores PIR, demonstrando redução significativa no consumo elétrico.

Em termos educativos, a construção de protótipos com Arduino tem sido utilizada para ensinar conceitos de energia renovável e automação. Hassan e Abubakar [5] apresentaram um sistema de rastreamento solar com Arduino e LDRs, obtendo ganhos de até 30% na captação energética em comparação com painéis fixos. Estes protótipos, aliados a simulações em plataformas abertas, permitem comparar soluções como rastreamento solar versus sistemas estáticos, ou iluminação convencional versus automatizada, promovendo aprendizagem prática e análise crítica de viabilidade técnica e económica.

3 METODOLOGIA e MATERIAIS

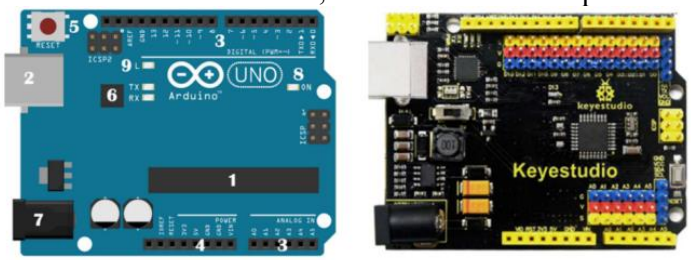
3.1 Metodologia

No sentido de haver uma integração de conhecimentos na área de Engenharia Eletrotécnica e de Matemática foi realizado um projeto que envolveu as unidades curriculares de Laboratórios de Matemática I (LMAT1) e Métodos de Engenharia (MTENG) do 1º ano do curso de licenciatura de Engenharia de Sistemas Elétricos de Energia, ano letivo de 2022/23. Os objetivos da atividade de aprendizagem lançado aos alunos foi da concretização de uma proposta/solução de implementação e simulação de um sistema fotovoltaico a instalar num edifício do ISEP, em particular o edifício E. Para a implementação da solução foi necessário usar e programar na plataforma Arduino, simular e programar por blocos e/ou código na plataforma TinkerCad o funcionamento de circuitos eletrónicos. Para além dessas *hard skills* do fórum da engenharia, houve o intuito de aprendizagem de conceitos matemáticos, estudo de funções multivariável (funções implícitas e paramétricas), resolução de equações pelos métodos numéricos. Em relação às aprendizagens *soft skills*, o projeto envolveu trabalho de equipa com realização de relatórios escritos, construção de poster e apresentação pública de resultados.
O projeto iniciou com estudos e pesquisas prévias sobre funcionamento geral: do Arduíno e forma da sua interligação com sensores de luminosidade e controlo de servo motores; TinkerCad e como este permitia o desenvolvimento de modelos em 3D, criar e simular circuitos elétricos e eletrónicos; GeoGebra e como o mesmo permitia explorar graficamente e numericamente de problemas de matemática e simultaneamente simular o movimento de rotação dos painéis fotovoltaicos na direção do sol.
Partindo desses estudos iniciais, elaborou-se uma proposta para dimensionar o sistema de painéis fotovoltaicos rotativos real para gerar energia para o edifício E, e com vista a uma maior eficiência de produção de energia, movimentando-se em direção ao sol ao longo do tempo com o melhor ângulo de inclinação [6]. Para sustentar a proposta, com apoio do TinkerCad, realizou-se um estudo do local da localização dos painéis no telhado, apresentando-se um modelo 3D do edifício, criou-se os esquemas elétricos, a programação e a simulação de funcionamento eletrónico do sistema. Para uma melhor visualização e demonstração de funcionamento da solução construiu-se um protótipo/maquete em pequena escala que mostra a dinâmica de movimento dos painéis com integração de sensores, controlo e monitorização do sistema eletrónico na plataforma Arduíno. Já o GeoGebra serviu com base para reprodução visual e conhecimento do movimento solar com determinação de algumas posições do sol ao longo do dia.

3.2 Materiais

Para a monitorização e controlo do sistema fotovoltaico houve necessidade da procura, no domínio dos microcontroladores, de uma plataforma de implementação simples, rápida, fiável e de baixos custo. A escolha recaiu no Arduíno, porque permite ter acesso a um variado número de possibilidades de soluções na área da eletrónica de automação e controlo, desde o controlo de sistemas de iluminação até ao envio de comandos para motores. A plataforma Arduíno foi criada por Massimo Banzi, David Cuartielles, Tom

Igoe e David Mellis [7] e criaram aquilo a que deram o nome de Arduíno (Fig. 2 A)) e teve como objetivos permitir a toda a gente de forma simples desenhar e criar, com recurso a equipamentos eletrónicos e software, a interação das tecnologias com o mundo físico, desde estudantes a criadores, até desenvolvedores profissionais.



A) Arduíno Uno B) Keyestudio Uno

Fig. 2. Microcontrolador Arduíno

Existem algumas variantes destes microcontroladores e que vão desde o Uno até ao Mega, passando pelo Micro ou o Zero [8]. Apesar de algumas diferenças eles, por exemplo, número de portas de comunicação disponíveis ou capacidade de ligação Bluetooth, a adoção da versão Uno (Fig. 2) para a realização do projeto verificou-se ser suficiente e por ser uma das versões mais baratas e simples. Sendo o Arduíno um sistema baseado em *open source*, com liberdade de criação e desenvolvimento, fez com que fosse possível encontrar no mercado, mas com preços mais competitivo, a placa Keyestudio Uno (Figura 1 B), da Key Estudio, construída com base no mesmo sistema e código fonte [9], e adotada como solução final a integrar no projeto.

Após a escolha da plataforma de controlo em Arduíno, houve a necessidade de escolher todos os componentes (hardware) de suporte à construção da maquete do protótipo da solução de geração fotovoltaica. Como os painéis fotovoltaicos deverão estar sempre (ou o máximo possível) voltados de forma direta para a fonte de energia solar [6], houve a necessidade da utilização de 2 servo motores (Fig. 3A) para movimentar os painéis, um para o eixo vertical e outro o horizontal. Este componente é semelhante a um motor comum, com a particularidade de poder rodar até uma determinada posição, mantendo-a numa espécie de bloqueio, até que seja dada indicação para rodar novamente [10].

A) Servo motor B) LDR C) BH1750 D) I2C1602

Fig. 3. Hardware de suporte ao Arduino

Para apoio à decisão desse movimento de rotação dos painéis foram usadas 4 LDR e que é um componente eletrónico em que o valor a sua resistência varia consoante a luz que é captada (Fig. 2 B). Tipicamente, quanto maior o valor da luz que incide, menor é o valor da resistência e vice-versa [11]. Um outro componente utilizado neste projeto, pese embora não tenha uma implicação direta no resultado, é um medidor de intensidade de luz. Serviu como componente de suporte extra, permitindo ter uma noção sobre

os valores de luminosidade que estavam a ser captados pelo sistema em si. Este componente em particular, é o BH1750, versão Keyestudio (Figura 2 C). Por fim, outro componente utilizado e que permitiu visualizar as leituras em tempo real dos valores que estão a ser manipulados foi o Display de Cristais Líquidos (LCD) I2C1602 (16 caracteres por linha, 2 linhas) (Fig. 2 D).

4 Resultados e Discussão

Atendendo aos objetivos do seminário de Engenharia Informática do ISEP, neste seção dar-se-á enfoque à análise de resultados relativos à proposta a implementar, programação da simulação e da implementação da maquete/protótipo do sistema fotovoltaico.

4.1 Proposta de sistema fotovoltaico para o edifício E

Para a resposta à tarefa 1 do projeto, contactou-se os responsáveis pela manutenção do Edifício E do ISEP (Fig. 3) de forma a obter dados sobre os consumos atuais do mesmo. Tal não foi possível, com a justificação do disposto no Regulamento Geral da Proteção de Dados, como tal, considerando que o presente trabalho é meramente académico, optou-se prosseguir com base em estimativas de consumo energético considerados os sistemas de iluminação e os equipamentos periféricos por constatação no local. Assumiu-se 40 KW/h de consumo do edifício, projetando-se com um coeficiente de simultaneidade de funcionamentos da iluminação e equipamentos de 80%.
Para melhoria da eficiência energética para o edifício E propõem-se dimensionar um sistema gerador de energia elétrica com base em painéis fotovoltaicos rotativos sempre orientados em direção ao sol, assumindo a sua melhor eficiência como relata a literatura. A comparação da eficiência do sistema estático em relação ao não foi possível ser confirmada neste trabalho, por não ter havido acesso a um painel real comercial que se propõe para a solução. Na opção de escolha do tipo de painel fotovoltaico foram apenas considerados dois tipos, monocristalino e policristalino. Após consulta a alguns fabricantes selecionou-se o painel Canadian Solar 650 W HIKU7 monocristalino com capacidade de 650 W e com dimensão de cerca 2 m^2 [12]. Considerando o consumo de 40 kWh para o edifício, serão necessários montar 62 painéis (62 x 0,650 kWh = 40.3 kWh) ocupando aproximadamente 124 m^2 (62 x 2 m^2) de telhado. Para confirmação de área disponível de telhado do edifício E para a instalação dos 62 painéis, com recurso ao Google Maps (Fig. 3), determinou-se haver disponibilidade de cerca de 1.800 m2, mais que suficientes para as necessidades.

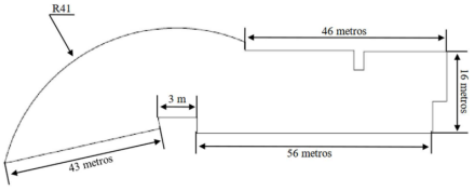
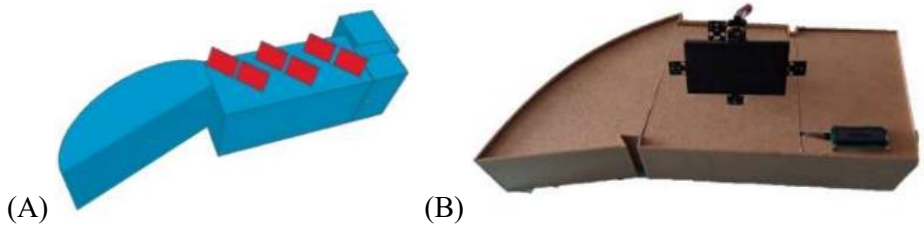


Fig. 4. Edifício E do ISEP

A fase seguinte foi criar uma representação 3D do edifício, com recurso ao TinkerCad (Fig. 5A). Neste modelo 3D foi gerido a localização dos painéis fotovoltaicos e que forneceu a base para a construção da maquete física representativa do Edifício E do ISEP, construída com recurso ao MDF (Medium Density Fiberboard) (Fig. 5B).



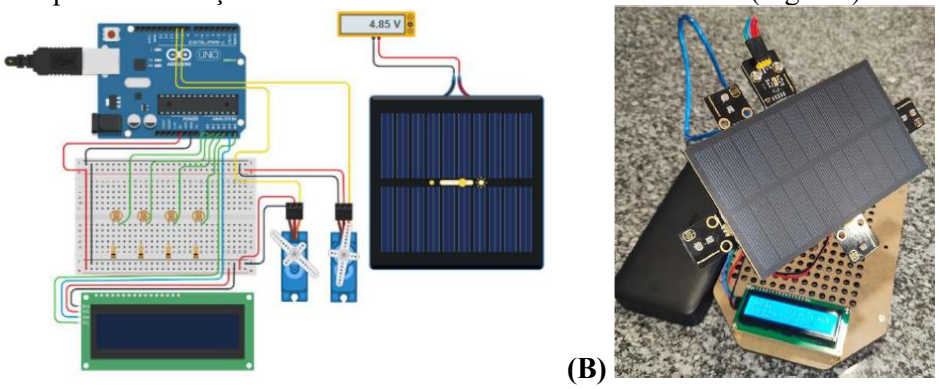
(A) (B)

Fig. 5. Modelo 3D em Tinkercad (A) e Maquete física do Edifício (B)

Na maquete física construída (Fig. 5B) é possível observar a existência de um mini painel fotovoltaico com 4 sensores LDR montados nas suas laterias e está acoplado a todo o circuito composto pelo microcontrolador e restantes componentes eletrónicos.

4.2 PROTÓTIPOS E DESENVOLVIMENTO ARDUÍNO

Para a construção do protótipo final, desenvolveu-se circuito eletrónico, sua programação e respetiva simulação de funcionamento através do TinkerCad (Fig. 6A)



(A) **(B)**

Fig. 6. Protótipo do sistema em Tinkercad

Com base no sucesso dessa simulação foi possível avançar para a construção física do protótipo eletrónico do painel fotovoltaico (Fig. 6B) que rodada na direção do foco de luz (teste com luz solar e uma lanterna) e possibilitou o carregamento de um carregador portátil (*powerbank*). Após a sua implementação carregou-se e reajustou-se o software para a IDE do Arduíno para a realização dos testes à sua programação. Nessa programação foi necessário criar linhas de código (Fig. 7) para definir quais as bibliotecas de apoio aos sensores (servomotores, LCD e do medidor de intensidade de luz), definirem variáveis gerais, os pinos onde se ligam os sensores ao Arduíno, valores mínimos e máximos para a posição de cada servomotor (evitando que o painel rodasse para além do necessário), valores de tolerância para leitura de valores nos LDR de modo a evitar movimento de rotação instável do movimento do painel (vibrações).

```
#include <Servo.h> //biblioteca para controlo dos servos
#include <LiquidCrystal_I2C.h> //biblioteca para controlo do LCD
#include <BH1750.h> //biblioteca para controlo do medidor de intensidade de luz

LiquidCrystal_I2C lcd(0x27, 16, 2);//Definir o tipo de LCD
BH1750 medidorLUX;//Definir o medidor de intensidade de luz

Servo ud_servo; //servo vertical (up down)
Servo lr_servo; //servo horizontal (left right)

byte ud_servopin=10;//pino do servo vertical
byte lr_servopin=9;//pino do servo horizontal

int ud_posicao=135; //posição inicial servo vertical
int lr_posicao=90; //posição inicial servo horizontal

byte ud_posicao_MAX = 180; //valor máximo para a posição do servo vertical
byte ud_posicao_MIN = 90; //valor mínimo para a posição do servo vertical
byte lr_posicao_MAX = 180; //valor máximo para a posição do servo horizontal
byte lr_posicao_MIN = 0; //valor mínimo para a posição do servo horizontal

int LDR_Direito = A0;//pino do LDR direito
int LDR_Esquerdo = A1;//pino do LDR esquerdo
int LDR_Inferior = A2;//pino do LDR inferior
int LDR_Superior = A3;//pino do LDR superior

byte tolerancia = 300;//definição de margem de tolerância para que os servos
                      //não estejam em constante movimento
byte intervalo = 50;//valor da taxa de atualização (em ms)
byte rotacao_minima = 1;//Definição do ângulo de rotação mínimo
```

Fig. 7. Programação intermédia 1

O bloco da função setup (Fig.8) permite a alocação e a definição dos pinos (input ou output) do Arduíno na ligação com os sensores e a sua inicialização.

```
void setup(){
  ud_servo.attach(ud_servopin);//Definição do pin de controlo do servo
  ud_servo.write(ud_posicao);  //Colocar o servo vertical na posição inicial
  lr_servo.attach(lr_servopin);//Definição do pin de controlo do servo
  lr_servo.write(lr_posicao);  //Colocar o servo horizontal na posição inicial

  pinMode(LDR_Direito,INPUT); //Definir este LDR como input
  pinMode(LDR_Esquerdo,INPUT);//Definir este LDR como input
  pinMode(LDR_Inferior,INPUT);//Definir este LDR como input
  pinMode(LDR_Superior,INPUT);//Definir este LDR como input

  lcd.init();//Inicializar o LCD
  lcd.backlight();//Inicializar o backlight do LCD
  EscreveLCD(6, 0, "SolarTrack", 0, "");//Escreve no LCD 1ª linha (V2)

  medidorLUX.begin();//inicializar o medidor de luz

  delay(1000);//Aguardar 1 segundo
}
```

Fig. 8. Programação intermédia 2

O bloco de loop (Fig. 9) reflete a rotina principal e que chama as funções de escrita da intensidade de luz incidente no painel e a tomada decisão de rotação dos painéis.

```
void loop(){
  EscreveLCD(6, 1, "LUX:", MedirLuz(), "");//Escreve no LCD 2ª linha (V2)
  MoverServos();
}
```

Fig. 9. Programação intermédia 3

A função EscreveLCD permite escrever informação no LCD a intensidade da luz incidente e que é obtida através da função MedirLuz que devolve um valor inteiro correspondente à medição de intensidade de luz do respetivo sensor.

A função MoverServos (Fig. 10) é responsável pela rotação dos servomotores de acordo com a leitura de valores dos LDR.

```
//Esta função move os servos. Esta é a V2 da função (após otimização do código)
void MoverServos(){
  int L = analogRead(LDR_Esquerdo);//leitura do valor do sensor
  int R = analogRead(LDR_Direito);//leitura do valor do sensor
  int U = analogRead(LDR_Superior);//leitura do valor do sensor
  int D = analogRead(LDR_Inferior);//leitura do valor do sensor

  lr_posicao = CalcularMovimento(L, R, lr_posicao, lr_posicao_MIN, lr_posicao_MAX);//Calcular a nova posição do servo horizontal (V2)
  lr_servo.write(lr_posicao);//muda a posição do servo (caso o valor seja o mesmo, o servo não se mexe)
  EscreveLCD(0, 0,"H:", lr_posicao, "");//Escreve no LCD (V2)
  //lcd.setCursor(0,0);lcd.print("HORZ: ");lcd.print(lr_posicao);if(lr_posicao<100)lcd.print(" ");//Escreve no LCD V1
  delay(intervalo);//intervalo entre ajustes para evitar danos aos servos

  ud_posicao = CalcularMovimento(U, D, ud_posicao, ud_posicao_MIN, ud_posicao_MAX);//Calcular a nova posição do servo vertical (V2)
  ud_servo.write(ud_posicao);//muda a posição do servo (caso o valor seja o mesmo, o servo não se mexe)
  EscreveLCD(0, 1,"V:", ud_posicao, "");//Escreve no LCD (V2)
  //lcd.setCursor(0,1);lcd.print("VERT: ");lcd.print(ud_posicao);if(ud_posicao<100)lcd.print(" ");//Escreve no LCD V1
  delay(intervalo);//intervalo entre ajustes para evitar danos aos servos
}
```

Fig. 10. Programação intermédia (função MoverServos)

A função MoverServos chama uma outra, CalcularMovimento (Fig. 11), e que permite realizar o posicionamento dos servomotores em cada iteração.

```
//Esta função calcula a nova posição de um servo. Recebe o valor de duas LDR (L e D ou U e D), a posição
//atual do servo e os mínimos e máximos do mesmo. Devolve o valor da nova posição. Esta função fez parte
//do processo de otimização do código
int CalcularMovimento(int valor1, int valor2, int posicao, int min, int max){
  //Otimizacao de código V3 - substitui todas as linhas acima desta na função CalcularMovimento
  if (abs(valor1 - valor2) > tolerancia){//Apenas muda de posição se ultrapassar a tolerância (diferença entre valores)
    posicao = valor1 > valor2 ? posicao - rotacao_minima : (valor1 < valor2 ? posicao + rotacao_minima : posicao);
    //Valida qual o lado para onde rodar
  }
  return posicao < min ? min : (posicao > max ? max : posicao);//Valida os limites dos servos e devolve a nova posição
}
```

Fig. 11. - Programação intermédia (CalcularMovimento V3)

5 Conclusões

Este trabalho reflete uma proposta académica de um sistema de autoprodução fotovoltaica para o edifício E do ISEP usando a plataforma Arduíno para monitorizar e controlar os diversos sensores eletrónicos. O edifício tem uma utilização intensiva, particularmente durante o dia, e sugere-se implementar e incluir no telhado um sistema de 62 painéis fotovoltaicos com rotação automatizada em direção ao movimento do sol e capaz de gerar 40 kWh de energia, permitindo uma redução efetiva da despesa mensal com custos energéticos aliado uma maior sustentabilidade ambiental.

O protótipo físico representativo da solução para implementação do sistema fotovoltaico rotativo no edifício E, demonstram a viabilidade técnica do uso da plataforma eletróncia Arduíno associado a um conjunto de sensores de luz para fazer o controlo dinâmico de processo de autoprodução energética. No entanto, em termos futuros, sugere-se que uma melhoria desta solução no sentido de incluir funcionalidades IoT, nomeadamente, controle e monitorização por telemóvel e gestão integrada computacional da produção e consumo energético de todos os edifícios do campus do ISEP. Atendendo à dimensão, poderá haver necessidade de acrescentar novas funcionalidades, migrar o sistema de microcontroladores para ESP32, Raspberry ou outra com maior capacidade de processamento digital e com integração de interfaces de Bluetooth e WiFi.

Como este trabalho se integrou numa atividade de aprendizagem com alunos de engenharia de Sistemas Elétricos de Energia, os estudantes reconhecem que esta metodologia de aprendizagem lhes permitiu aprofundar conhecimentos relativamente aos sistemas de produção de energia fotovoltaica, solidificar o que aprenderam durante as aulas, utilização de um microcontrolador, cálculo e aplicação de modelos matemáticos. De uma forma geral, os estudantes consideraram o trabalho proposto muito interessante, do ponto de vista académico, mas ao mesmo tempo com uma aplicabilidade prática muito elevada e, tecnicamente, perfeitamente possível de implementar.

Referências

1. H. Alsamamra and J. Shoqeir, "Solar Photovoltaic Systems on Public Schools Buildings: A Case Study in Palestine," American Journal of Electrical Power and Energy Systems, vol. 10, no. 1, pp. 1–5, Feb. 2021.
2. Carolina Hisatomi, "Painel solar monocristalino: o que é e propriedades" (2022). [Online]. Available: https://www.ecycle.com.br/painel-solar-monocristalino/. [Acedido em 13 12 2022].
3. M. Abdillah, F. M. Satria, N. I. Pertiwi, and H. Setiadi, "Design of photovoltaic system for public school building," Indonesian Journal of Electrical Engineering and Computer Science, vol. 29, no. 1, pp. 56–65, Jan. 2023. http://doi.org/10.11591/ijeecs.v29.i1.pp56-65.
4. D. Dhakla et al., "Designing of Automatic Room Lights Controller by Using Arduino," International Journal for Scientific Research & Development, vol. 9, no. 6, 2021.
5. [R. Hassan and B. Abubakar, "Intelligent Arduino Based Automatic Solar Tracking System Using Light Dependent Resistors (LDRs) and Servo Motor," Optics, vol. 9, no. 2, pp. 13–18, Dec. 2020.
6. P. Reis, "Qual a inclinação que um painel solar deve ter?," Portal Energia (2022).[Online]. Available: https://www.portal-energia.com/qual-inclinacao-painel-solar/. [Acedido em 20 12 2022].
7. Arduino, "About Arduino" (2024). [Online]. Available: https://www.arduino.cc/en/about#founders. [Acedido em 12 12 2024].
8. Arduino, "Hardware, boards and components," [Online]. Available: https://www.arduino.cc/en/hardware#boards. [Acedido em 13 12 2022].
9. KeyEstudio, "Distribuidora de sistemas de microcontrolo" (2022). [Online]. Available: https://www.keyestudio.com/. [Acedido em 13 12 2022].
10. H. Mattede, "O que é Servo motor e como funciona?" (2022). [Online]. Available: https://www.mundodaeletrica.com.br/o-que-e-servo-motor-e-como-funciona/. [Acedido em 20 12 2022].
11. P. Alves, "LDR – O que é e como funciona!," (2022) [Online]. Available: https://www.manualdaeletronica.com.br/ldr-o-que-e-como-funciona/. [Acedido em 20 12 2022].
12. S. Shop, "PAINEL SOLAR CANADIAN SOLAR 650W HIKU7 MONO" (2022). [Online]. Available: https://www.solarshop.pt/painel-solar-canadian-solar-650w-hiku7-mono. [Acedido em 20 12 2022].

Session 4B

"Soluções Inovadoras em Desenvolvimento de Software e Aplicações Multiplataforma"

Chair: Rosa Reis **Room:** H207 **Time:** 14:15 - 15:15

Paper	Presenter
Cliente Mobile e Desktop de Comunicações Unificadas para IPBrick OS	Diogo Paiva
Kotlin and Compose Multi-Platform App	Gonçalo Medeiros
Development of a mass drug administration monitoring system for a Humanitarian NGO	Rafael Ribeiro
Property-based testing	António Sousa

Cliente Mobile e Desktop de Comunicações Unificadas para IPBrick OS

Diogo Paiva[1], Miguel Ramalhão[3], Carlos José Campos[1,4], Veríssimo Lima[1], Pedro Reis[1], Vítor Cardoso[1], António Sousa[1,2], Sandra Aires[1], Carla Pinto[1] e Fernando Carvalho[1]

[1] Instituto Superior de Engenharia do Porto (ISEP) – P.Porto, Porto, Portugal
[2] LEMA, ISEP-IPP, Porto, Portugal
[3] IPBRICK - Expandindustria - Estudos, Projectos e Gestão de Empresas, S.A
[4] Instituto de Engenharia de Sistemas e Computadores em Tecnologia e Ciência (INESC TEC), Porto, Portugal
{1221340, crc, vms, mps, vcc, ats, sma, cap, fjc}@isep.ipp.pt

Resumo. O IPBRICK OS é um sistema operativo baseado em Linux Debian, concebido para suportar as soluções empresariais da IPBRICK no domínio das Comunicações Unificadas sobre IP (UCoIP) e Ferramentas Colaborativas para Gestão de Documentos e Processos. Entre os serviços UCoIP integrados pelo módulo IPBRICK.PBX (baseado em Asterisk), destaca-se o *Voice Over Internet Protocol* (VoIP), por norma, suportado por um conjunto distribuído de servidores que executam, entre outros, os protocolos *Session Initiation Protocol* (SIP), responsável pela sinalização e gestão de chamadas, e o *Real-Time Transport Protocol* (RTP), que assegura a transmissão de áudio em tempo real.
A versão 8.0 exigiu a implementação e integração de uma aplicação cliente Mobile e Desktop de Comunicações Unificadas (*softphone* VoIP) que privilegia a simplicidade de configuração, a eficiência de utilização e a estética do IPBRICK OS. Nesse sentido, este trabalho apresenta uma aplicação *softphone* VoIP responsiva, implementada em Dart através da framework Flutter. É baseada numa arquitetura cliente-servidor, em que a comunicação é realizada via HTTP, recorrendo a APIs PHP que estabelecem a interface com o servidor SIP. Deste modo, descrevem-se os principais aspetos técnicos e arquiteturais da solução proposta, incluindo a modelação dos fluxos de comunicação, a estrutura das APIs e os layouts funcionais da interface gráfica. Apresentam-se ainda diagramas de fluxo e os protótipos de ecrã das principais componentes da aplicação – autenticação *Lightweight Directory Access Protocol* (LDAP), interface principal, teclado de marcação, lista de contactos, registo de histórico e de chamadas.

Palavras-chave: Flutter, IPBRICK OS, PHP, SIP, *Softphone*, RTP, VoIP.

1 Introdução

A evolução tecnológica tem promovido o uso intensivo de telemóveis, estimando-se que, em abril de 2025, cerca de 5.81 mil milhões de pessoas, correspondendo a 70.7%

da população mundial, utilizem telemóveis [1]. No centro desta tendência encontram-se as aplicações móveis, que simplificam tarefas do quotidiano [2]. As aplicações são geralmente classificadas como móveis, desktop e Web [3]. No domínio das aplicações móveis, a App Store disponibiliza 1.8 milhões de aplicações [4], enquanto a Google Play Store oferece cerca de 1.55 milhões [5]. Neste vasto universo de aplicações, destacam-se pela sua popularidade, as dedicadas à comunicação baseada em texto, áudio e vídeo – facto que motivou o desenvolvimento de uma aplicação multiplataforma de comunicações com capacidade de realizar chamadas de voz através da Internet – um dos serviços de *softphone VoIP* (doravante designado por *softphone*). Tratando-se de uma aplicação multiplataforma, tem a vantagem de funcionar em vários dispositivos – *write once, run everywhere* – permite diminuir custos e tempo de desenvolvimento. No entanto, as aplicações multiplataforma comparadas com as aplicações nativas, apresentam desempenho inferior e limitações impostas por atualizações realizadas nos sistemas operativos que podem comprometer total ou parcialmente as suas funcionalidades [6-8].

Tecnicamente, os *softphones* são aplicações que recorrem a tecnologias de Voz sobre IP (VoIP), oferecendo protocolos como o *Session Initiation Protocol* (SIP), o *Real-time Transport Protocol* (RTP) e o *Extensible Messaging and Presence Protocol* (XMPP) para mensagens instantâneas. Os mais modernos já integram *Web Real-Time Communication* (WebRTC), uma tecnologia que combina múltiplos protocolos, entre outros, o RTP, o *Secure* RTP (SRTP) e o *Datagram Transport Layer Security* (DTLS); permitindo transmitir áudio, vídeo e dados entre navegadores e aplicações em tempo real [9]. Apesar da sua flexibilidade e da compatibilidade com diferentes sistemas operativos, a maioria dos *softphones* exige configurações manuais complexas, tais como: a introdução de endereços de servidores SIP, credenciais do utilizador e definições específicas dos serviços. Muitas destas configurações envolvem conhecimento que os utilizadores nem sempre possuem, o que dificulta a sua utilização, especialmente em ambientes empresariais. Por esse facto, a solução proposta foi desenvolvida com o objetivo de simplificar a experiência de utilização dos clientes dos serviços IPBRICK, através de uma aplicação funcional e prática, que requer o mínimo de configuração possível. Para o efeito, uma análise ao estado da arte relativamente à evolução das comunicações VoIP, mostra que esta tecnologia tem impulsionado o desenvolvimento de diversos *softphones* que oferecem funcionalidades avançadas, compatibilidade multiplataforma e suporte a codecs de alta qualidade. Entre os mais utilizados, destacam-se: o Linphone, o MicroSIP, o Zoiper, o Jitsi, o Bria e o PhonerLite, com características específicas à medida da experiência do utilizador [10]. O Linphone é open-source e permite elevada flexibilidade e integração com serviços VoIP, mas a sua interface e a configuração inicial são complexas para utilizadores inexperientes. O MicroSIP é muito eficiente para o Windows, mas apresenta limitações de compatibilidade noutros ambientes. O Zoiper permite configurações automáticas nas versões empresariais, mas já na sua versão gratuita exige conhecimentos de configurações SIP. O Jitsi disponibiliza suporte nativo a videoconferência e é muito usado em ambientes colaborativos. O Bria é uma solução comercial para integração

empresarial e oferece suporte técnico dedicado [11]. Finalmente, o PhonerLite é uma aplicação bastante simples para uso pessoal em Windows, mas a interface que disponibiliza é considerada pouco apelativa. [10]

Em [12] e [13], comparam arquiteturas baseadas em WebRTC e SIP, concluindo que o WebRTC apresenta melhor desempenho em métricas como latência, PSNR e estabilidade de rede.

2 Tecnologias e Protocolos

A framework Flutter utiliza linguagem Dart, semelhante a Java e JavaScript, permitindo o desenvolvimento de interfaces nativas multiplataforma baseadas em *widgets*. A integração da biblioteca *dart_sip_ua* na framework possibilita a comunicação via protocolo SIP, utilizado em sistemas VoIP para o estabelecimento, controlo e terminação de sessões de comunicação sobre WebSocket. Embora a sinalização da chamada seja gerida pelo *dart_sip_ua*, a transmissão de áudio e vídeo é realizada através do protocolo RTP que, integrado no protocolo WebRTC, garante que a mesma transmissão se realize em tempo real [9].

O processo de autenticação de utilizadores recorre a *Lightweight Directory Access Protocol* (LDAP), tecnologia que fornece acesso a informações de diretório, ou seja, armazena dados estruturados, entre outros, sobre utilizadores e dispositivos [14].

3 Implementação

De acordo com a arquitetura geral do projeto apresentada na Fig. 1, a solução desenvolvida considera três entidades distintas: a aplicação, os serviços de comunicação VoIP da IPBRICK e as APIs que permitem estabelecer a comunicação entre ambas. A comunicação é realizada através de pedidos HTTP POST, com o objetivo de definir os diferentes campos de configuração necessários para que as bibliotecas se liguem aos respetivos servidores. Quando as bibliotecas não são necessárias, as APIs estabelecem a ligação direta entre a aplicação e os servidores.

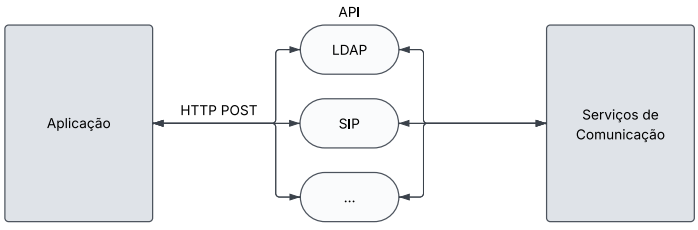


Fig. 1. Arquitetura geral do projeto.

A aplicação integra funcionalidades de autenticação e de gestão de chamadas, cujo respetivo fluxo de utilização se encontra representado na arquitetura geral da aplicação

da Fig. 2. Começa por apresentar ao utilizador um ecrã inicial, onde devem ser inseridas as credenciais de autenticação. Caso a autenticação seja válida, ocorre o registo no servidor SIP; caso contrário, o utilizador é redirecionado para o ecrã de autenticação. Após a autenticação e o respetivo registo no servidor SIP, o utilizador tem acesso ao ecrã principal, podendo navegar entre os ecrãs disponíveis e dedicados ao teclado, aos contactos e ao histórico, a partir dos quais é possível efetuar ou receber chamadas e regressar ao ecrã principal após a chamada terminar.

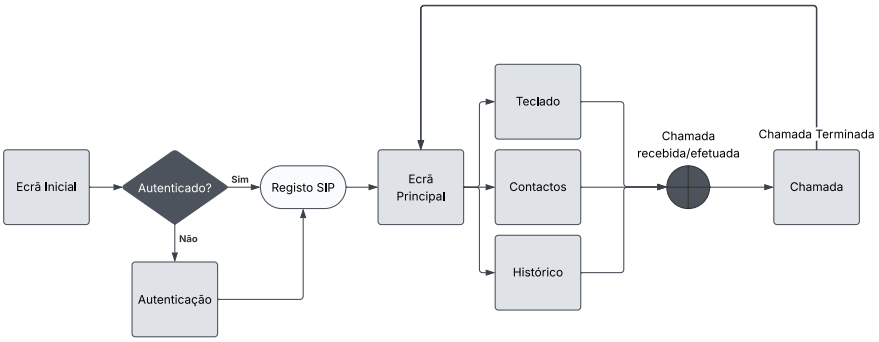


Fig. 2. Arquitetura geral da aplicação - *softphone*.

Considerando os numerosos detalhes que envolvem a lógica de programação dos vários ecrãs que constituem a arquitetura geral da aplicação, apresentam-se apenas os principais e de fluxo mais complexo, utilizando fluxogramas, assumindo-se que os restantes possam ser facilmente concebidos a partir destes exemplos.

3.1 Ecrã de Autenticação

O ecrã de autenticação assegura que só utilizadores autorizados acedem às funcionalidades da aplicação, apresentando um formulário simples, que solicita o preenchimento do email e da palavra-passe submetidos pela ação de um botão, Fig. 3.

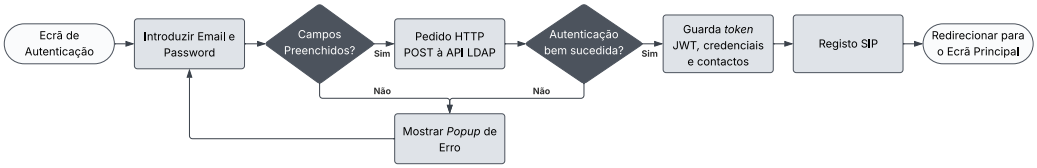


Fig. 3. Fluxograma do ecrã de autenticação.

Após a submissão do formulário, verifica-se se os campos foram preenchidos. Havendo campos vazios, é apresentado um *popup* de erro solicitando o preenchimento. Já se foram preenchidos, é efetuado um pedido HTTP POST à API de autenticação, que envia as credenciais em formato JSON, conforme é ilustrado no fluxograma da Fig.4.

A API procede à receção e extração dos dados do corpo da mensagem para autenticação no servidor LDAP. Se a autenticação não for bem-sucedida, é enviada uma mensagem de erro para a aplicação a informar o utilizador. Em caso de sucesso, é gerado um token JWT, contendo informações relevantes do utilizador e o tempo de expiração do token (atualmente definido para uma hora), que será utilizado nas iterações subsequentes com a API. De seguida, é efetuada uma consulta aos restantes utilizadores existentes no servidor LDAP, sendo estes filtrados e ordenados alfabeticamente para integração na lista de contactos da aplicação. Por fim, é enviada uma mensagem de sucesso, incluindo o token e a lista de contactos, os quais são armazenados localmente no dispositivo do utilizador através das bibliotecas *shared_preferences* e *flutter_secure_storage*, Fig. 4.

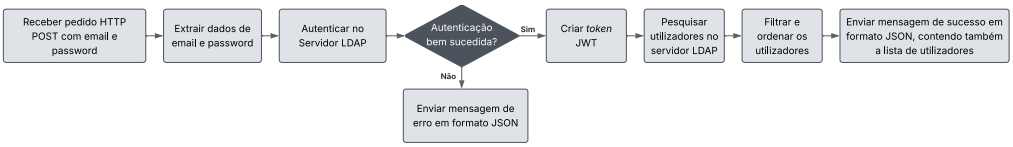


Fig. 4. Fluxograma relativo à API de autenticação dos utilizadores.

Após o processo de autenticação, a aplicação regista o utilizador no servidor SIP, redireciona-o para o ecrã principal e habilita-o à realização e receção de chamadas.

```
<? php
      header ('Content - Type : application / json');
      header ('Access - Control - Allow - Origin : * ');
      header ('Access - Control - Allow - Methods : POST');
      header ('Access - Control - Allow - Headers : Content - Type');
      $data = json_decode (file_get_contents ("php :// input"),true );
      $email = $data ['email'];
      $pass = $data ['password'];
      $displayName = $data ['displayName'];
      list ( $uid , $domain ) = explode ( '@' , $email );
      list ( $emailp , $orgp ) = explode ( '.' , $domain );
      $uri = " ";
      $webSocketUrl = " ";
      $authorizationUser = $uid;
      $password = $pass ;
      $displayName = $displayName;
      $userAgent = "CAFE Phone";
      $port = "5060";
      $register_expires = 3600;
      $sip_config = [
      " uri " => $uri ,
      " webSocketUrl " => $webSocketUrl,
      " authorizationUser " => $authorizationUser,
      " password " => $password,
      " displayName " => $displayName,
      " userAgent " => $userAgent,
      " register_expires " => $register_expires]
      echo json_encode ( $sip_config ) ;
 ? >
```

List. 1. API de configuração SIP.

Para que tal suceda, os dados de autenticação armazenados na aplicação – como o nome, o email e a palavra-passe – são enviados através de um pedido HTTP POST para uma API (List. 1), responsável por processar essa informação e gerar a configuração completa dos parâmetros necessários à ligação com o servidor SIP. Entretanto, esses parâmetros são devolvidos à aplicação em formato JSON, permitindo-lhe prosseguir com o registo (REGISTER SIP) do utilizador no servidor SIP através da biblioteca *dart_sip_ua*. Conforme o desejado, este procedimento garante características de configuração mínima e dinâmica da aplicação, uma vez que as definições específicas do servidor SIP são geridas no servidor HTTP, evitando a necessidade de modificar a aplicação sempre que ocorrem alterações ao nível do processo de comunicação e alojar de forma permanente os dados sensíveis sobre o utilizador na aplicação. Adicionalmente, no registo do servidor SIP são configurados notificadores de eventos, para o estado de registo e para o estado das chamadas, que permitem que os restantes ecrãs recebam, em tempo real, o estado da ligação e das chamadas, bem como alterações registadas na aplicação.

3.2 Ecrã Inicial (*Splash Screen*)

O ecrã inicial exibe-se momentaneamente no ecrã e atua como um *splash screen*, sendo exibido durante um curto período enquanto são verificadas as credenciais e carregados os recursos essenciais, Fig. 5.

Fig. 5. Fluxograma relativo ao ecrã inicial.

3.3 Ecrã Principal

O ecrã principal permite ao utilizador aceder às suas funcionalidades essenciais, como o teclado, contactos e histórico de chamadas, Fig. 6.

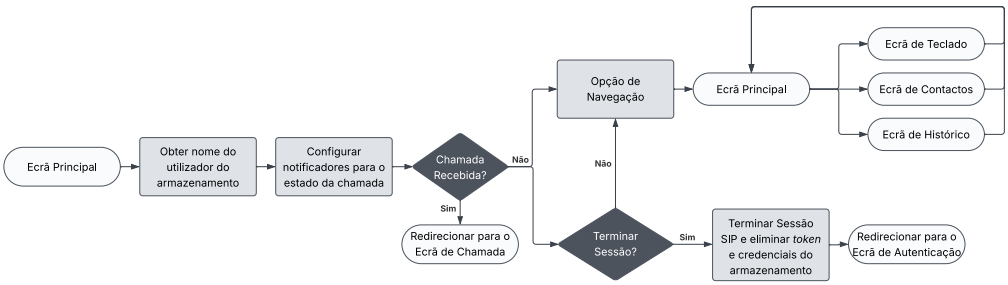


Fig. 6. Fluxograma do ecrã principal.

3.4 Ecrã do Teclado

O ecrã de teclado permite ao utilizador a introdução de números para efetuar chamadas ou pesquisar contactos, Fig. 7.

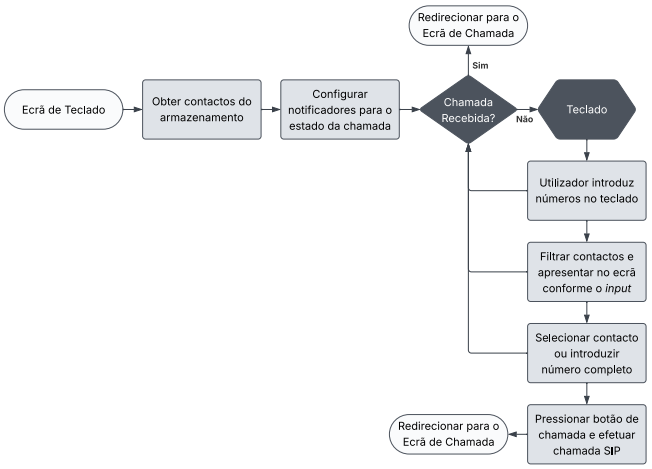


Fig. 7. Fluxograma do ecrã do teclado.

3.5 Ecrã da Chamada

O ecrã da chamada é responsável por gerir toda a experiência de comunicação em tempo real, tanto para chamadas efetuadas como recebidas. O seu funcionamento encontra-se representado no fluxograma da Fig. 8.

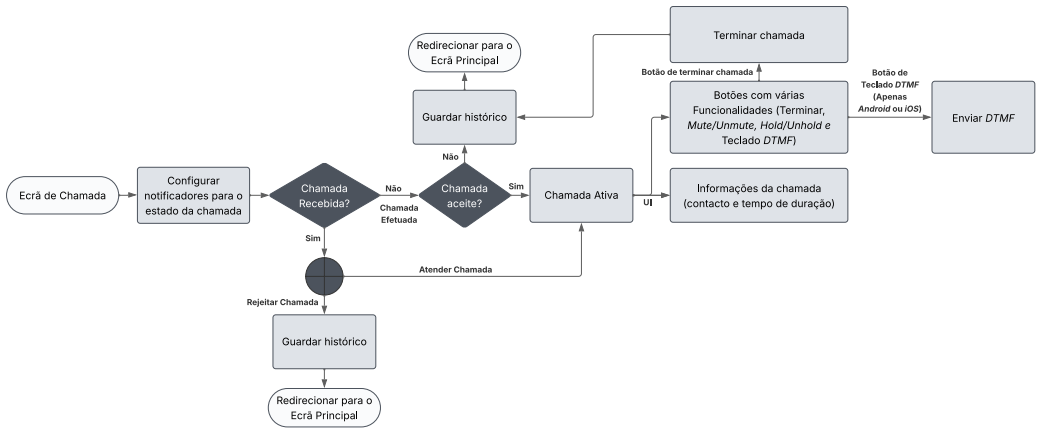


Fig. 8. Fluxograma do ecrã da chamada.

Neste ecrã, a aplicação configura os notificadores do estado de chamada, que verificam se a chamada foi recebida ou efetuada. Nesta fase, são apresentadas informações sobre a chamada, como: o nome, o email e a fotografia do contacto.

4 Resultados

A Fig. 9 apresenta os *layouts* da maioria dos ecrãs da aplicação, alguns dos quais implementados de acordo com as descrições realizadas na seção anterior.

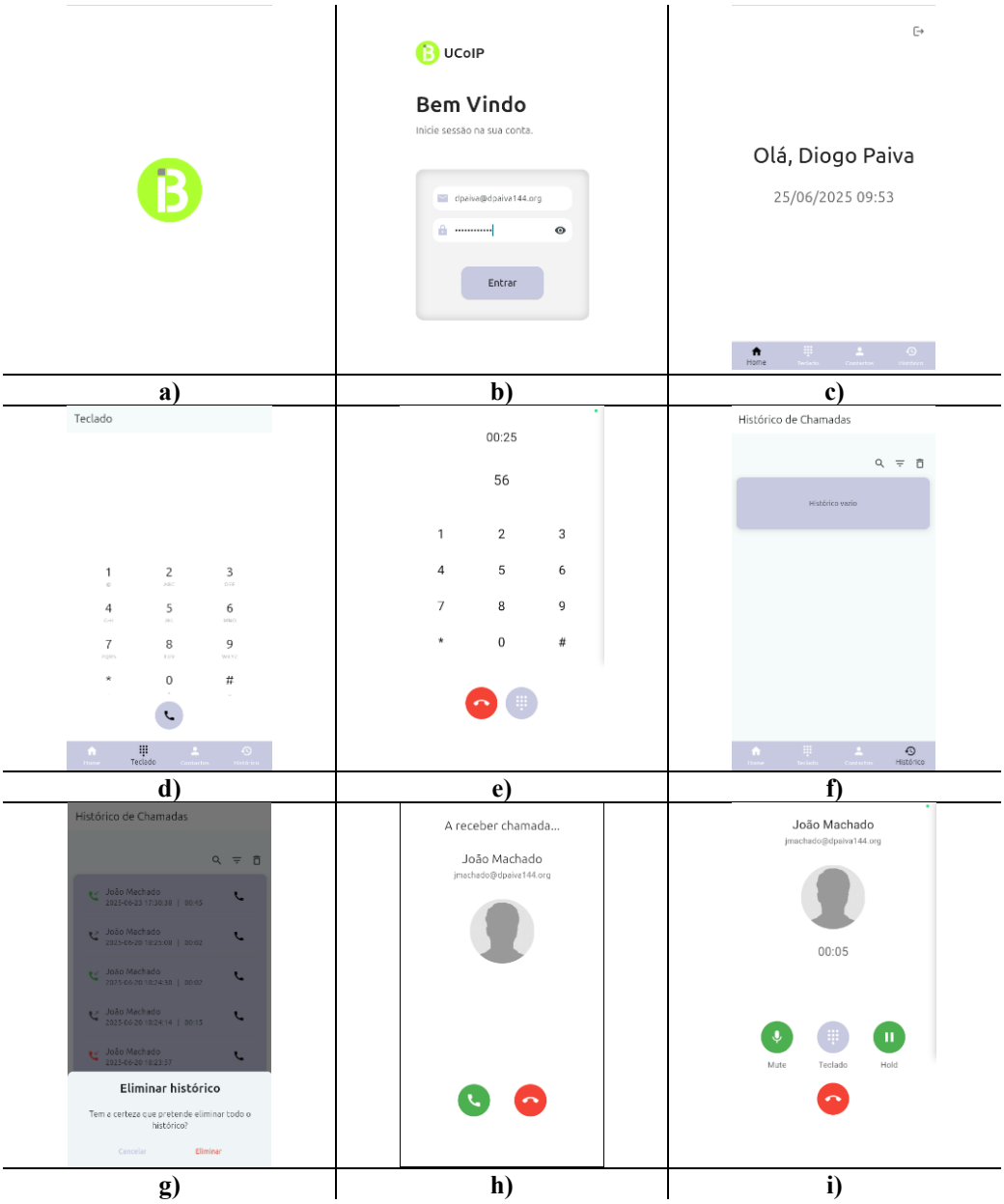


Fig. 9. Ecrãs da aplicação para equipamentos *mobile*: ecrã inicial em a), ecrã de autenticação em b), ecrã principal em c), ecrã do teclado com números inseridos em d), ecrã do teclado em marcação em e), ecrã de histórico das chamadas em f), ecrã de eliminação do histórico das chamadas em g), ecrã de chamada recebida em h) e ecrã de chamada em curso em i).

De referir que, apesar da Fig. 9 apresentar ecrãs destinados a dispositivos *mobile*, alguns deles são responsivos, ajustando a disposição dos seus elementos para ecrãs de desktop. Como exemplo, o ecrã ilustrado na Fig. 10 corresponde ao ecrã da Fig. 9 c).

Fig. 10. Ecrã principal da aplicação destinado a equipamentos desktop.

Aspetos como configuração minimalista, design, funcionalidade e robustez do *softphone* foram avaliados qualitativamente pelo responsável do projeto, colaboradores e alguns clientes do IPBRICK OS. A avaliação revelou: uma perceção positiva relativamente à simplicidade de configuração da aplicação; o design do *softphone* destaca-se pela interface intuitiva e agradável; em termos de funcionalidade, a aplicação disponibiliza recursos de utilização completos para chamadas de voz e sem opções redundantes; a robustez do sistema foi confirmada em testes práticos, mostrando estabilidade durante chamadas e fácil integração com o IPBRICK.PBX. Holisticamente, o *softphone* apresenta-se como uma solução eficiente, confiável e responde às necessidades dos utilizadores empresariais do ecossistema IPBRICK OS.

5 Conclusões e perspetivas de trabalho futuro.

Este projeto permitiu desenvolver uma aplicação multiplataforma dedicada a serviços de telefonia – um *softphone* – que comunica com os serviços de VoIP do IPBRICK OS através de APIs desenvolvidas em PHP.

Apesar dos resultados positivos alcançados, a aplicação ainda carece da implementação de funcionalidades adicionais. Assim, considera-se necessário implementar funcionalidades de *chat* e chamadas de vídeo, que poderão ser desenvolvidas recorrendo a bibliotecas do Flutter como *xmpp_stone* ou *xmpp_plugin* para mensagens e *flutter_web_rtc* ou *jitsi_meet_wrapper* para chamadas de vídeo. A interface da aplicação pode ser modificada para integrar estas funcionalidades adicionais, incluindo a criação de um ecrã de definições que permita ao utilizador personalizar a aplicação tanto a nível estético como funcional.

A implementação de notificações *push* revela-se essencial no futuro. Para dispositivos móveis, soluções como o *Firebase Cloud Messaging* (Android) e o *Apple*

Push Notification Service (iOS) permitirão alertar os utilizadores sobre chamadas pendentes, mensagens instantâneas e chamadas de vídeo. Atualmente, no caso das chamadas de voz, se um utilizador ligar para outro que não se encontre registado no servidor SIP, a chamada é direcionada para o voice-mail. Para ultrapassar esta limitação, a chamada deverá permanecer em espera no servidor enquanto este envia um pedido aos serviços de *push notifications*, que despertam a aplicação no dispositivo do destinatário, permitindo o registo no servidor SIP e a receção da chamada.

Referências

1. Kemp, S.: *Digital 2025 April Global Statshot Report.* In: DataReportal. Disponível em: https://datareportal.com/reports/digital-2025-april-global-statshot (Último acesso em 03/06/2025) (2025)
2. Shirey, R. W.: *Internet Security Glossary, Version 2.* In: RFC 4949, Aug. (2007)
3. Kissflow: *What are the Different Types of Applications?* In: Kissflow. Disponível em: https://kissflow.com/faq/what-are-the-different-types-of-applications (Último acesso em 03/06/2025) (2025)
4. Apple: *App Store.* In: Apple. Disponível em: https://www.apple.com/pt/app-store/ (Último acesso em 01/06/2025) (2025)
5. AppBrain: *Number of Android Apps on Google Play (Jun 2025).* In: AppBrain. Disponível em: https://www.appbrain.com/stats/number-of-android-apps (Último acesso em 01/06/2025) (2025)
6. El-Kassas, W. S., Abdullah, B. A., Yousef, A. H., Wahba, A. M.: *Taxonomy of Cross-Platform Mobile Applications Development Approaches.* In: Ain Shams Engineering Journal, vol. 8, no. 2, pp. 163–190 (2017)
7. Fentaw, A. E.: *Cross-Platform Mobile Application Development: A Comparison Study of React Native vs Flutter.* In: Master's Thesis, University of Jyväskylä Faculty of Information Technology, Jyväskylä (2020)
8. Farooq, S., Riaz, S., Alvi, A., Ali, A., Rehman, I.: *Cross-Platform Mobile Development Approaches and Frameworks.* In: VFAST Transactions on Software Engineering, vol. 10, pp. 79–93 (May 2022)
9. Voizcall: *WebRTC SIP Client & Softphone App: Explained.* In: Voizcall. Disponível em: https://www.voizcall.com/blog/webrtc-sip-client-softphone-app-explained (2025)
10. Cotocus: *Top 10 SIP/VoIP Clients Tools in 2025 – Features, Pros & Cons Comparison.* In: Cotocus. Disponível em: https://www.cotocus.com/blog/top-10-sip-voip-clients-tools-in-2025-features-pros-cons-comparison/ (2025)
11. CounterPath: *Bria Softphone Product Overview.* In: CounterPath. Disponível em: https://www.counterpath.com/bria/ (2024)
12. Hasby, M. A., Putrada, A. G., Dawani, F.: *The Quality Comparison of WebRTC and SIP Audio and Video Communications with PSNR.* In: Indonesian Journal on Computing, vol. 6, no. 1, pp. 73–84. https://doi.org/10.34818/INDOJC.2021.6.1.549 (2021)
13. Fowdur, T. P., Ramkorun, N., Chiniah, P. K.: *Performance Analysis of WebRTC and SIP-Based Audio and Video Communication Systems.* In: SN Computer Science, vol. 1, p. 362. https://doi.org/10.1007/s42979-020-00380-z (2020)
14. Sermersheim J.: *Lightweight Directory Access Protocol (LDAP): The Protocol*. In: RFC 4511 (2006)

Kotlin and Compose Multi-Platform App

Gonçalo Medeiros[1], Paulo Proença[1], Rui Almeida[2]

[1] Instituto Superior de Engenharia do Porto, Porto, Portugal
[2] Mindera, Portugal

Resumo. A diversidade crescente de dispositivos e sistemas operativos constitui um desafio no desenvolvimento de aplicações móveis. Este artigo descreve o desenvolvimento de uma aplicação móvel multiplataforma desenvolvida em colaboração entre a Mindera e a Federação Portuguesa de Rugby. Recorrendo a Kotlin Multiplatform e Compose Multiplatform, a solução permite gerar aplicações para Android, iOS e desktop a partir de uma única base de código, promovendo reutilização, eficiência e consistência. São discutidas as vantagens e limitações da abordagem, com vista a orientar futuros projetos nesta área. Os resultados demonstram que a abordagem multiplataforma é viável e eficiente, fornecendo uma base sólida para o desenvolvimento de aplicações móveis

Palavras-chave: Kotlin Multiplatform, Compose Multiplatform, Aplicações móveis.

1 Introdução

Este trabalho foi desenvolvido no âmbito da unidade curricular de Projeto/Estágio (PESTI), em colaboração com a empresa Mindera e a Federação Portuguesa de Rugby (FPR), surgindo como resposta à crescente necessidade de soluções que simplifiquem o desenvolvimento de aplicações móveis, num contexto em que a diversidade de plataformas e sistemas operativos aumenta significativamente os custos e a complexidade do processo, a par da necessidade de centralizar a informação atualmente dispersa da FPR.

O projeto propõe o desenvolvimento de uma aplicação móvel multiplataforma recorrendo ao uso das tecnologias *Kotlin Multiplatform* (KMP) e *Compose Multiplatform* (CMP), que permitem criar aplicações para Android, iOS e desktop a partir de uma base de código única, com elevado grau de reutilização e desempenho próximo do nativo [1].

Neste contexto, os objetivos incluem o estudo do Estado da Arte, implementação de funcionalidades essenciais, adoção de boas práticas de programação para uma solução escalável, análise das tecnologias utilizadas e documentação do processo como referência para trabalhos futuros.

Assim, o trabalho realizado com suporte na *framework* Kanban moderniza o acesso à informação da FPR, permite à Mindera testar novas tecnologias, documentando as respetivas vantagens e limitações, com utilidade para projetos futuros e promove a adoção de novas tecnologias no setor desportivo. Por fim, o projeto poderá inspirar outras organizações desportivas a adotar soluções tecnológicas semelhantes.

2 Estado da arte

2.1 Trabalhos relacionados

Diversas soluções digitais têm modernizado a forma como o rugby é acompanhado, aproximando adeptos, jogadores e federações e melhorando a experiência dos utilizadores.

Entre as aplicações analisadas, destaca-se a app do torneio *Six Nations*, que oferece cobertura em tempo real, conteúdos exclusivos, notificações personalizadas, análises interativas jogos *fantasy* rugby, focando na interação e envolvimento dos fãs [2]. Por outro lado, a aplicação da Federação Italiana de Rugby (FIR) privilegia a informação institucional e a ligação à comunidade, disponibilizando calendários, notícias, perfis de jogadores e integração com redes sociais [3].

Estas plataformas serviram como referência para o desenvolvimento do presente projeto, inspirando funcionalidades como notícias, perfis e informação atualizada dinamicamente. Contudo, não se optou pelo uso de notificações push devido a custos associados, mantendo-se em aberto a possível futura integração de jogos e análises interativas.

2.2 Tecnologias existentes

A escolha da tecnologia é essencial e depende dos requisitos do projeto. Apresentam-se, por isso, as soluções existentes para justificar a abordagem selecionada.

Soluções Multiplataforma. As tecnologias multiplataforma constituem uma abordagem moderna ao desenvolvimento de software, permitindo a criação de aplicações a partir de uma base de código única, com capacidade de compilação para diversos sistemas operativos e plataformas.

KMP, lançada em 2023 pela JetBrains, é uma tecnologia *open source* que permite partilhar código entre plataformas como Android, iOS, web e desktop, a partir de uma base única. O código *Kotlin* é transformado numa representação intermédia (IR), sendo depois compilado consoante a plataforma de destino, para *bytecode* da JVM para Android ou qualquer ambiente que suporte Java, *JavaScript* em aplicações web e binários nativos em iOS, macOS, Windows e Linux, através do *Kotlin/Native* com suporte do LLVM [4] [5].

CMP é uma *framework* para interfaces gráficas multiplataforma, baseado no Jetpack Compose do Android e mantido pela JetBrains. Suporta Android, iOS, desktop e web, permitindo o desenvolvimento de interfaces declarativas reutilizáveis. A renderização é assegurada pelo motor *Skia*, usado diretamente no Android e, nas restantes plataformas, através da biblioteca *Skiko*, garantindo desempenho e consistência visual [6] [7].

Além destas tecnologias, destacam-se outras soluções multiplataforma, como o Flutter, da Google, que utiliza a linguagem *Dart* para criar aplicações para Android, iOS, desktop e web a partir de um único código-fonte. O Flutter garante um desenho consistente da interface em todas as plataformas através do seu motor gráfico, que evoluiu do *Skia* para o mais eficiente *Impeller*.

O *React Native*, da Meta, permite desenvolver aplicações móveis em *JavaScript* com base no *ReactJS*, seguindo uma abordagem declarativa e modular próxima da experiência web. Apesar de permitir elevada reutilização de código entre plataformas, o seu desempenho é inferior ao nativo, devido à necessidade de comunicação através de uma ponte entre o motor *JavaScript* e os componentes nativos, o que introduz *overhead*.

Por fim, o *Skip* é uma tecnologia emergente centrada numa abordagem *iOS-first*, desenvolvida em *Swift* e *SwiftUI*, que gera automaticamente código equivalente para Android utilizando *Kotlin* e *Jetpack Compose*. Esta abordagem permite tirar partido do ecossistema nativo iOS, mantendo a compatibilidade com Android sem duplicação manual de lógica [1] [8] [9] [10] [11]. A Tabela 1 apresenta uma síntese comparativa das suas principais características e diferenças.

***Tabela 1**. Comparação entre as tecnologias existentes*

	KMP CMP	Flutter	React Native	Skip
Single Code	Sim	Sim	Sim	Sim
Linguagem	Kotlin	Dart	JavaScript	Swift
Execução	Compilado	Compilado	Interpretado	Compilado
Desempenho	Alto	Alto	Baixo	Alto
Render Canvas	Skia	Skia	Flexbox	Impeller/Skia
Popularidade	Alta	Muito alta	Muito alta	Alta

Injeção de dependências. A injeção de dependências é uma técnica que promove uma arquitetura mais modular, facilitando a reutilização e a testabilidade do código [12].

Koin é uma biblioteca de gestão de dependências escrita em *Kotlin*, conhecida pela simplicidade e fácil integração com projetos KMP. Combina características de injeção de dependências e *Service Locator*, oferecendo flexibilidade na resolução de dependências [13] [14].

Dagger é uma biblioteca de injeção de dependências para Android e *Java* que opera em tempo de compilação, garantindo eficiência e segurança, ao validar o grafo de dependências e evitar erros em tempo de execução [15].

Kotlin Inject é uma biblioteca de injeção de dependências totalmente escrita em *Kotlin*, que oferece segurança em tempo de compilação e uma API similar ao *Dagger*, mas mais idiomática para o ecossistema *Kotlin* [15].

Testes. A criação de *mocks* permite isolar componentes e garantir uma validação eficaz em ambientes multiplataforma KMP. As limitações da *MockK* fora da JVM tornam necessário recorrer a alternativas com suporte multiplataforma. Entre as opções mais utilizadas, *Mockative* recorre a *KSP* para gerar *mocks* compatíveis com múltiplos *targets*, enquanto a *Mokkery* utiliza um plugin de compilador para produzir *mocks* totalmente suportados em KMP [16] [17]. A Tabela 2 compara a compatibilidade das *frameworks* de *mocking* com targets KMP.

***Tabela 2**. Frameworks de mocking em KMP*

	JVM	**iOS**
Mockk	Sim	Não
Mockative	Sim	Sim
Mokkery	Sim	Sim

Base de dados em tempo real. Uma base de dados em tempo real permite a sincronização imediata de dados entre dispositivos, refletindo atualizações em milissegundos, sendo especialmente útil em contextos interativos [18].

Supabase, lançada em 2020, é uma base de dados relacional que oferece serviços de *backend*, como autenticação e armazenamento, para aplicações Android, iOS e web [18].

Firebase, criado em 2011 e adquirido pela Google em 2014, é uma base de dados *NoSQL* que disponibiliza serviços semelhantes aos do *Supabase*, incluindo autenticação e armazenamento de dados [18]. A Tabela 3 sintetiza as principais diferenças entre as bases de dados Supabase e Firebase, destacando características relevantes para aplicações multiplataforma.

***Tabela 3**. Comparação entre Firebase e Supabase*

	Supabase	**Firebase**
Tipo	Relacional	NoSQL
Autenticação	E-mail, senha e terceiros	E-mail, senha e terceiros
Armazenamento	Buckets com políticas de segurança	Google Cloud Storage para multimédia
Hospedagem	Indisponível	Disponível para conteúdo estático e dinâmico
Preços	API ilimitada (plano gratuito); até 10.000 utilizadores	Cobra por operações; utilizadores ilimitados
Self-Hosting	Disponível (open source)	Indisponível

3 Análise e desenho da solução

3.1 Análise do Problema

O projeto centrou-se na seleção nacional de rugby ("Os Lobos"), tendo em consideração a relevância e a atenção que, esta, tem vindo a obter no contexto desportivo português. A solução foi desenvolvida sob a forma de um *Minimum Viable Product* (MVP), com foco exclusivo na seleção nacional. A aplicação tem suporte multiplataforma, abrangendo os ambientes Android, iOS e desktop.

3.2 Domínio do Problema

A modelação do domínio baseou-se nos princípios do *Domain-Driven Design* (DDD). O núcleo funcional da aplicação assenta na gestão de jogos, cujos eventos influenciam diretamente as estatísticas e classificações. O domínio da aplicação abrange entidades como equipas, jogadores, treinadores, membros do staff, grupos de liga, épocas desportivas, campos de jogo, entre outros.

3.3 Requisitos Funcionais e Não Funcionais

A engenharia de requisitos é uma etapa essencial no desenvolvimento de software, definindo tanto as funcionalidades do sistema como os critérios de qualidade, nomeadamente desempenho, segurança e usabilidade [19].

Requisitos Funcionais (RF). Em termos funcionais, foram identificados os seguintes RF:

- **Consulta de jogadores:** permitir ao utilizador consultar informação detalhada sobre jogadores;
- **Notícias:** aceder a notícias da modalidade;
- **Resultados e calendário:** visualizar resultados e o calendário de jogos;
- **Classificações e equipas:** consultar classificações e dados das equipas;
- **Atualização:** garantir um mecanismo de atualização forçada da aplicação;

Requisitos Não Funcionais (RNF). Para a classificação dos requisitos não funcionais, recorreu-se ao modelo FURPS+. Assim, foram identificados os seguintes RNF:

- **Funcionalidade:** utilização de *HTTPS* nas comunicações com o servidor;
- **Usabilidade:** adotar a identidade visual da FPR e apresentar uma experiência consistente entre plataformas suportando gestos nativos;
- **Desempenho:** tempos de resposta < 100 ms em conexões ideais e fluidez ≥ 60 fps;
- **Manutenibilidade:** arquitetura modular baseada em *Clean Architecture* com padrão *MVVM*;
- **Suportabilidade:** A arquitetura deve ser modular, facilitando manutenção, escalabilidade e integração de novas funcionalidades;
- **Extra (+) Restrições de implementação:** uso de KMP e CMP, para compilação iOS é necessário *macOS*, adoção de uma arquitetura *Clean Architecture* com *MVVM*, e a cobertura por testes deve ultrapassar os 70%.

3.4 Arquitetura do Sistema

O sistema adota uma arquitetura modular baseada na *Clean Architecture*, organizada em camadas que seguem os princípios SOLID, respeitando a regra da inversão de dependências, segundo a qual as camadas internas do domínio são independentes das camadas externas. Este modelo garante que o núcleo do domínio permanece independente das camadas externas, promovendo isolamento, testabilidade e facilidade de manutenção. A arquitetura do sistema organiza-se em seis módulos principais, *Domain*, *Data*, *Presentation*, *DI*, *Shared* e *App* [20].

Inicialmente, a arquitetura era composta pelos módulos *Presentation*, *Data*, *Domain* e *Shared*. Contudo, esta estrutura revelou-se insuficiente para uma segregação eficaz de dependências, nomeadamente devido à dependência direta e desnecessária entre o arranque da aplicação e

Domain, bem como o desalinhamento estrutural causado pela coexistência, no *Shared*, de utilitários e da configuração de injeção de dependências. Os utilitários não necessitam de conhecer os restantes módulos, enquanto a injeção de dependências exige uma dependência sobre eles, o que comprometia o princípio de isolamento das dependências. Para resolver estas limitações, como demonstrado na Fig. 1, foram introduzidos dois novos módulos: *App*, responsável pela inicialização da aplicação, e *DI*, dedicado à configuração da injeção de dependências. Esta reformulação permitiu isolar a lógica partilhada no *Shared* e centralizar a gestão de dependências no *DI*, eliminando problemas de dependências circulares.

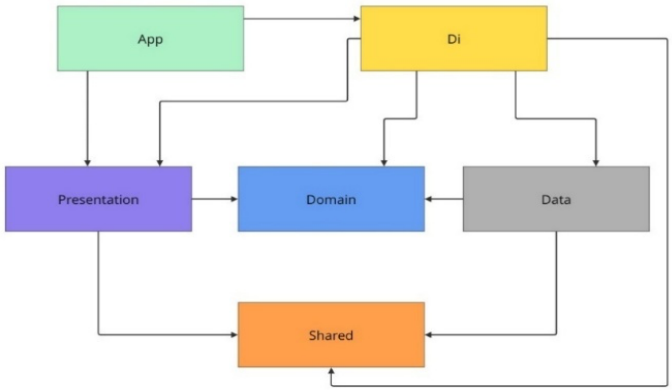


***Fig. 1**. Arquitetura do projeto*

4 Implementação da Solução

4.1 Descrição da implementação

A aplicação foi implementada com base numa arquitetura modular, com o objetivo de garantir uma separação clara de responsabilidades, promover a escalabilidade e facilitar a manutenção do sistema. O fluxo de dados e as interações entre os módulos podem ser visualizados na Fig. 2.

A disponibilização de dados inicia-se no módulo *Data*, conforme a Fig. 2, os *Data Sources* comunicam com a base de dados através da API, transformando os dados com *mappers* e encapsulando-os em *Data Transfer Objects* (DTOs). A camada de repositório executa operações de forma assíncrona através de *coroutines*, evitando bloqueios na *main thread* e tratando erros com objetos *Result*.

No módulo *Domain* encontram-se definidos os modelos de domínio, as interfaces de repositório e os casos de uso, que servem de intermediários entre o domínio e a camada de apresentação. Os casos de uso executam operações de forma assíncrona e devolvem os resultados aos *ViewModels*, evidenciado na Fig. 2, promovendo a reutilização do código e centralizando a lógica de negócio, sem impacto nas restantes camadas.

O módulo Presentation gere a interface gráfica seguindo o padrão *MVVM* e o paradigma *Unidirectional Data Flow* (UDF). Conforme detalhado na Fig. 2, cada ecrã é controlado por um *ViewModel*, responsável pela gestão de múltiplos *ViewStates* (como carregamento, sucesso ou erro) e pela execução dos casos de uso através de *coroutines*. A interface foi desenvolvida com Jetpack Compose, adotando *state hoisting* para separar lógica e renderização através de

composables stateful e *stateless*. A aplicação suporta ainda atualizações em tempo real, asseguradas por um ouvinte que deteta alterações na base de dados e atualiza automaticamente o estado apresentado ao utilizador.

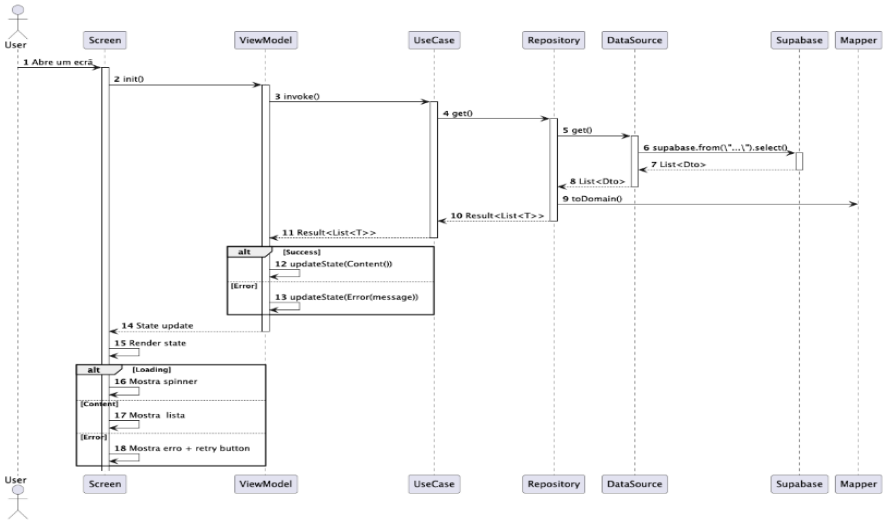


***Fig. 2**. Vista de Processo Nível 3*

O módulo *DI* centraliza a gestão de dependências com Koin, integrando os módulos de dados, domínio, apresentação e utilitários. Esta configuração garante a disponibilização consistente dos componentes ao longo do ciclo de vida da aplicação, promovendo modularidade, testabilidade e facilidade de manutenção.

O módulo *Shared* agrega recursos comuns, como *dispatchers* para operações concorrentes, suportando tarefas de I/O, processamento intensivo e execução na *main thread*, sem conhecer os restantes módulos, o que reforça a sua independência.

O módulo *App* representa o ponto de entrada da aplicação, sendo responsável pela inicialização de toda a infraestrutura base, ativação do sistema de injeção de dependências e definição da interface gráfica com *Jetpack Compose*.

Inicialmente, optou-se pela utilização da biblioteca *Voyager* para suportar *deeplinks* e gestos nativos. Contudo, a sua rigidez motivou a adoção da navegação nativa do Jetpack Compose, recorrendo a *NavHostController* e *type tokens* para assegurar uma passagem de dados segura. Com a posterior integração destas capacidades na biblioteca oficial da JetBrains, a solução adotada ficou consolidada.

4.2 Testes

Para validação do sistema foram realizados testes unitários, de integração e de interface (UI). Os testes unitários verificaram componentes isolados. Embora existam outras bibliotecas de *mocking* mencionadas compatíveis com KMP, optou-se por utilizar a *MockK*, devido à sua maior adoção e familiaridade na comunidade, o que exigiu a criação de testes específicos por plataforma para contornar as limitações fora da JVM, afetando a consistência. Os testes de integração asseguraram o funcionamento conjunto dos componentes, validando a comunicação entre camadas com dados simulados. Por fim, os testes de UI validaram a interação e apresentação da interface, com

tempos de resposta próximos de <100 ms e fluidez da UI ≳60 fps, avaliados através de ferramentas de *profiling* do IDE, adaptados por plataforma devido a limitações de compatibilidade.

A cobertura dos testes, indicador essencial da qualidade da validação, situou-se acima dos 70% nos principais módulos o módulo *Data* atingiu 75,7%, conforme ilustrado na Fig. 3; o módulo *Domain* alcançou 78,2%, como mostrado na Fig. 4; e o módulo *Presentation* registou 73,2%, segundo a Fig. 5. Apesar de a cobertura elevada não garantir ausência de erros, aumenta a confiança e reduz regressões, limitada pela falta de testes multiplataforma.

data: Overall Coverage Summary

Package	Class, %	Method, %	Branch, %	Line, %	Instruction, %
all classes	75.7% (48/64)	72.2% (85/118)	65.3% (139/212)	68.4% (1668/2439)	62.1% (4627/7451)

***Fig. 3**. Cobertura do Módulo Data*

domain: Overall Coverage Summary

Package	Class, %	Method, %	Branch, %	Line, %	Instruction, %
all classes	78.2% (20/26)	74.8% (25/33)		61.1% (87/143)	57.1% (397/696)

***Fig. 4**. Cobertura do Módulo Domain*

presentation: Overall Coverage Summary

Package	Class, %	Method, %	Branch, %	Line, %	Instruction, %
all classes	73.2% (136/186)	64.8% (219/337)	49.6% (275/554)	59.2% (1371/2317)	54.9% (12348/22492)

***Fig. 5**. Cobertura do Módulo Presentation*

4.3 Avaliação da solução

A avaliação da solução baseou-se na realização de testes unitários, de integração e de UI, garantindo o funcionamento correto e a conformidade com os requisitos. Estes testes permitiram identificar erros precocemente e assegurar uma experiência de utilização consistente. Complementarmente, a aplicação foi avaliada pelo supervisor, Rui Almeida e *stakeholder* do projeto, Pedro Seruca, através de um questionário de usabilidade numa escala de 0-5, cujos resultados, detalhados na Tabela 4, foram positivos, reforçando a confiança na solução e apontando oportunidades para melhorias.

***Tabela 4**. Resultados Questionário*

	Pedro Seruca	**Rui Almeida**
Os dados são carregados corretamente?	5	5
A UI é consistente com o design?	4	4
As pesquisas funcionam corretamente?	5	5
A app funciona em diferentes dispositivos/sistemas?	5	5
O desempenho é consistente em listas longas?	5	5
A navegação é consistente nas plataformas nativas?	4	4
A navegação é confortável com uma mão? (mobile)	5	5
A aplicação é estável e livre de bugs?	4	4
Informação fácil de encontrar?	5	5

5 Vantagens e Limitações

5.1 Vantagens

Destaca-se a partilha eficiente da lógica de negócio entre várias plataformas a partir de uma única base de código, o que reduziu esforço e facilitou a manutenção. A reutilização das interfaces garantiu consistência visual e comportamental, enquanto a arquitetura modular garantiu escalabilidade ao projeto. A integração de novas plataformas torna-se facilitada, e as funcionalidades nativas foram asseguradas, suportadas por uma comunidade ativa e boa documentação.

5.2 Limitações

Foram identificadas limitações relevantes, como a incompatibilidade da biblioteca de *mocks* com KMP/CMP, que exigiu testes unitários separados por plataforma, e a ausência de suporte multiplataforma para testes de UI. Inicialmente, a navegação nativa não suportava gestos e *deeplinks* no iOS, embora tenha sido corrigida em versões posteriores. Por fim, algumas bibliotecas, como a de navegação para desktop, apresentaram instabilidade.

6 Conclusões

Foi possível concluir que o projeto atingiu todos os objetivos inicialmente propostos.

Durante o desenvolvimento, enfrentaram-se algumas limitações, como atrasos na entrega de designs por parte da equipa de design, que condicionaram a implementação visual e funcional de certos componentes. As tecnologias adotadas apresentaram também restrições, nomeadamente a ausência de suporte direto para testes unitários e de UI multiplataforma, contrariando o objetivo de manter uma base de código totalmente comum.

Apesar destes constrangimentos, o projeto representa uma base promissora para futuros desenvolvimentos. A aplicação criada tem potencial de expansão, com adoção inicial prevista para a seleção nacional de rugby e, futuramente, para todo o panorama da modalidade em Portugal.

Agradecimentos. Agradeço ao ISEP e aos seus docentes, em especial ao Professor Eng. Paulo Proença, pelo acompanhamento ao longo da licenciatura e deste projeto. À Mindera, agradeço a oportunidade de estágio curricular e ao Eng. Rui Almeida pelo suporte técnico essencial.

Bibliografia

1. Stanic, N., Ćirković, S.: Analysis of Approaches to Developing Kotlin Multiplatform Applications and Their Impact on Software Engineering. 53–59 (2024).
2. Six Nations Rugby: Six Nations Mobile App. https://www.sixnationsrugby.com/en/app, last accessed 2025/04/22
3. Federazione Italiana Rugby: Federazione Italiana Rugby - Official App. Mac App Store. https://apps.apple.com/mt/app/federazione-italiana-rugby/id1074242258, last accessed 2025/04/22

4. Weninger, M.: Tracing Performance Metrics in Kotlin Multiplatform Projects. Master's thesis, Austria (2024)
5. Mir, S.: Kotlin Multiplatform - The Future of Cross-Platform Development. Medium (2024). https://medium.com/@mrsaykatm4/kotlin-multiplatform-the-future-of-cross-platform-development-5b95a548879c, last accessed 2025/03/21
6. Exyte: Jetpack Compose Multiplatform Android & iOS. ProAndroidDev (2023). https://proandroiddev.com/jetpack-compose-multiplatform-android-ios-4a87ba417caa, last accessed 2025/03/29
7. Kella, S.: Skia in Jetpack Compose: The Core Graphics Engine. Medium (2024). https://medium.com/@sandeepkella23/skia-in-jetpack-compose-the-core-graphics-engine-77e4f34b6695, last accessed 2025/03/25
8. Aripirala, S.S., Airan, P., Peddint, D.R.: The Polyglot's Playground: Navigating KMP, Flutter, and React Native in Cross-Platform Ecosystems. International Journal for Research in Applied Science and Engineering Technology 12 (2024)
9. Skip Tools: Skip and Kotlin Multiplatform (2024). https://skip.tools/blog/skip-and-kotlin-multiplatform/#caveats, last accessed 2025/03/14
10. Skip Tools: Skip Comparison Matrix. https://skip.tools/compare/, last accessed 2025/03/15
11. Emerline: React Native vs Flutter: Which Is Better for App Development? (2023). https://emerline.com/blog/flutter-vs-react-native, last accessed 2025/03/16
12. Android Developers: Dependency injection in Android. https://developer.android.com/training/dependency-injection, last accessed 2025/03/31
13. Chiriac, A.L.: Development of a multi-platform application for the University Virtual Campus. Bachelor's thesis, Universitat Oberta de Catalunya, Barcelona, Spain (2025)
14. Koin: Why Koin? https://insert-koin.io/docs/setup/why/#koin-a-dependency-injection-framework, last accessed 2025/05/10
15. Porciúncula, F.: From Dagger & Hilt into the multiplatform world with kotlin-inject (2023)
16. Mockative: Mocking for Kotlin/Native and Kotlin Multiplatform using KSP. GitHub (2025). https://github.com/mockative/mockative
17. Lupuuss: Mokkery - Mocking library for Kotlin Multiplatform. GitHub. https://github.com/lupuuss/Mokkery, last accessed 2025/11/24
18. Amanuel, A.: Supabase vs Firebase: Evaluation of performance and development of Progressive Web Apps. Master's thesis, Metropolia University of Applied Sciences, Finland (2022)
19. GeeksForGeeks: Functional vs. Non-Functional Requirements. https://www.geeksforgeeks.org/functional-vs-non-functional-requirements/, last accessed 2025/04/14
20. Martin, R.C.: Clean Architecture: A Craftsman's Guide to Software. Prentice Hall (2018)
21. Sczip, J.G.B.: Clean Architecture - A Little Introduction. Medium (2020). https://medium.com/swlh/clean-architecture-a-little-introduction-be3eac94c5d1, last accessed 2025/06/22

Development of a mass drug administration monitoring system for a Humanitarian NGO

Rafael Ribeiro[1,2], Cláudio Teixeira[2], Pedro Vicente[2], and Paulo Baltarejo Sousa[1]

[1] School of Engineering/Polytechnic Institute of Porto, Porto, Portugal
{1190977,pbs}@isep.ipp.pt
[2] Mindera, Porto, Portugal
{rafael.ribeiro,claudio.teixeira,pedro.vicente}@mindera.com

Abstract. Mass drug administration is the process of distributing medication to people who are in areas with risk of neglected tropical diseases. The monitoring of this process is supported by the application of forms in the communities, which are then sent to the headquarters, where they are manually entered into governmental health information systems. This paper aims at finding a solution for automating the process, making it more efficient, reliable and scalable.
The proposed solution involves the usage of Optical Character Recognition (OCR) to extract the data, and is composed of a Backend, an Android Application and a Backoffice. It was evaluated using several metrics, achieving mean values of 0.8% Character Error Rate (CER), 2.5% Word Error Rate (WER), 3.7% Field Error Rate (FER) and 96.2% Precision, Recall and F1 score. Forms filled in English and forms filled using print handwriting style had significantly better results than forms filled in Portuguese and forms filled using cursive style, respectively. The application was used by two testers from the Nongovernmental Organization (NGO), resulting in positive feedback, with some improvements being suggested.

Keywords: Optical Character Recognition · Text extraction · Mass drug administration · Web Information System · Android development.

1 Introduction

The MENTOR Initiative is a Nongovernmental Organization (NGO) that works with the most vulnerable and hard to reach communities to reduce death and suffering from tropical diseases [20]. They cooperate closely with local communities, health workers, health authorities and other international organizations in insecure and high risk environments, such as conflict zones and other humanitarian emergencies [14].

Mass drug administration is one of the key interventions carried out by The MENTOR Initiative to prevent the risk of neglected tropical diseases, and its monitoring is essential for assessing the coverage and identify communities in

need [11]. The monitoring process involves collecting data from the field using forms that contain information about the number of drugs distributed, side effects, inventory and supervision of the drug administration. The collected data is analysed and taken into account for decision-making, allowing the adoption of proper strategies and an effective resource allocation.

The monitoring process is based in the communication between the people appointed to distribute the medicine in the villages and the country headquarters. For each village, multiple paper forms with several pages are filled, which makes the data compilation difficult, inefficient and error-prone. The collected data is sent to the country headquarters, where a person is responsible for manually submitting the data into governmental health information systems. Besides the inefficiency, this process is both time-consuming and error-prone.

The main goal of the project is the development and evaluation of a tool that allows the digitization of mass drug administration forms, as well as the automation of data submission for different countries and NGOs.

This paper is structured as follows. Section 2 presents the state of the art about the text extraction technology and the available tools. Section 3 describes the analysis, design and implementation of the proposed solution. In Section 4, the developed solution is evaluated through experimentation. Finally, Section 5 presents the conclusions and future work about the project.

2 State of the Art

Optical Character Recognition (OCR) is a technology that extracts text from different types of documents, such as photos or scanned documents, so that it can then be indexed, stored or analysed. This technology serves as the foundational component of the proposed solution for the problem.

The concept of "reading documents by other than human means" appeared during World War II, motivated by the necessity of automating the transcription of documents [16]. During the early ages of OCR, it was also used to support visually impaired individuals [1], as well as reading musical notations and small words, even though it had a lot of limitations [2]. The commercially available OCR systems originated in early 1960s [1], and evolved into the latest generations, in which Machine Learning (ML) techniques, such as Artificial Neural Networks (ANN), were introduced, leading to a wider supportability and higher accuracy [1, 12].

Most of the modern OCR systems include multiple phases, such as preprocessing, segmentation, feature extraction, classification and post-processing [7, 19, 8, 13]. During the classification phase, several methods can be used, such as ML techniques, Template Matching and Structural Pattern Recognition [12].

The technology is widely used in multiple areas, such as document digitization, archiving, vehicle plate recognition, language translation and accessibility. However, there are some challenges to its implementation, like supporting multiple languages, recognizing handwritten text and dealing with low quality input images.

The literature was reviewed in order to understand the feasibility of using OCR for handwritten text extraction, answering the research question "Are there case studies about solutions that use OCR to extract handwritten text with accuracy?".

Several case studies were found using OCR for handwritten text recognition in real-world scenarios. One of them is [15], in which the technology is being used for the digitization of child protection cards. A custom solution was built using ML, achieving accuracies between 93% and 98% [15].

Both [21] and [9] explored the application of OCR for historical documents with crowdsourced post-correction. The first study focus on ancient manuscripts and reported difficulty in maintaining a stable community of contributors. The second study focus on historical Dutch documents, in which recognizing person names was a challenge.

In [18], a solution was developed for automating mark entry in educational institutions. It significantly accelerated data processing workflows by efficiently converting and processing the marks of students. The main challenge was detecting decimal numbers [18].

The case study [17] used an OCR cloud service to develop a social pension management system. The solution had a high accuracy, but struggled at differentiating uppercase and lowercase letters, as well as struck-out text [17].

In the case study [5], the recognition of handwritten code from Computer Science students was explored. When combining an OCR cloud service with post-correction techniques, it proved to be accurate enough to be used in real learning environments [5]. One of the main challenges of the solution is dealing with struck-out text [5].

The literature was also reviewed to find the OCR tool with higher accuracy for handwritten text, answering the research question "What studies are there about OCR tools with the best accuracy for handwritten characters?".

The OCR tools can be divided into three categories: proprietary software (e.g., ABBYY FineReader and Transym OCR), open source engines (e.g., Tesseract and Calamari) and online services (e.g., AWS Textract, Google Document AI and Azure Document Intelligence) [6].

The study [3] compares Tesseract, AWS Textract and Google Document AI, using a dataset that includes scanned documents in English and Arabic. The results suggest that the cloud services perform significantly better than Tesseract, with Google Document AI achieving a slightly higher accuracy than Textract.

In [4], all the types of OCR tool were analysed, by comparing Google Vision AI (cloud service), Tesseract (open source), ABBYY FineReader and Transym OCR (proprietary). The dataset included key-frames extracted from lecture videos with variations in the text orientation, noise and skewness. Google Vision AI was the tool that performed better, with an average accuracy of 96.7%.

In the comparative study [10], the tools Tesseract, AWS Textract and Google Vision AI were compared using a dataset that includes printed and handwritten medical records. Both cloud services performed significantly better than Tesser-

act, with AWS Textract achieving a slightly higher accuracy for handwritten text recognition, with 88.89% accuracy.

In general, the application of OCR for recognizing handwritten text has proven to be accurate in many case studies, either by using commercially available OCR tools or training custom ML models. The main challenges found were struck-out text and noisy images. Regarding the commercially available tools, the results suggest that the cloud services have better performance compared to the other types of tool, with the services from AWS (Textract) and Google (Vision AI and Document AI) achieving higher accuracy for handwritten text.

3 Development

This section presents an analysis of the problem and outlines the design for the proposed solution, with its implementation being described.

In order to understand the problem and specify the requirements, the domain was modelled into entities and relationships, as represented by the diagram in Fig. 1.

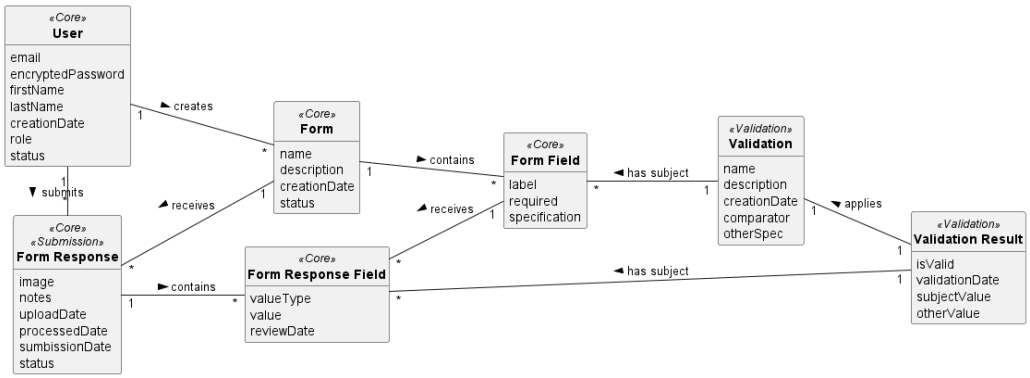


Fig. 1. Simplified domain model

The base concept of the domain is the Form entity, which represents the structure of a form that can be filled, being related to multiple Form Fields. The Form Response entity represents an uploaded filled response, being related to its respective Form Response Fields. As for the Validation entity, it specifies a comparison between Form Field values to validate their correctness. When it is applied to specific Form Response Fields, the Validation Result entity indicates whether it was successful or not. The User is a supporting entity that represents the user of the system.

Considering the domain model, the functional requirements were specified as use cases. A regular user can upload photos of filled forms, review the extracted data, apply validations and submit form responses. Besides that, an administrator can also register new users and define the structure of forms and validations.

The architecture of the system was designed as shown in Fig. 2.

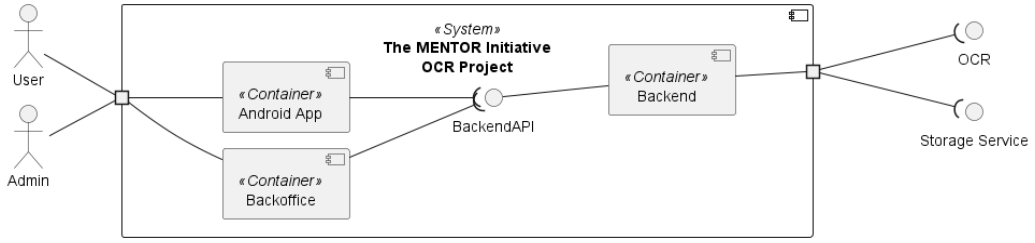


Fig. 2. Component diagram of the architecture of the system

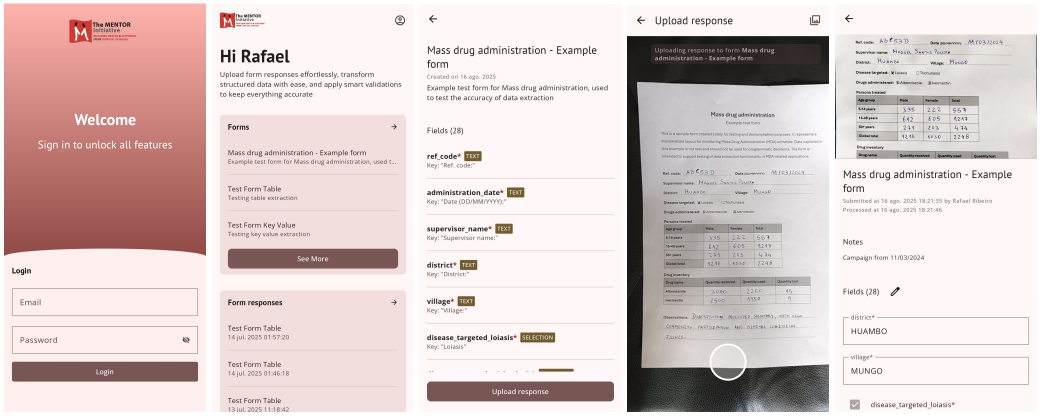
The system is composed of three components: Backend, Android App and Backoffice. The Backend is responsible for the business logic and data management, serving as a foundation for the frontend components by exposing a Representational State Transfer (REST) Application Programming Interface (API). It interacts with the external services, for data extraction and image storage. The Android App component allows the users to interact with the system and perform the main actions, such as uploading photos and applying validations. As for the Backoffice component, it allows the administrators to manage the system, such as creating new users and editing the structure of forms or validations.

The architecture of the Backend is based on a combination of Model-View-Controller (MVC) and layered architecture. It was built using NodeJS with the programming language TypeScript and the framework Express, which is responsible for exposing the REST API. PostgreSQL was used for the database, with the Object-Relational Mapping (ORM) framework Prisma being used to handle the database operations and migrations. The uploaded images were stored using AWS S3. For the data extraction, AWS Textract was used as the main OCR tool, considering the results of the literature review, with Google Gemini being used as a fallback tool to improve the extraction accuracy. The output from both tools is processed according to the form fields defined and their respective types.

A Model-View-ViewModel (MVVM) architecture was followed in the Android App, which was built natively using the programming language Kotlin. The framework Jetpack Compose was used to build the User Interface (UI) in a declarative style, with Jetpack Navigation being responsible for handling the navigation of the application. Retrofit was used for the network requests, while Jetpack DataStore was used for internal storage. The Fig. 3 includes some screenshots from the Android App.

After entering the credentials in the login screen, Fig. 3 (a), the user is redirected to the home screen, Fig. 3 (b). By accessing the details screen of a form, Fig. 3 (c), the user can upload a photo using the camera or gallery, Fig. 3 (d). When the upload is successful, the user is redirected to the details screen of the form response, Fig. 3 (e), with the extracted data.

The Backoffice follows a component-based architecture, being built using React with the programming language TypeScript, in a NodeJS environment.



(a) (b) (c) (d) (e)

Fig. 3. Android App screenshots of the main screens

This component was implemented in a partnership with Mindera School, which is a learning programme that introduces young people to the world of technology and programming.

4 Experimentation & Evaluation

Two experiments were conducted to evaluate the accuracy of the text extraction in the developed solution, as well as its usability and efficiency.

The first subsection will focus on measuring the accuracy of the text extraction, as well as testing the conditions that may affect it. The second subsection includes the analysis of the analytics data and the direct feedback from the testers.

4.1 Text extraction accuracy

In this context, the accuracy represents the quality and classification performance of the text extraction, which will be evaluated with a set of appropriate metrics. The accuracy of text extraction was tested using an example test form that was filled by multiple people, following a non-probability convenience sampling technique. Some forms were filled in English, while others were filled in Portuguese. Different styles of handwriting were also used, such as cursive or print handwriting. Each form response was captured in two different ways: by taking a picture with the camera inside the application or by scanning the form and picking the scanned image from the gallery.

Several metrics were collected at different depths. The Character Error Rate (CER), Word Error Rate (WER) and Field Error Rate (FER) were used to quantify the percentage of characters, words and fields that were incorrectly extracted, respectively. Besides that, the Precision was measured to determine

the proportion of correctly extracted fields out of the fields that were extracted, while the Recall determines the proportion of correctly extracted fields out of the number of fields that should have been extracted. The F1 score represents the harmonic mean between the Precision and Recall.

The form was filled by 8 individuals, with 16 images being uploaded to the application. Table 1 summarizes the results that were achieved.

Table 1. Summary table for accuracy tests dataset

Metric	Mean	Standard deviation	Minimum	Maximum
CER	0.008	0.009	0.000	0.026
WER	0.025	0.026	0.000	0.063
FER	0.037	0.040	0.000	0.107
Precision	0.962	0.040	0.892	1.000
Recall	0.962	0.040	0.892	1.000
F1 score	0.962	0.040	0.892	1.000

The results indicate that the text extraction accuracy is generally high for the forms used, with low mean error rates (**0.8%** CER, **2.5%** WER and **3.7%** FER) and high performance metrics (**96.2%** Precision, Recall and F1 score). The higher error rates at word and field level can be explained by their lower granularity, which causes an incorrect character to have a higher impact on the value. The parity of the field-level metrics can be explained by the lack of fields in which the system could not extract a value.

To further analyse the results, the data was grouped by the method of capturing the form, language and handwriting style. Statistical tests were performed in order to compare the distributions for each group and determine if their differences are statistically significant. Given the small sample size and the non-normal distribution of the data, non-parametric tests were used.

In the performed tests, the null hypothesis states that there are no differences between the distributions. Considering the mean values for each group, the alternative hypothesis states that the error rates are lower and the field-level performance metrics are higher for scanned images, forms filled in English and forms filled with print handwriting style. Table 2 includes the results of the statistical tests.

Considering a significance level of 5%, there is not enough evidence to conclude that the scanned images allow for significantly better results than taking the picture with the camera inside the application. Conversely, there is enough evidence to conclude that the forms filled in English had significantly better results than the forms filled in Portuguese. For the handwriting style, it is also possible to conclude that the forms filled with print handwriting style had significantly better results than the forms filled with cursive handwriting style.

Table 2. Statistical tests for differences in grouped distributions

	CER	WER	FER	Precision	Recall	F1 score
Scanned vs Camera (upload type)						
Skewness	-1.132	-1.627	-1.564	1.564	1.564	1.564
p-value	0.0625*	0.25*	0.25*	0.25*	0.25*	0.25*
English vs Portuguese (language)						
p-value	0.0018**	0.0017**	0.0023**	0.0023**	0.0023**	0.0023**
Print vs Cursive (handwriting style)						
p-value	0.0307**	0.0178**	0.0195**	0.0195**	0.0195**	0.0195**

*Sign test; **Mann-Whitney U test

4.2 User testing

The user testing session was conducted in a partnership with two persons from the NGO, with the objective of gathering feedback from the target users and evaluate the extraction of fields. Before starting the testing, a demonstration meeting was held to showcase the application and provide instructions. The tests were performed during the first week of August 2025, with official forms being used, filled with fake data due to confidentiality reasons.

After performing the tests on the application, a meeting was held with the testers. The feedback gathered from them during the meeting was generally positive, with some annotations being shared about their experience using the application. Even though the solution is missing the capability to submit the form response directly to the health information system of the partner, the testers saw the potential of the application to improve their workflow.

Some points for improvement were mentioned, such as the ordering of the fields in the form response details screen, the capability to delete an uploaded form response and the size of the numeric fields in the form response details screen. Besides that, the testers mentioned that validations could be more intuitive and be applied automatically after uploading an image.

To monitor the usage of the application during the tests, Firebase Analytics was integrated with custom events, tracking the extraction rate, which indicates the proportion of fields that the system managed to extract a value, the review rate, indicating the proportion of fields that were reviewed, and the result of the validations. The extraction and review rates do not represent real accuracy metrics, as the fields can be extracted incorrectly or be left empty.

A total of 10 form responses were uploaded and had their fields extracted during the testing session. The mean extraction rate was **82.8%**, which means that, on average, 17.2% of the fields could not be extracted from the uploaded forms. Through observation, it was possible to notice that the forms were partially filled, leading to the system not being able to extract a value from fields that were empty. The mean review rate was **25%**, which could have been caused by the empty fields. During the tests, 18 validations were applied, with **10 (56%)** of them returning a valid result.

5 Conclusions

This paper aimed at finding a solution to improve the efficiency of the paper-based mass drug administration process. A solution with a Backend, an Android App and a Backoffice was designed and implemented, using AWS Textract and Google Gemini for the text extraction. It achieved a mean value of 0.8% CER, with 96.2% Precision at field level, which is similar, or even better in some cases, compared to the results found in the literature. It was possible to conclude that filling forms in English and with print handwriting style resulted in significantly better results than filling in Portuguese and using cursive style, respectively. On the other hand, there were no significant differences for the upload type. The proposed solution showed potential to improve the efficiency of the process, which can lead to a faster decision-making and, consequently, a more timely and accurate response to neglected tropical diseases.

The main goal was mostly completed, except for the automation of data submission. The lack of time and the complexity of the project were the biggest challenges during the project. Another limitation was the small sample size in the experimentation. The main piece of future work is the integration with the external health information systems to automate the submission of form responses.

References

1. Berchmans, D., Kumar, S.S.: Optical character recognition: An overview and an insight. 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies, ICCICCT 2014 pp. 1361–1365 (12 2014). https://doi.org/10.1109/ICCICCT.2014.6993174
2. Dubey, M.R.: Machine learning in the field of optical character recognition (ocr). International Journal of Trend in Scientific Research and Development (IJTSRD) **4** (2020)
3. Hegghammer, T.: Ocr with tesseract, amazon textract, and google document ai: a benchmarking experiment. Journal of Computational Social Science **5**, 861–882 (5 2022). https://doi.org/10.1007/S42001-021-00149-1/FIGURES/10
4. Hukkeri, G.S., Goudar, R.H., Janagond, P., Patil, P.S.: Machine learning in ocr technology: Performance analysis of different ocr methods for slide-to-text conversion in lecture videos. International Journal of Advanced Computer Science and Applications **13** (2022). https://doi.org/10.14569/IJACSA.2022.0130839
5. Islam, M.S., Doumbouya, M.K.B., Manning, C.D., Piech, C.: Handwritten code recognition for pen-and-paper cs education. L@S 2024 - Proceedings of the 11th ACM Conference on Learning @ Scale pp. 200–210 (7 2024). https://doi.org/10.1145/3657604.3662027
6. Jain, P., Taneja, K., Taneja, H.: Which ocr toolset is good and why : A comparative study. Kuwait Journal of Science **48**, 1–12 (4 2021). https://doi.org/10.48129/KJS.V48I2.9589
7. Karthick, K., Ravindrakumar, K.B., Francis, R., Ilankannan, S.: Steps involved in text recognition and recent research in ocr; a study. International Journal of Recent Technology and Engineering (IJRTE) **8**, 2277–3878 (2019)

8. Kaur, A., Baghla, S., Kumar, S.: Study of various character segmentation techniques for handwritten off-line cursive words: A review. International Journal of Advances in Science Engineering and Technology **3**, 2321–9009 (2015)
9. Koolen, M., Hoekstra, R., Oddens, J., Sluijter, R., Koert, R.V., Brouwer, G., Brugman, H.: The value of preexisting structures for digital access: Modelling the resolutions of the dutch states general. Journal on Computing and Cultural Heritage **16**, 2019–2024 (6 2023). https://doi.org/10.1145/3575864/ASSET/D262D520-E752-4D06-8012-329CCFB74EEF/ASSETS/GRAPHIC/JOCCH-21-0143-F08.JPG
10. Kumar, A., Singh, P., Lata, K.: Comparative study of different optical character recognition models on handwritten and printed medical reports. International Conference on Innovative Data Communication Technologies and Application, ICIDCA 2023 - Proceedings pp. 581–586 (2023). https://doi.org/10.1109/ICIDCA56705.2023.10100213
11. Mass drug administration - the mentor initiative, https://mentor-initiative.org/activity/neglected-tropical-diseases/mass-drug-administration/
12. Memon, J., Sami, M., Khan, R.A.: Handwritten optical character recognition (ocr): A comprehensive systematic literature review (slr). IEEE Access **8**, 142642–142668 (2020). https://doi.org/10.1109/ACCESS.2020.3012542
13. Nguyen, T.T.H., Jatowt, A., Coustaty, M., Doucet, A.: Survey of post-ocr processing approaches. ACM Computing Surveys (CSUR) **54**, 1 – 37 (2021). https://doi.org/10.1145/3453476
14. Our vision - the mentor initiative, https://mentor-initiative.org/our-vision/
15. PantDevesh, TalukderDibyendu, SethAaditeshwar, PantDinesh, SinghRohit, DuaBrejesh, PandeyRachit, MaruthiSrirama, JohriMira, AroraChetan: Robust ocr pipeline for automated digitization of mother and child protection cards in india. ACM Journal on Computing and Sustainable Societies **1**, 1–24 (9 2023). https://doi.org/10.1145/3608114
16. Robertson, A.: Optical character recognition: A study of success and failure in innovation. Management Decision **9**, 213–223 (1971). https://doi.org/10.1108/EB000971
17. Samonte, M.J.C., Bejar, A.M.L., Bien, H.C.L., Cruz, A.M.D.: Senior citizen social pension management system using optical character recognition. ICTC 2019 - 10th International Conference on ICT Convergence: ICT Convergence Leading the Autonomous Future pp. 456–460 (10 2019). https://doi.org/10.1109/ICTC46691.2019.8940013
18. Shaju, A.T., Jo, J.T., Abraham, E.S., Vishnuprasad, K.G., Deepa, V., Mathew, J.: Document digitization using optical character recognition - a case study of mark entry automation in educational institutions. 2024 1st International Conference on Trends in Engineering Systems and Technologies, ICTEST 2024 (2024). https://doi.org/10.1109/ICTEST60614.2024.10576116
19. Thanki, J.D., Davda, D., Swaminarayan, P.: A review on ocr technology **8** (2021)
20. The mentor initiative, https://mentor-initiative.org/
21. Wecker, A.J., Raziel-Kretzmer, V., Kiessling, B., Ezra, D.S.B., Lavee, M., Kuflik, T., Elovits, D., Schorr, M., Schor, U., Jablonski, P.: Tikkoun sofrim: Making ancient manuscripts digitally accessible: The case of midrash tanhuma. ACM Journal on Computing and Cultural Heritage (JOCCH) **15** (4 2022). https://doi.org/10.1145/3476776

Property-based Testing

André Leal Oliveira[1], Fabiana Manuela Alves[3], Fernando Carvalho[1], Luís Afonso[1], Jorge Mendonça[1], and António Sousa[1,2]

[1] IPP-ISEP Instituto Superior de Engenharia do Porto, Porto, Portugal
[2] LEMA - Laboratório de Engenharia Matemática, Porto, Portugal ats@isep.ipp.pt
[3] Capgemini Engineering

Resumo. Este manuscrito apresenta o desenvolvimento e aplicação de testes unitários e de testes baseados em propriedades (*Property-Based Testing* — PBT), utilizando a biblioteca Hypothesis, dedicada ao suporte de PBT em Python. A abordagem iniciou-se com a criação de testes PBT para pequenos exemplos de código, explorando a capacidade da ferramenta para gerar automaticamente conjuntos diversificados de entradas, incluindo alguns casos extremos, permitindo analisar e validar as propriedades do software. Posteriormente, esses testes foram aplicados à biblioteca py-money e à *Embedded Test Automation Framework* (ETAF) da Capgemini Engineering, que já dispunham de alguns testes unitários. Assim, para potenciar o seu melhor desempenho, são usados testes unitários, complementados com testes PBT para ultrapassar algumas das suas falhas, particularmente, em cenários envolvendo a biblioteca py-money operando com os diversos tipos de moedas mundiais – onde o número de casas décimais pode variar ou simplesmente não ser usado. A formulação de propriedades PBT e a geração automática de casos extremos pela biblioteca Hypothesis acelera a identificação de erros, reduzindo o esforço manual necessário para implementação de testes dedicados a casos específicos. A avaliação comparativa entre os dois testes considerou os resultados obtidos, o tempo de execução e a cobertura dos cenários a analisar, corroborando a utilidade dos testes baseados em propriedades como um complemento para o incremento da robustez do software. Conclui-se que a geração automática de casos de teste diversificados e críticos antecipa a identificação de erros, aumentando significativamente a cobertura e a robustez dos sistemas testados.

Palavras-chave: *Property-Based Testing* · Hypothesis · Python · py-money · ETAF

1 Introdução

Os testes de software são componentes essenciais do ciclo de desenvolvimento de software, tendo como objetivo verificar e validar se um sistema cumpre requisitos e funciona conforme o esperado. Segundo a norma IEEE Std 610.12-1990, o teste consiste em analisar um componente ou sistema para detetar falhas [1]. Essa atividade pode ser classificada em diversos níveis, como testes unitários, de

integração, sistema e aceitação, cada um com objetivos e alcances específicos, conforme detalhado em Spillner et al. [2].

A abordagem explorada neste trabalho combina *Property-Based Testing* (PBT) com metodologias baseadas em testes unitários. O PBT define propriedades gerais que o software deve satisfazer, em vez de casos específicos, e usa ferramentas que geram automaticamente uma ampla variedade de entradas para verificar essas propriedades. Esta metodologia permite identificar falhas que podem passar despercebidas em testes tradicionais e segue o princípio do *shift-left testing*, antecipando a detecção de erros nas fases iniciais do desenvolvimento.

Para empresas do setor tecnológico, a aplicação e avaliação de metodologias inovadoras de teste, contribui para a melhoria da qualidade dos seus produtos internos, aumentam a robustez dos sistemas entregues aos clientes e reforçam a inovação e competitividade da organização num mercado cada vez mais exigente.

Nas secções seguintes, apresenta-se o enquadramento teórico detalhado sobre as metodologias de testes de software, com foco em PBT e nas ferramentas associadas. Segue-se a descrição da metodologia adotada, abrangendo o processo de aprendizagem, implementação dos testes e sua aplicação prática. A secção de resultados destaca os principais resultados, comparando eficácia, cobertura e desempenho entre os testes unitários tradicionais e os baseados em propriedades. Por fim, discute-se a contribuição do estudo desenvolvido, as suas limitações e propostas para trabalho futuro, encerrando com conclusões que evidenciam a relevância da abordagem implementada para a melhoria da qualidade do software.

2 Testes de Software

A implementação de testes unitários está na base do desenvolvimento incremental e verificação da qualidade do código, permitindo a deteção precoce de erros, facilitando depurações e revisões seguras. Estes testes focam-se em unidades isoladas, como funções e métodos, que garantem o seu funcionamento. Apesar das vantagens em modularidade e manutenção, a sua capacidade de detetar falhas em cenários complexos é limitada, especialmente em funções que envolvem múltiplas condições ou entradas inválidas [1, 2]. Para colmatar estas limitações, as metodologias tradicionais têm evoluído, incluindo técnicas de cobertura de código e análises de risco que visam assegurar que a maior parte do código seja testada de forma eficaz. No entanto, a implementação extensiva de testes unitários pode ser dispendiosa e trabalhosa, produzindo uma cobertura que, embora abrangente, nem sempre é suficiente para capturar todos os casos de erro. Entre esses casos incluem-se os *edge cases* (casos extremos), que correspondem a valores nos limites válidos de entrada, e os *corner cases*, que representam combinações simultâneas de casos extremos e particularmente difíceis de prever manualmente. Para mitigar esse problema, recorre-se a práticas de *shift-left testing*, que permitem antecipar atividades de teste para fases iniciais do desenvolvimento. Deste modo, do QuickCheck para Haskell, surgiu o PBT estendido a várias linguagens, incluindo Python, através da biblioteca Hypothesis [3, 4]. Esta técnica tem por base a definição de propriedades gerais que o sistema deve satisfazer, em vez de

exemplos específicos de entradas e saídas. Assim, um sistema pode ser testado automaticamente com uma vasta gama de entradas, incluindo situações limite e casos inesperados, aumentando a cobertura do teste de forma exponencial. Diversos estudos demonstraram que o PBT é eficaz na deteção de falhas que os testes unitários não conseguem identificar, beneficiando especialmente sistemas complexos com múltiplas condições de entrada [7]. Além de aumentar a cobertura da análise, o PBT reduz o tempo necessário para criar casos de teste, uma vez que estes são gerados automaticamente, permitindo validar de forma mais abrangente o comportamento do sistema.

A integração dos testes unitários e PBT apresenta vantagens evidentes em termos de robustez do software e eficiência do ciclo de desenvolvimento. Estudos recentes indicam que a combinação das duas abordagens aumenta significativamente a fiabilidade do software, reduz falhas críticas em fases iniciais e diminui o custo de manutenção ao longo do ciclo de vida do projeto [7, 8].

Neste sentido, foram selecionados testes realizados a partir da biblioteca pymoney e da *Embedded Test Automation Framework* (ETAF) da Capgemini Engineering (doravante designada por framework ETAF) sobre as quais se procedeu à análise de cobertura do código existente no sentido de identificar os tipos de testes e as lacunas existentes nos testes já realizados. Após essa análise foram criados novos testes unitários e introduzidos PBT para avaliar vários cenários. Por fim procedeu-se à comparação e análise dos resultados obtidos, avaliando as vantagens e limitações de cada abordagem. Esta metodologia permitiu avaliar a robustez do código e compreender em que situações é que cada uma das técnicas de teste se revela mais eficaz.

3 Ferramentas e Metodologias

Esta secção apresenta as principais ferramentas e metodologias utilizadas no desenvolvimento do trabalho de projeto.

3.1 Ferramentas Utilizadas

Para garantir um ambiente robusto e eficiente para o desenvolvimento e teste, foram selecionadas várias ferramentas alinhadas com os objetivos do projeto.

A linguagem Python 3.13 [9] foi adotada devido à sua sintaxe clara e moderna, além da ampla gama de bibliotecas disponíveis que facilitaram a implementação de testes unitários e baseados em propriedades. A sua flexibilidade e facilidade de integração com frameworks como Hypothesis e Pytest tornaram-na ideal para prototipagem rápida e execução eficiente de testes automatizados.

Para o controlo de versões e gestão do desenvolvimento colaborativo, utilizou-se o Git [10], que permitiu acompanhar o histórico de alterações, facilitar o trabalho em equipa e garantir a integridade do código durante o ciclo de vida do projeto.

O Visual Studio Code [11] foi escolhido como ambiente de desenvolvimento integrado, provendo uma interface intuitiva e um conjunto de extensões dedicadas ao Python e ferramentas de teste.

Em conjunto, essas ferramentas permitem obter uma infraestrutura tecnológica sólida para a implementação de uma metodologia de teste eficaz e integrada, desde a codificação até à análise de resultados. As fases da metodologia adotadas para aplicar e avaliar testes unitários e testes PBT foram organizadas por etapas sequenciais. O fluxograma da Fig. 1 ilustra de forma esquemática a metodologia seguida, destacando as principais etapas desde a fase inicial de aprendizagem até à avaliação final dos métodos de teste. Além disso, evidencia a integração entre os testes unitários e os testes PBT, bem como a sua aplicação prática em casos reais que sustentam as análises apresentadas.

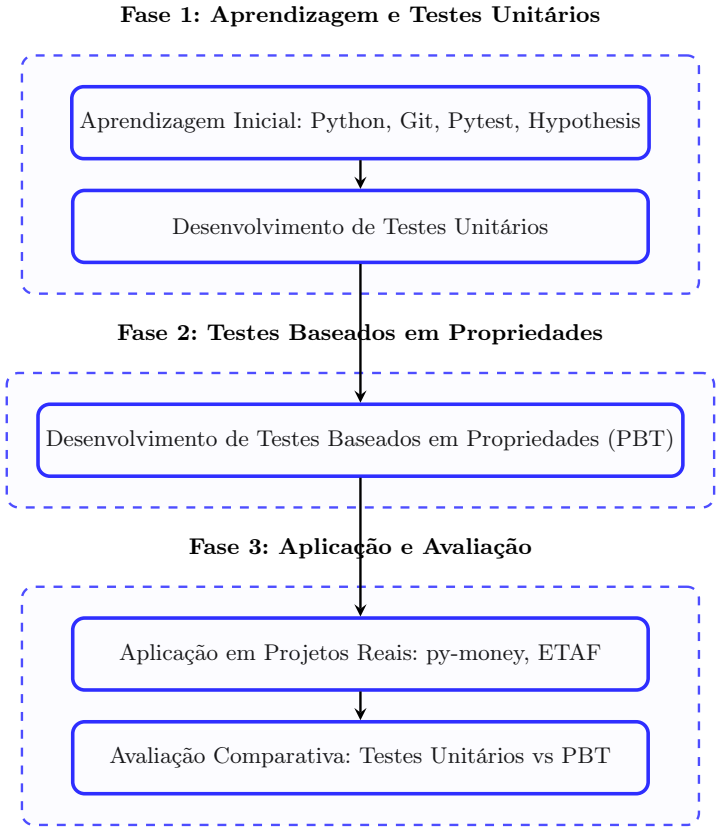


Fig. 1. Fluxograma esquemático da metodologia implementada.

3.2 Aprendizagem de Testes de Software

A primeira etapa do projeto consistiu na familiarização com as ferramentas usadas, em síntese, com a linguagem Python, o sistema de controlo de versões Git, a framework Pytest para testes unitários e a biblioteca Hypothesis para PBT [3]. Seguidamente, iniciou-se a implementação dos testes unitários recorrendo ao Pytest, que fornece uma interface simples e permite verificar o comportamento de funções e métodos individuais [2]. A Listagem 1 exemplifica este processo a partir de um teste que valida o lançamento de uma exceção numa divisão por zero.

Listagem. 1: Teste de divisão em divisão por zero.

```
def test_divisao_por_zero():
    with pytest.raises(ZeroDivisionError):
        divisao(10, 0)
```

Já para implementação de PBT, usou-se a biblioteca Hypothesis que permitiu definir as propriedades gerais do sistema e testar automaticamente uma ampla gama de entradas [3,4]. A Listagem 2 apresenta um exemplo baseado no uso da propriedade de idempotência numa função de ordenação de uma lista.

Listagem. 2: Teste de idempotência numa ordenação a partir da biblioteca Hypothesis.

```
from hypothesis import given
import hypothesis.strategies as st

@given(st.lists(st.integers()))
def test_idempotencia_da_ordenacao(lista):
    assert sorted(sorted(lista)) == sorted(lista)
```

A propriedade de idempotência garante que aplicar uma operação múltiplas vezes produz o mesmo resultado que aplicá-la uma única vez. No caso da ordenação, `sorted(sorted(lista))` deve sempre igualar `sorted(lista)`. Este teste valida a propriedade matemática de idempotência da função `sorted()`. O decorador `@given` instrui o Hypothesis a gerar automaticamente múltiplas listas de inteiros (incluindo listas vazias, com elementos repetidos, negativos, etc.). A propriedade testada estabelece que ordenar uma lista já ordenada não altera o resultado, garantindo que a função é estável e determinística.

Esta abordagem substitui potencialmente centenas de testes unitários específicos por uma única propriedade universal, demonstrando a eficiência do PBT.

Também foram criadas estratégias compostas para entradas complexas, necessárias para validar integrações na framework ETAF [5] usada na empresa.

3.3 Aplicação e Avaliação

Os métodos desenvolvidos foram aplicados em dois contextos complementares, selecionados por razões distintas mas convergentes. A biblioteca open-source py-money [6] foi escolhida por possuir um domínio bem definido, operar sobre valores monetários com múltiplas moedas e apresentar complexidade suficiente para explorar limites numéricos de forma controlada a partir de ambos os testes. Já a framework ETAF, enquanto solução interna da empresa, representa um sistema de automação com múltiplas integrações e casos de uso em contexto empresarial, permitindo avaliar em particular o impacto dos PBT em cenários próximos da utilização em produção.

A avaliação comparativa considerou métricas de tempo de execução, cobertura de casos e tipos de defeitos encontrados. Verificou-se que, apesar dos testes unitários serem eficazes para cenários previstos, os testes baseados em propriedades melhoram a capacidade de capturar erros complexos e *edge cases* [7, 8].

A formulação das propriedades apresentou uma curva longa de aprendizagem inicial, mas revelou-se eficiente em conjunto com as abordagens tradicionais.

A integração de testes unitários e PBT foi concebida desde o início deste trabalho com um papel complementar: os primeiros focados em casos concretos e conhecidos, os segundos orientados para a exploração sistemática de espaços de entrada mais amplos. Esta abordagem foi aplicada em algumas funções desenvolvidas, na biblioteca py-money e na framework ETAF, estabelecendo a base para a análise apresentada na seção seguinte.

4 Implementação e Resultados

A aplicação da metodologia mista de testes unitários e testes PBT mostrou ganhos claros e mensuráveis em vários aspetos da qualidade do software.

4.1 Cobertura e Deteção de Falhas

A Listagem 3 mostra um teste PBT, que gera automaticamente diversos dados de entrada para validar se o método `fromsubunits` da classe `Money` da bibioteca py-money produz o resultado esperado. A estratégia `itemsstrategy` cria valores realistas para moedas, montantes e precisões, que aumenta a cobertura e confere robustez ao teste ao incluir casos extremos e *edge cases* que testes manuais poderiam não contemplar.

Listagem. 3: Teste de propriedade para o método `Money.fromsubunits`.

```
@given(itemsstrategy)
def testfromsubunits(data):
    currency, amount1, subunit, precision, data = data
    money = Money.fromsubunits(amount1, currency)
    expected = Money(f"{amount1}.{subunit * precision}", currency)
    assert money == expected
```

A integração entre testes unitários e PBT na biblioteca foi implementada de forma explícita e complementar. Os testes unitários validam casos específicos e conhecidos (por exemplo, USD e JPY), enquanto os PBT analisam o espaço de entrada para todas as moedas suportadas, usando combinações que não tinham sido consideradas pelos testes unitários - exemplo na Listagem 4.

Listagem. 4: Integração de teste unitário vs PBT na biblioteca py-money.

```
def test_from_sub_units(self):
    money = Money.from_sub_units(101, Currency.USD)
    assert money == Money('1.01', Currency.USD)

    money = Money.from_sub_units(5, Currency.JPY)
    assert money == Money('5', Currency.JPY)

@given(items_strategy())
def test_from_sub_units(data):
    currency, amount1, sub_unit, precision, *_ = data
    money = Money.from_sub_units(amount1, currency)
    expected = Money(f"{amount1 / sub_unit:.{precision}f}", currency)
    assert money == expected
```

O PBT testou todas as moedas da biblioteca e identificou o erro relacionado com o PYG, cujo resultado `default_fraction_digits=0` identifica uma moeda sem casas decimais - um caso não coberto pelos testes unitários.

A Tabela 1 sintetiza de forma qualitativa as diferenças entre as duas abordagens realizadas na biblioteca py-money. Esta avaliação não recorre a métricas formais de cobertura de código, mas à perceção prática de vários cenários.

Tabela 1: Comparação qualitativa entre testes unitários e PBT.

Critério	**Testes unitários**	**PBT**
Cobertura de cenários(*)	Limitada a casos definidos manualmente	Alta, abrangendo combinações não antecipadas
Deteção de falhas	Boa em casos conhecidos	Muito boa em casos extremos e combinações raras
Esforço de criação	Baixo a médio por teste simples	Inicialmente mais elevado (definição de propriedades e estratégias)
Tempo de execução	Geralmente rápido	Pode ser superior devido ao número de casos gerados
Complexidade conceptual	Baixa	Maior, exige reflexão sobre propriedades e domínio

*Cobertura de cenários de teste (qualitativa), distinta de cobertura de código (%) medida com coverage.py.

4.2 Eficiência na Detecção de Erros

Na Listagem 5 apresenta-se um teste unitário em que o método `testfromsubunits` verifica se a conversão de um valor inteiro expresso em subunidades de dólar para o seu valor monetário decimal (com duas casas decimais) está correto.

Listagem. 5: Teste unitário do método `testfromsubunits`.

```
1 def testfromsubunits():
2     money = Money.fromsubunits(123456, "USD")
3     assert money == Money("1234.56", "USD")
```

Nas Figuras 2 e 3, observa-se o impacto do uso de testes PBT em conjunto com os testes unitários tradicionais.

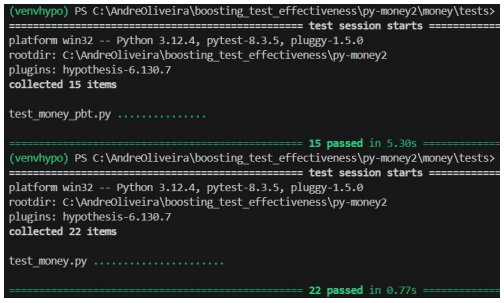


Fig. 2. Tempo para detecção de erros críticos usado testes PBT e unitários.

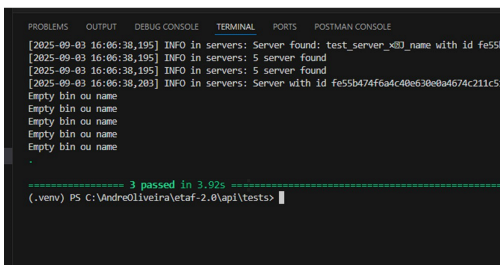


Fig. 3. Distribuição dos tipos de testes e cobertura obtida.

A Figura 2 ilustra a redução no tempo médio necessário para detetar erros críticos ao longo do ciclo de desenvolvimento, destacando a maior eficiência dos testes PBT na identificação antecipada de falhas. A Figura 3 complementa essa análise mostrando a maior cobertura de cenários testados com o uso de PBT, especialmente na captura de *corner cases*, o que reforça a robustez e a eficácia da abordagem proposta.

4.3 Redução de Esforço e Produtividade

A Listagem 6 é um exemplo simplificado que representa um teste onde múltiplas condições são interligadas e testadas em conjunto. Essa abordagem é útil para garantir que o sistema funcione corretamente quando várias restrições e condições coexistem, o que é comum na validação de sistemas complexos e serve como complemento aos testes unitários focados em partes isoladas do código.

Listagem. 6: Código de exemplo para condições combinadas de teste.

```
def test_combined_conditions():
    data = generate_complex_conditions()
    result = run_test_with_conditions(data)
    assert result
```

Além dos benefícios quantitativos em cobertura e deteção, a metodologia indicou a redução do esforço manual para criação e manutenção dos testes, como mostra a tabela 2. Isto traduz-se em maior produtividade da equipa e menor propensão a erros humanos na elaboração de casos de teste.

Tabela 2: Avaliação do Esforço e Produtividade

Aspecto Avaliado	**Testes Unitários**	**PBT**
Tempo médio para desenvolvimento	Médio	Médio
Frequência de atualização dos testes	Alta	Média
Necessidade de revisão manual	Elevada	Reduzida

4.4 Cobertura de Casos Críticos e *Corner Cases*

A metodologia de testes PBT demonstrou ser particularmente eficaz na identificação de *corner cases* associados a cenários específicos e extremos, que frequentemente permanecem ocultos em estratégias tradicionais de testes unitários.

Experimentalmente, ao aplicar a metodologia na biblioteca py-money, foi identificada uma falha relacionada com a manipulação incorreta de moedas com diferentes números de casas decimais, gerando erros em operações matemáticas com valores monetários. Esta falha não foi detectada pelos testes unitários convencionais, sendo revelada apenas após a geração automática de casos de teste PBT variados pelo Hypothesis. A Figura 4 ilustra um erro crítico descoberto exclusivamente pelos testes PBT, onde se evidencia o facto de a moeda PYG que tem 100 subunidades, gerar o erro `default_fraction_digits=0`, causando falha na conversão de sub-unidades; esta ocorrência evidencia a utilidade prática da geração de dados aleatórios e de propriedades invariantes no código testado.

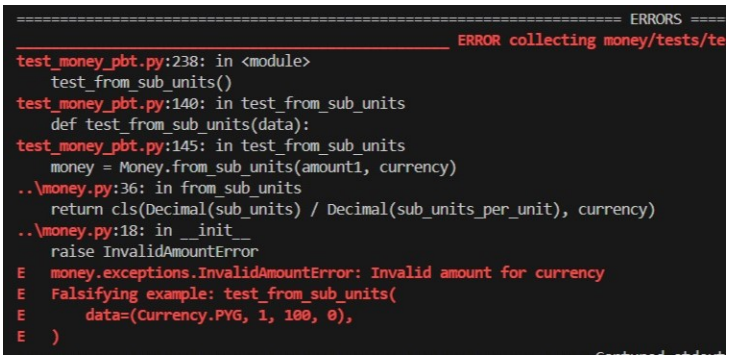


Fig. 4. Erro detectado usando PBT na biblioteca py-money, ilustrando a importância dos *corner cases* para garantir a robustez do sistema.

Além disso, na framework ETAF, os PBT permitiram testar conjuntos amplos e incomuns de entradas, como strings vazias para nomes de servidores, valores nulos ou formatos inválidos para identificadores, ampliando substancialmente a cobertura dos testes e contribuindo para a robustez geral do sistema.

A estratégia de geração automática de dados, utilizando técnicas como a `uuid4strategy` para valores realistas combinada com geradores de strings abertas, ampliou o espectro de entrada, explorando casos que dificilmente seriam concebidos manualmente.

Em síntese, esta capacidade dos testes PBT para explorar e validar *corner cases* críticos reforça a confiabilidade e a resiliência do software, reduzindo o risco de falhas em ambientes reais e justificando a integração dessa metodologia nos processos de qualidade e desenvolvimento. Os dados qualitativos permitem percecionar o ganho substancial com a metodologia integrada de testes unitários e PBT, corroborando as premissas deste trabalho e justificando a adoção cumulativa dessas abordagens para maximizar a qualidade e eficiência no desenvolvimento de software.

Os resultados obtidos confirmam a hipótese inicial de complementaridade entre testes unitários e PBT, com os últimos detetando falhas críticas (ex.: PYG em py-money, Fig. 4) não capturadas pelos primeiros, alinhando-se com [7] sobre detecção de *corner cases* em bibliotecas financeiras. A Tabela 1 e Fig. 2 qualificam ganhos em cobertura de cenários (alta vs. limitada) e redução de tempo de deteção, enquanto a Tabela 2 evidencia menor esforço de manutenção a longo prazo, apesar da curva longa inicial de aprendizagem. Esta abordagem mista reduz riscos em produção (ETAF) e preenche lacunas em testes manuais para domínios numéricos complexos; as limitações incluem dependência de estratégias Hypothesis bem definidas e escalabilidade em sistemas [8].

5 Conclusões

O presente trabalho demonstrou que a integração de PBT em conjunto com testes unitários tradicionais representa um avanço significativo na qualidade,

robustez e eficiência do desenvolvimento de software. A metodologia proposta permitiu explorar um espectro mais amplo de cenários, especialmente *corner cases*, que são frequentemente negligenciados em abordagens convencionais.

Os resultados obtidos evidenciam que o uso precoce e sistemático de PBT conduz a uma deteção mais rápida e eficaz de falhas, reduzindo o tempo total destinado à elaboração e manutenção dos testes. Essa abordagem automatizada e orientada a propriedades contribuiu para aumentar a confiança na entrega de software mais resiliente e confiável, diminuindo o risco de erros críticos em produção.

Neste trabalho também se destacou desafios inerentes ao PBT, como a necessidade de um entendimento aprofundado acerca do domínio da aplicação para definir propriedades relevantes, bem como o esforço inicial para formular essas propriedades e configurar os geradores de dados de teste adequadamente.

Como trabalho futuro, propõem-se três direções principais: o desenvolvimento de mecanismos para integração automatizada entre PBT e outras estratégias de verificação; a criação de ferramentas para geração e manutenção de propriedades complexas; a expansão da metodologia a outros domínios e sistemas para validação em maior escala.

Estas extensões poderão consolidar o valor da abordagem e promover uma cultura de testes mais rigorosa no desenvolvimento de software.

Referências

1. IEEE Computer Society: IEEE Standard Glossary of Software Engineering Terminology (IEEE Std 610.12-1990). IEEE, New York, 1990.
2. A. Spillner, T. Linz, and H. Schaefer, *Software Testing Foundations: A Study Guide for the Certified Tester Exam*, 4th ed. Rocky Nook, 2014.
3. Hypothesis Development Team, "Hypothesis: Property-based Testing for Python," 2025. [Online]. Disponível: https://hypothesis.works
4. K. Claessen and J. Hughes, "QuickCheck: A Lightweight Tool for Random Testing of Haskell Programs," in *Proceedings of the International Conference on Functional Programming*, 2000, pp. 268–279.
5. Equipe Interna, "Embedded Test Automation Framework - ETAF," Documentação interna da empresa, 2025.
6. Vimeo, "Py-money Python library for handling monetary values," Disponível em: https://github.com/vimeo/py-money, 2025.
7. J. Foster et al., "Empirical Evaluation of Property-Based Testing," *Software Testing Journal*, vol. 15, no. 3, pp. 234–249, 2023.
8. B. Meyer, "Combining Property-Based Testing with Unit Testing," em *Proceedings of the Software Quality Conference*, 2018, pp. 55–63.
9. Python Software Foundation, "What's new in Python 3.13," 2025.
10. GitLab Inc., "What is a distributed version control system?," 2025.
11. Microsoft Corporation, "Visual Studio Code," 2025.

Session 5A

"Soluções Avançadas para Processos Empresariais e Infraestruturas"

Chair: Rui Marques **Room:** H202 **Time:** 15:15 - 16:15

Paper	Presenter
Simplifying Complex Insurance Product Management with AI	João Teixeira
Automated Extraction of Insurance Product Characteristics	Francisco Oliveira
Metodologias Ágeis e Ensino Invertido na Engenharia: Um Caso de Aplicação Real	Miguel Correia
NetBox-Zabbix Plugin para Gestão e Organização de Data Center	Ivo Pereira

Simplifying Complex Insurance Product Management with AI

João Teixeira[1][1201399], Paulo Gandra de Sousa[1], Luiz Faria[1], Duarte Cardoso[1], and Francisco Oliveira[1][1201545]

Instituto Superior de Engenharia do Porto mail@isep.ipp.pt
https://www.isep.ipp.pt/

Abstract. The digital transformation of the insurance sector presents significant challenges in internal processes, particularly in the management of product information. These challenges arise from the high complexity and variability of insurance product models, which are frequently updated, and from the technical demands of existing tools that require extensive user expertise to operate effectively. Recent advances in Generative Artificial Intelligence (GenAI) and the growing use of Large Language Models (LLMs) and intelligent agents are opening up new opportunities to automate and streamline processes, enabling organizations to adapt to a rapidly evolving technological landscape.
The Product Machine Explorer leverages Generative AI, Large Language Models (LLMs), and AI agents to make exploring insurance product models easier. Integrated with msg Life Iberia's Product Machine platform, it allows users to interact with complex product data using natural language. By leveraging structured data and advanced query techniques, it can understand user requests and deliver accurate, context-aware responses, improving both efficiency and usability in product model exploration.
Built using the Evaluation-Driven Development (EDD) methodology and supported by software, prompt, and context engineering, the tool was assessed using defined metrics and expert feedback. The results demonstrate significant efficiency improvements, with professionals spending considerably less time on repetitive tasks. Overall, the Product Machine Explorer demonstrates how LLM-powered agents can simplify complex information management and support better decision-making in the insurance sector.

Keywords: Insurance Products · Model Querying · Generative AI · Large Language Models · AI Agents · Agentic Workflows

1 Introduction

1.1 Context

Over the past years, the insurance industry has gone through a major digital shift that has changed how companies handle information, develop products, and

interact with customers. This shift has brought new challenges, particularly in product management, where the complexity arises from multiple variations, interrelations, and specific rules that must be continually adapted to match market demands. Even though tools exist to centralize information on product structures, detailed configuration and exploration of each product are still performed manually, leading to time-consuming, unintuitive, and operationally inefficient processes.

1.2 Problem

The growing complexity of insurance products and the increasing demand for faster data retrieval remain major challenges for companies striving to maintain competitiveness. Although platforms such as Product Machine - one of msg Life Iberia's products - centralize insurance product information, internal inefficiencies persist due to the unintuitive model navigation process. This complexity requires significant user expertise in the platform's internal operations, ultimately reducing productivity and increasing time spent on repetitive tasks. Furthermore, product models often embed intricate business rules and exhibit significant heterogeneity across different clients, further amplifying the difficulty of managing and evolving these systems.

As a result, there's a growing demand for easier and more efficient ways to access, search, and manage insurance product data.

1.3 Objective and Methodology

Generative Artificial Intelligence (AI) offers significant potential to address these challenges by enabling advanced natural language understanding, translating user intent into structured data operations, and generating coherent natural language responses, thereby bridging the gap between human communication and structured data interaction. Therefore, the main goal of this project is the design, development, and implementation of an Artificial Intelligence agent capable of performing natural language queries on insurance product models within the Product Machine. Additionally, the project aims to integrate this agent into the main Product Machine platform, enabling its use in a real operational context and ensuring the practical relevance of the proposed solution.

To achieve this goal, the Evaluation-Driven Development (EDD) methodology is adopted. This approach relies on the systematic definition and application of evaluations (evals) to objectively measure the quality and consistency of the system's responses against a predefined set of reference results - the gold standard [1]. EDD shares similarities with Test-Driven Development (TDD) in traditional software engineering but distinguishes itself through its more dynamic nature and specific suitability for AI-based systems, ensuring continuous improvement and performance-driven evolution of the agent [1].

2 Related Work

Recent advances in Generative Artificial Intelligence (AI) and Large Language Models (LLMs) have significantly impacted natural language processing and structured data interaction. Transformer-based models, such as GPT-4, demonstrate strong capabilities in contextual understanding, reasoning, and text generation [3]. However, their application to structured data querying, particularly in specialized domains like insurance, requires careful prompt design, data representation, and management of domain-specific business rules [3].

Research in Text-to-SQL has shown that LLMs can translate natural language into executable queries, yet challenges remain in handling complex schemas, ensuring query correctness, and managing multi-step reasoning for large or highly interrelated datasets [2]. Parallel developments in multi-agent LLM architectures, using frameworks such as LangGraph, enable collaborative, stateful workflows that integrate reasoning, tool use, and stepwise query execution [2]. However, recent evaluations reveal a gap between benchmark performance and real-world applicability. While LLMs achieve high accuracy on simple SQL tasks, their performance drops sharply on complex analytical queries, often producing semantic inconsistencies and structural errors [6]. Studies also indicate that even high-performing models may struggle when transitioning from generic benchmarks to enterprise environments, partly due to misalignment or oversimplification in widely used datasets [7]. These findings underscore the difficulty of applying LLMs to enterprise-grade schemas and highlight the need for domain-aware reasoning and structured workflows.

In the context of insurance product modeling, data representation is critical. Models are typically stored in relational databases, which remain dominant in enterprise environments due to their maturity, integration capabilities, and practical applicability. While the use of Knowledge Graphs (KGs) have been proposed as an alternative for representing complex relationships and supporting semantic reasoning [4], most practical implementations rely on SQL as the core data structure for LLM-assisted querying and knowledge exploration, given the SQL-centric infrastructures predominant in enterprise environments.

3 Product Machine

The Product Machine (PM), developed by msg Life Iberia, is a modeling platform that centralizes insurance product information - including coverages, rules, and calculations - and supports integration with downstream systems. It simplifies product modeling, validation, calculations, and reporting, helping teams develop products more efficiently. Built on the Eclipse Modeling Framework (EMF) and based on Model-Driven Engineering principles, PM uses technologies like the Object Constraint Language (OCL) and Query/View/Transformation (QVT) to support structured modeling, enforce rules, and perform model-to-model transformations.

The core strength of Product Machine is its flexibility, allowing each client to create a **custom metamodel** and generate unique product structures tailored

to their unique needs. Adding to this versatility, **component relationships within a product structure are dynamic**, adapting to specific conditions and contexts. This complexity is managed through Composition Rules (CRs) and Composition Rule Variations (CRVs). CRs define parent-child relationships, including cardinalities and dependencies, while CRVs allow these relationships to vary according to context, such as policy type, geographic region, or customer attributes. Availability formulas and functional states further regulate component behavior and visibility, enabling precise, rule-driven product configurations. Such flexibility, however, poses significant challenges for consistent querying and exploration of client-specific models.

Typical components that integrate client models include:

- **Products:** Primary insurance offerings that define the structure and rules for a specific type of insurance.
- **Packages:** Sets of coverages and entities grouped within a product to facilitate management, typically associated with the insurance risk.
- **Coverages:** Specific types of protection provided by an insurance product, detailing what is covered.
- **Participants:** Entities associated with the contract. For example, a person or organization can be the policyholder - responsible for payment and management - while another may be the insured party.
- **Pricing/Premium Details:** Contain the calculations and logic required to determine premiums, representing the detailed values computed during the process.

Overall, the Product Machine provides a powerful and highly configurable framework for insurance product modeling, but its flexibility requires careful handling when querying and analyzing client-specific configurations.

4 Solution

4.1 Functional Requirements

Defining the system's functional requirements is essential to establish a foundation for evaluating its performance across the key elements of a typical insurance product model. They outline the key information the agent must accurately retrieve and interpret. There are seven primary functional requirements (**RF**), each addressing a specific aspect of information gathering and interpretation.

- **RF1:** Query information about insurance products and product families
- **RF2:** Query information about packages within products
- **RF3:** Query information about coverages and their associations
- **RF4:** Query information about premium components and details
- **RF5:** Query information about insurable objects
- **RF6:** Query information about composition rule variations (CRVs)
- **RF7:** Query information about premium calculation formulas

Together, these requirements ensure comprehensive coverage of the core entities and relationships within insurance product models, forming the basis for evaluating the agent's accuracy and effectiveness.

4.2 System Design

The Product Machine Explorer was designed with a modular and extensible architecture. It integrates multiple components that work together to manage agent workflows, user interaction, and data access. The system comprises four main elements:

1. Backend agent orchestration using the LangGraph framework.
2. Frontend interface developed with Streamlit for rapid prototyping and user interaction.
3. MariaDB database that stores structured product information.
4. API layer exposing defined endpoints, through which Product Machine version five (PM5) can invoke dedicated agent workflows.

Fig. 1 illustrates the overall system architecture.

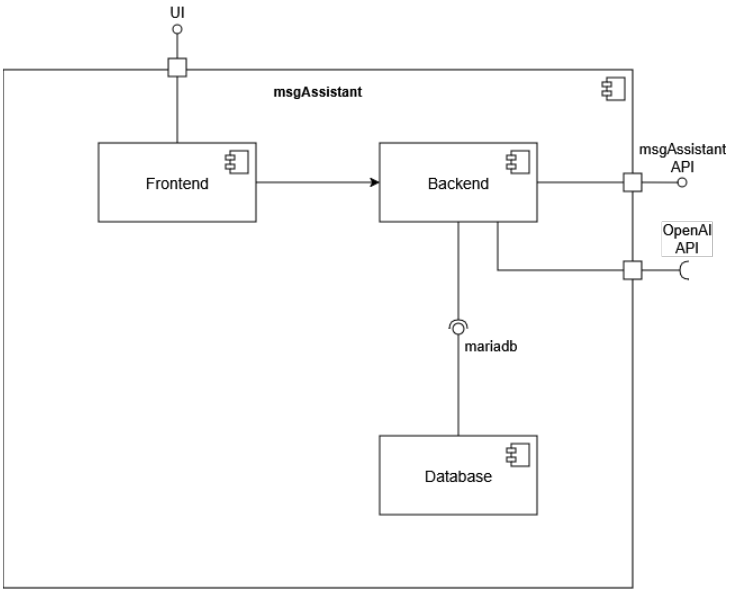


Fig. 1. System Design

This modular design ensures flexibility, ease of maintenance, and seamless coordination across all system components.

4.3 Data Structure Preparation

An essential step in the system's design was restructuring Product Machine's complex data model to enable efficient LLM-driven querying. This process involved:

- **View Creation:** Database views were implemented to encapsulate complex multi-table JOIN operations (often involving more than five tables) and to present unified logical groupings of data of component relationships.
- **Schema Simplification:** Column names in the view schemas were standardized and simplified to enhance LLM comprehension.
- **Composition Rule Variations (CRVs):** Special handling was implemented for CRVs, which represent conditional relationships between product components. Views were enriched to include availability formulas that determine when relationships apply.
- **Formula Extraction:** Premium calculation formulas were extracted from runtime-generated files and structured into dedicated tables, enabling queries about pricing logic.

4.4 Agent Workflow

The agent employs a directed acyclic graph (DAG) workflow consisting of nodes with edges that determine its execution flow, as illustrated in Fig. 2.

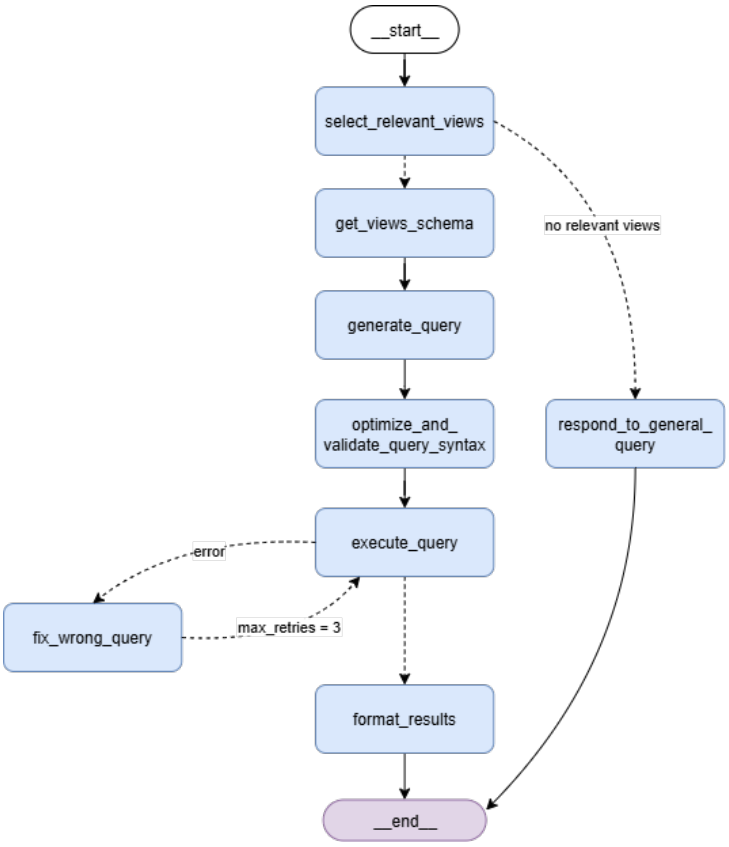


Fig. 2. Agent Workflow Diagram

This workflow consists of the following main components:

1. **View Selection:** Given a user query, the agent identifies relevant database views containing information needed to answer the question. This step employs few-shot prompting with examples of query-to-view mappings.
2. **View Schema Retrieval:** Extracts schema details for the selected views from the database to provide structural context for query construction.
3. **Query Generation:** The agent generates a SQL query based on the selected views and user intent. This step incorporates schema information, hierarchical relationship rules, and domain-specific constraints.
4. **Query Validation and Optimization:** The generated query is validated for syntax correctness and optimized for performance. Common SQL errors are identified and corrected, including improper clauses, missing JOIN conditions, and inefficient subqueries.
5. **Query Execution:** The validated query is run against the database with guard-rails in place to prevent any unauthorized operations. Only SELECT operations are allowed.
6. **Wrong Query Fix:** If a query fails, the system analyzes the error and tries to automatically fix it, allowing up to three retries.
7. **Result Formatting:** The query results are turned into clear, natural-language responses that directly answer the user's question, providing the right context and structure.

Additionally, a conditional path lets the agent handle user questions that aren't related to database operations, such as follow-ups or general questions about the Product Machine.

In short, this workflow creates a clear and reliable pipeline that turns natural language queries into accurate, optimized, and context-aware database responses, thus ensuring both operational reliability and real business value.

4.5 Prompt Engineering

Effective prompt engineering proved critical for system performance. Key techniques include:

- **Structured Instructions:** Prompts are organized hierarchically using Markdown formatting with clear sections for context, guidelines, and examples. In general, modern LLMs respond particularly well to well-structured prompts with specific sections and hints to direct the LLM [5].
- **Few-Shot Learning:** Critical nodes include examples that illustrate the correct query patterns for complex situations, especially when dealing with CRVs or navigating hierarchical structures.
- **Ordered Execution Steps:** For tasks that involve multiple steps, using ordered lists helps the model carry out each step in the correct sequence [5].
- **Domain Context:** Prompts provide detailed descriptions of insurance product structures, hierarchical relationships, and business rules. The provided context includes key terminology and conceptual frameworks from the insurance field to ensure accurate understanding and minimize ambiguity.

In addition to these techniques, a strategy was put in place to manage **multiple client-specific insurance models** that may vary in structure. Configuration dictionaries were introduced to capture instructions unique to each client that can't be applied universally. These dictionaries include database view mappings, component hierarchies, domain-specific terminology, and example queries, ensuring prompts are accurately adapted to each client's data.

5 Results

By adopting an Evaluation-Driven Development methodology, the system's iterative implementation was guided by the early definition of evaluation metrics designed to assess the agent's performance across diverse test cases. Two primary metrics form the basis of this evaluation:

- **M1 - Query Accuracy:** Average string similarity between generated and expected results must exceed 0.90 across all test cases.
- **M2 - Consistency:** Standard deviation of results across multiple executions must remain below 0.10, ensuring reliable performance.

These metrics ensure that development remains closely aligned with measurable performance goals throughout the system's lifecycle, promoting both accuracy and stability.

5.1 Quantitative Evaluation

Table 1 presents evaluation results for the final system iteration, demonstrating achievement of both defined metrics across all functional requirements.

Table 1. Agent Evaluation Results

Functional Requirement	Mean	Std Dev
RF1: Products	0.935	0.000
RF2: Packages	0.945	0.000
RF3: Coverages	0.960	0.000
RF4: Premium Components	0.965	0.000
RF5: Insurable Objects	0.985	0.000
RF6: Variations (CRVs)	0.958	0.015
RF7: Calculation Formulas	0.922	0.004
Overall Average	**0.953**	**0.003**

The system successfully exceeds the defined threshold (M1: 0.90) with an overall average accuracy of 0.953. Consistency metrics (M2) remain well below the 0.10 threshold, with most requirements showing zero deviation across five execution runs.

5.2 Qualitative Evaluation

Beyond quantitative metrics, the system was successfully integrated into the PM5 production environment and subsequently validated by insurance product specialists. Fig. 3 shows the newly added component.

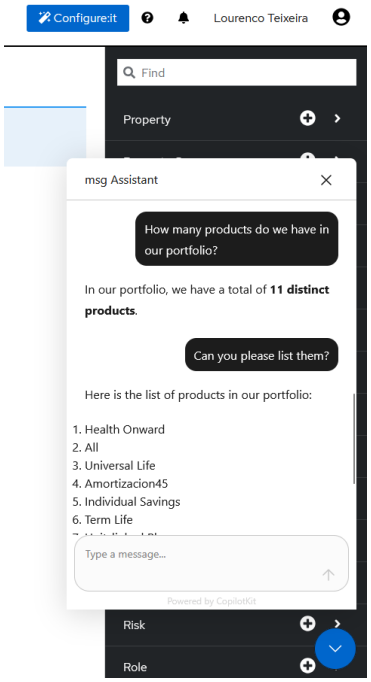


Fig. 3. Integration within PM5's interface

Technical users reported substantial efficiency improvements, as complex queries that previously required extensive manual analysis can now be executed in seconds. Moreover, it streamlines the retrieval of valuable product insights, demonstrating the system's practical value and robustness.

6 Conclusions

6.1 Key Findings

This work demonstrates that an SQL-based approach, supported by targeted knowledge engineering, is highly effective for querying insurance product models, leveraging SQL's proven maturity and enterprise-grade integration. Careful data structure design, through simplified naming, logical view grouping, and explicit business rule representation, significantly enhances system performance and query reliability. By combining language models of varying sizes, teams

can save costs without compromising performance, with lighter models handling simple tasks and more powerful ones tackling complex problems.

Finally, an evaluation-driven development process, grounded in continuous metric-based assessment, proved essential for iterative refinement and production-level reliability.

6.2 Limitations and Future Work

While the system demonstrates strong performance, several areas remain open for improvement. Currently, only the core product components within insurance product models are supported, and achieving full coverage will require refining context engineering and introducing purpose-built views.

Performance can also be enhanced, as complex queries with large results may exhibit latency. Future work could explore the use of query decomposition and result aggregation to enhance responsiveness.

Finally, evaluation methods rely on string-similarity metrics, which can unfairly penalize responses due to minor formatting differences and often require manual inspection of the results. Adopting LLM-as-a-Judge approaches could enable more reliable semantic evaluation of response quality, allowing assessments to capture correctness beyond superficial formatting differences and reducing the need for manual inspection [8].

References

1. Xia, B., Lu, Q., Zhu, L., Xing, Z., Zhao, D., Zhang, H.: Evaluation-Driven Development of LLM Agents: A Process Model and Reference Architecture. arXiv:2411.13768 [cs.SE] (2025). https://arxiv.org/abs/2411.13768
2. Zhu, X., Li, Q., Cui, L., Liu, Y.: Large Language Model Enhanced Text-to-SQL Generation: A Survey. arXiv:2410.06011 [cs.DB] (2024). https://arxiv.org/abs/2410.06011
3. Wang, B., Ren, C., Yang, J., Liang, X., Bai, J., Chai, L., Yan, Z., Zhang, Q.-W., Yin, D., Sun, X., Li, Z.: MAC-SQL: A Multi-Agent Collaborative Framework for Text-to-SQL. arXiv:2312.11242 [cs.CL] (2025). https://arxiv.org/abs/2312.11242
4. Huang, X., Ku, C.: Designing an Interpretable Question Answering System for Vertical Domains Based on Large Language Model and Knowledge Graph. In: Book Title or Collection Name (if applicable), pp. xx–xx. Publisher (2024). https://doi.org/10.3233/ATDE240503
5. OpenAI: GPT-4.1 Prompting Guide. OpenAI Cookbook (2024). https://cookbook.openai.com/examples/gpt4-1_prompting_guide
6. Choi, J.: Fact-Consistency Evaluation of Text-to-SQL Generation for Business Intelligence Using Exaone 3.5. arXiv:2505.00060 [cs.CL] (2025). https://arxiv.org/abs/2505.00060
7. Zhang, F., Zhou, R.: Refining Zero-Shot Text-to-SQL Benchmarks via Prompt Strategies with Large Language Models. Appl. Sci. 15(10), 5306 (2025). https://doi.org/10.3390/app15105306
8. Zeng, Q., Ma, S., Niknafs, A., Basran, A., Szabo, C.: Taming SQL Complexity: LLM-Based Equivalence Evaluation for Text-to-SQL. arXiv:2506.09359 [cs.DB] (2025). https://arxiv.org/abs/2506.09359

Automated Extraction of Insurance Product Characteristics

Francisco Oliveira[1 [0009−0006−3125−673X]], Paulo Gandra de Sousa[1,2 [0000−0001−9354−6608]], Luiz Faria[1 [0000−0002−1169−7785]], Duarte Cardoso[2 [0009−0005−9370−0085]], and João Teixeira[1 [0009−0005−5508−8446]]

[1] Instituto Superior de Engenharia do Porto 1201545@isep.ipp.pt
[2] msg Life Iberia

Abstract. The increasing complexity and diversity of insurance products have highlighted the need for efficient methods to interpret and manage the detailed information present in regulatory documents. This project explores the application of Natural Language Processing (NLP) and Large Language Models (LLMs) in the automatic extraction of relevant characteristics of these products, addressing challenges such as structuring technical texts and accurately identifying rules, conditions, and variations.
The research focuses on analyzing the state of the art in technologies such as vector databases, LLMs, knowledge graphs, and agentic workflows, as well as evaluating NLP tools and methodologies. The text goes on to explore the primary challenges encountered during the interpretation of insurance documentation, as well as the transformation of unstructured data into organized formats that are compatible with modelling systems. The solution developed responded satisfactorily to the objectives established, enabling the structured and consistent extraction of product characteristics from regulatory documents. To this end, AI agent-based workflows were used, supported by LLMs and validation schemes, ensuring the quality and consistency of the results.

Keywords: · Document extraction · Named Entity Recognition · Large Language Models · AI Agentic Workflows · Insurance Product

1 Introduction

The insurance sector plays a crucial role in the economy, but the increasing complexity of its products, defined by extensive rules, conditions, and regulatory constraints, makes their development and maintenance challenging.

These specifications are detailed in regulatory documents approved by supervisory authorities, often including contractual terms, actuarial behavior, and business rules. Their unstructured and technical nature makes the interpretation and extraction of information a demanding and time-consuming task. Additionally, frequent regulatory updates require insurers to adapt their products constantly, reinforcing the need for efficiency, precision, and compliance.

Currently, msg Life Iberia consultants manually interpret and translate regulatory specification documents into product modeling requirements. This manual process is inefficient, error-prone, and difficult to scale. Recent advances in AI present an opportunity to automate the extraction of relevant product information from textual sources. However, applying these technologies in the insurance domain remains complex due to the heterogeneity and lack of standard structure across regulatory documents, the difficulty in generalizing extraction rules for different products and insurers, and the frequent incompatibilities between the extracted data and existing modeling platforms.

This project aims to develop a tool capable of automatically interpreting insurance specification documents, extracting the main product characteristics, and converting them into a structured representation compatible with msg Life Iberia's modeling platform.

The main goals are:

- Define a standardized output structure for product information.
- Implement an extraction tool to convert unstructured text into structured data.
- Integrate the extracted data into the company's modeling system to generate initial insurance product models automatically.

This automation seeks to improve accuracy, reduce manual effort, and increase consistency in the transformation of regulatory documents into software-ready models.

The remainder of this document is organised into five sections. Section 2 reviews related work. Section 3 outlines the document analysis and conceptual foundations of the extraction model. Section 4 details the system's development, from document processing to testing. Section 5 presents the results of the proposed approach. Section 6 concludes the report with final considerations and improvement opportunities.

2 Related Work

Research on document understanding and information extraction has evolved significantly with the adoption of NLP and LLMs. Several studies have explored different strategies to transform unstructured insurance or regulatory documents into structured and machine-interpretable data.

Authors in [1] investigated the application of two widely used NLP libraries, NLTK and SpaCy, for automatic text summarization. Their study proposed a hybrid approach that integrates the strengths of both libraries to enhance precision and adaptability in document processing. By employing evaluation metrics such as ROUGE and BLEU, the authors provided practical insights into how these tools can be integrated into systems requiring automated analysis of extensive and unstructured textual data, laying foundational work for structured information extraction.

In the context of robustness in data extraction, work in [2] evaluated multi-modal LLMs capable of processing both textual and visual information. Their experiments demonstrated that integrating visual inputs (e.g., rotated or misaligned documents) significantly improves the extraction of structured data, especially when dealing with scanned or low-quality insurance documents. This work shows how multi-modal models can overcome limitations of traditional OCR-based methods.

Focusing on the insurance sector, the study in [3] explored the use of LLMs such as GPT for actuarial and insurance tasks, including risk modeling, predictive claims analysis, and policy simulation. The authors emphasized the transformative potential of these models in automating analytical processes, while noting challenges related to data privacy, regulatory compliance, and the need for domain-specific fine-tuning.

Earlier research by [4] introduced a semantic, ontology-based framework for processing cyber insurance policies. The proposed system used AI and logic-based reasoning to populate knowledge graphs from policy text, enabling automated identification of inclusions and exclusions. Although effective in capturing semantic relationships, such symbolic approaches are less scalable compared to modern LLM-based techniques.

Overall, the reviewed studies demonstrate a clear evolution from traditional NLP and semantic approaches toward hybrid and LLM-driven systems capable of performing complex reasoning, visual-text integration, and structured data generation. This progression highlights the growing feasibility of applying agentic and workflow-based architectures to automate the extraction of insurance product characteristics from regulatory documents.

3 Analysis

The analysis focused on two main aspects: the structure and characteristics of insurance documents, and the identification of core business concepts that define insurance products. This foundation was essential to address the challenges of processing unstructured information.

Insurance specification documents are the foundation of the entire modeling and automation process. They contain the rules, conditions, and characteristics that define how insurance products operate, and are the primary source from which structured representations must be derived. These documents combine various content types - such as contractual clauses, exclusions, eligibility conditions, and free-form text - which significantly increases their complexity.

Moreover, document structures vary widely between insurers, reinforcing the need for a flexible extraction approach capable of adapting to different layouts and writing styles. Despite these differences, most documents share a common conceptual base, including recurring elements such as products, coverages, conditions, and exclusions.

Based on the analysis of documents and information gathered through meetings with domain experts, a conceptual map of the insurance product model was

defined (cf. Fig. 1). This model identifies the core entities and their relationships, which form the semantic structure of an insurance product.

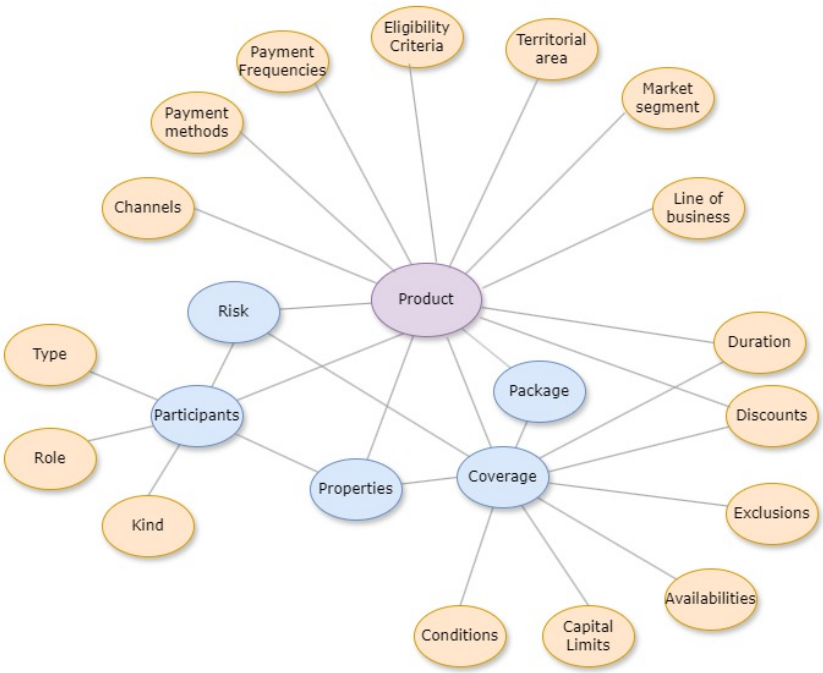


Fig. 1. Concept map of the insurance domain.

The central entity, *Product*, represents the core unit of the insurance offering. Each *Product* may include one or more *Packages*, corresponding to commercial variants or plans. A *Package* groups several *Coverages*, which describe the actual risks insured. Coverages include key elements such as exclusions, applicability rules, duration, limits, and conditions. The entity *Properties* defines variable attributes - such as effective date, birth date, or species - that influence underwriting and risk management. These properties may apply to participants, coverages, or the product itself. *Participants* represent the actors involved in the policy, such as the policyholder, insured person, or beneficiary, whose characteristics may determine eligibility and benefits. Finally, the *Risk* entity encapsulates the insurable exposure associated with the product and serves as a reference for underwriting and pricing rules. Together, these entities - *Product*, *Packages*, *Coverages*, *Properties*, *Participants*, and *Risk* - form a conceptual graph that enables a comprehensive and flexible representation of insurance products.

4 Development

This section details the development of the extraction system, covering document processing strategies, workflow orchestration, schema design, prompting

techniques, and testing methodology. Each subsection describes a specific stage of the implementation, from handling heterogeneous document formats to ensuring structured, validated output.

4.1 Document Processing

Document processing plays a crucial role in ensuring that the textual content extracted from insurance-related documents maintains the integrity and structure of the original source. Given the heterogeneity of file formats, layouts, and levels of complexity, distinct strategies and tools were adopted to handle different types of documents, including those containing tables, long documents, and scanned files.

Before structured information can be extracted, it is essential to correctly load and preprocess the input documents. For this purpose, the **PyMuPDF** library was employed as the main loader. It provides a low-level Python API that grants direct access to document structures - such as text blocks, positions, and metadata - while ensuring high performance even for large files [5]. Furthermore, it allows text extraction in Markdown format, facilitating interpretation by LLMs.

However, when processing documents containing complex tables, PyMuPDF proved insufficient for preserving the original tabular layout. In such cases, the **Docling** loader was used, offering superior preservation of table structures through native conversion to Markdown [6]. This approach ensures that key insurance data, such as coverage limits and package details, are accurately associated with their corresponding headers. Although more computationally demanding, Docling consistently provided better accuracy for complex tables, while PyMuPDF remained faster and effective for simpler cases.

Processing **long documents** introduced another challenge: maintaining sufficient context for the LLM without exceeding token limits. To address this, a batching strategy was initially adopted, splitting documents into multiple segments processed in parallel. Despite improving execution time, this method often produced duplicated or inconsistent outputs. Consequently, a sequential and incremental approach was implemented, preserving contextual continuity between batches and leading to more coherent and reliable extractions.

Finally, for **scanned or image-based documents**, traditional loaders failed to detect textual content due to the absence of machine-readable text. In these cases, an LLM-based loader was employed to perform Optical Character Recognition (OCR) directly through the model, converting the base64-encoded image of the document into structured text. This approach proved to be the only viable solution for such documents, successfully recovering the complete textual content with high fidelity to the original layout.

4.2 System Flow

The current system provides the LLM with the complete product schema, guiding the extraction process and reducing ambiguity. Fig. 2 illustrates the system

workflow. Once a document is uploaded, a central router determines the appropriate extraction path, invoking specialized agents such as the *Product Info Extractor*. This agent includes sub-agents responsible for specific extraction tasks and supports two main modes: document analysis or structured information extraction.

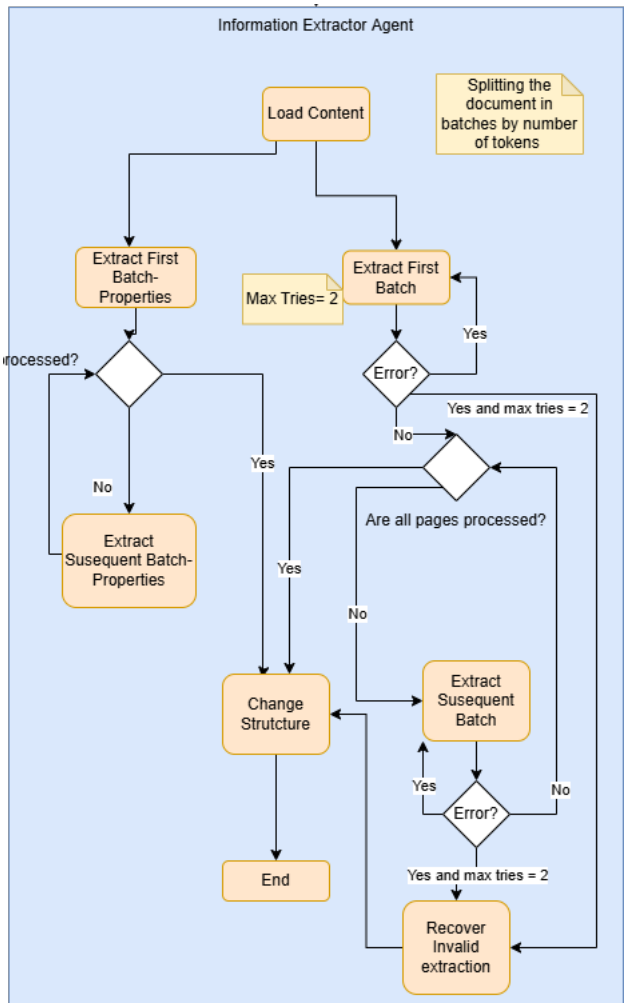


Fig. 2. Fluxo do Sistema na terceira abordagem

During the extraction phase, the document content is first divided into batches based on token count. This segmentation is necessary to manage context size and ensure efficient interaction with the LLM. Two agents operate in parallel on the initial batch: *Extract First Batch Properties* and *Extract First Batch*, which focus respectively on product properties and product characteristics. For larger documents, subsequent batches are processed iteratively by the *Extract Subse-*

quent Batch agent, which integrates newly extracted information with previously processed data to maintain contextual consistency.

The system also incorporates reliability mechanisms to handle potential failures during LLM interaction. If the extraction fails due to schema complexity or data quality issues, the process can automatically restart, with up to two attempts. Additionally, when partial results contain invalid or incomplete information, the *Recover Invalid Information* agent filters and restores valid data, ensuring maximum retention of useful content.

Finally, once all product details are extracted, the information is consolidated in the *Change Structure* module, responsible for transforming and formatting the results into the structure required by the *Product Machine*. This guarantees full compatibility with downstream systems for product import or update operations. The overall workflow thus ensures a robust, adaptive, and context-aware extraction process that balances efficiency, reliability, and data integrity.

4.3 Schemas and Structured Output

The *Pydantic* schemas constitutes the foundation of the system's structured output mechanism. It defines how extracted data is organized, validated, and represented throughout the pipeline. Acting simultaneously as a *data schema* and a *validation layer*, Pydantic ensures that all information produced by the LLM adheres to the predefined structure, maintaining both semantic integrity and technical consistency [7]. In practice, Pydantic serves as the formal contract between the extraction layer and the Product Machine, guaranteeing that outputs are immediately compatible with downstream components.

Each schema encapsulates detailed field descriptions and representative examples, providing the LLM with explicit guidance on the expected format and meaning of each attribute. In the insurance domain, where entities are highly interrelated, this structured definition is crucial. A product may include multiple coverages, participants, and modalities—each governed by its own Pydantic schema. This modular design mirrors the hierarchical organization of insurance products and enables consistent validation across heterogeneous documents.

In parallel, independent property schemas capture variable parameters—such as insured age, coverage limits, and contract duration—that influence product definitions. These schemas are later merged into the unified product model, allowing flexible yet controlled integration of all extracted elements within a single, validated structure. This structured representation allows the system to model complex relationships with precision and supports accurate, schema-driven extraction.

The extraction process itself is tightly coupled with these schemas. Using the *TrustCall* library, the system enforces schema-guided responses through Pydantic models. This ensures that every output generated by the LLM is automatically validated against the expected data structure. The process starts with *with_ structured_ output*, which applies Pydantic validation to the LLM's response, and then iteratively refines invalid or incomplete fields via targeted

patch operations. This approach allows for progressive, lossless refinement of the extracted data object while preserving all previously validated content [8].

At runtime, an extractor is initialized and configured with the appropriate Pydantic schema, guiding the LLM toward structured and compliant outputs. As documents are processed in batches, the system incrementally updates the existing structured object through controlled *patches*, maintaining alignment with the schema and ensuring consistent structured output across multiple extraction iterations.

4.4 Prompts

Prompts play a central role in guiding the LLM during the extraction process. They serve as the main communication interface between the user or system and the model, providing clear and precise instructions that define the extraction objectives. The quality and clarity of a prompt directly affect the coherence and accuracy of the generated output, especially in structured data extraction tasks.

Following OpenAI's official prompt design guidelines [9], three main prompt types were considered: *zero-shot* (instructions without examples), *few-shot* (with illustrative examples), and *chain-of-thought* (step-by-step reasoning). In this project, carefully crafted prompts ensured that the LLM adhered to the predefined data schema and minimized formatting or validation errors.

Different agent tasks required specific prompts, though common components were often reused to maintain consistency, particularly when processing documents in batches. For initial batches, prompts focused on extracting data according to the schema, while subsequent batches were instructed to update existing structures with new information. All prompts were written in Markdown format, clearly organizing roles, objectives, detailed instructions, and document content to optimize the LLM's understanding and output quality.

4.5 Testing and Evaluation

The entire development process followed an *evals*-oriented approach, where testing played a central role in ensuring reliability and performance. The main goal of these tests was to detect and minimize anomalies by continuously evaluating the extraction results against a predefined *Golden Standard Output*. This benchmark allowed for fast, objective validation without manual inspection after each extraction [10].

A similarity-based scoring method was used to compare each extracted output with its reference file, using the *Non-LLM String Similarity* metric. A similarity threshold of 0.70 (on a 0–1 scale) was empirically defined to account for semantically equivalent but textually different responses. This metric objectively measured structural and semantic coherence between runs and helped track performance variations over time.

To evaluate robustness, tests were performed on documents of varying complexity - simple, semi-complex, and complex - reflecting different structural and

formatting challenges. Additionally, a complementary test was introduced based on counting extracted elements per field, enabling detection of omissions regardless of wording differences. Although this approach had limitations (e.g., merged or split fields), combining it with similarity-based tests provided a reliable overall evaluation of the extraction quality and consistency across multiple executions.

5 Results

Overall, the final approach demonstrated more consistent and reliable results compared to previous iterations.

Table 1. Scores obtained over multiple runs (final approach)

Run	Simple	Semi_complex	Complex
1	0.8142	0.6251	0.7549
2	0.7544	0.6002	0.7379
3	0.7747	0.6600	0.7633
4	0.8388	0.6408	0.6174
5	0.8613	0.6408	0.7712
6	0.7497	0.7736	0.7595
7	0.8240	0.7681	0.5891
8	0.7624	0.7530	0.7675
9	0.7691	0.7210	0.5888
10	0.7875	0.6667	0.7718

The values in Table 1 represent similarity scores where higher values indicate closer alignment between the extracted output and the reference Golden Standard. Simpler documents achieved the highest and most stable scores, while semi-complex and complex ones showed slightly more variation, as expected due to their structural diversity. Importantly, integration tests revealed a significantly lower failure rate, confirming that most fields were accurately extracted. Manual validation further supported these findings, showing that the overall extraction quality achieved by the system is satisfactory and suitable for practical application.

6 Conclusion

Insurance product specifications are traditionally delivered as long, heterogeneous regulatory documents, making manual interpretation slow, costly, and error-prone. This project addressed the challenge of automating the extraction of structured product characteristics from these unstructured sources using LLM-based workflows. First, a unified output structure was defined and refined throughout the project, providing a consistent schema applicable to all document

types. Second, a functional prototype was developed to extract product characteristics from unstructured insurance documents, producing structured outputs aligned with the defined schema. Finally, the system was integrated with the PM, enabling the automatic generation of insurance product templates through an API-based connection.orgDespite the positive results, several limitations remain. LLMs can occasionally generate inaccurate or hallucinated responses, which affects data reliability. A possible mitigation would involve assigning confidence scores to each extracted field to support automatic validation. Performance efficiency is another concern, as extraction can be computationally expensive due to multiple LLM calls. Future improvements could include caching mechanisms, intelligent pre-filtering of content, and context compression using semantic embeddings to reduce redundant processing and optimize both runtime and cost.

Beyond the insurance sector, the proposed approach can be applied to other domains that rely on complex, heterogeneous, and technical documents. Its workflow, especially the separation of document loading, batch processing, and schema validation, supports adaptation to a wide range of file formats, including PDF, Word and Excel. In most cases, updating the schema to reflect the target domain is sufficient for the workflow to operate effectively.

References

1. Amade, D., Chandra, R., Sinha, V. K., Anand, D.: Automatic Text Summarization Using NLTK & SpaCy. *SSRN Electronic Journal* (2024). https://doi.org/10.2139/SSRN.4742012
2. Biswas, A., Talukdar, W.: Robustness of Structured Data Extraction from In-plane Rotated Documents using Multi-Modal LLMs. *arXiv preprint* arXiv:2402.07153 (2024).
3. Balona, C.: ActuaryGPT: Applications of Large Language Models to Insurance and Actuarial Work. *SSRN Electronic Journal* (2024). https://doi.org/10.2139/ssrn.4750341
4. Joshi, K., Pande Joshi, K., Mittal, S.: A Semantic Approach for Automating Knowledge in Policies of Cyber Insurance Services. In: *Proceedings of the 2020 IEEE International Conference on Web Services (ICWS)*, pp. 33–40. IEEE (2020). https://doi.org/10.1109/ICWS.2020.00018
5. PyMuPDF Developers: *PyMuPDF Documentation.* (2025). Available at: https://pymupdf.readthedocs.io/en/latest/
6. LangChain Documentation: *Docling Document Loader Integration.* (2025). Available at: https://python.langchain.com/docs/integrations/document_loaders/docling/
7. Pydantic Team: *Steering Large Language Models with Pydantic.* Pydantic Documentation (2024). Available at: https://pydantic.dev/articles/llm-intro
8. LangGraph Community: *TrustCall Integration with LangGraph.* LangChain Hub (2024). Available at: https://smith.langchain.com/hub/langchain/trustcall (Acedido em 7 set. 2025)
9. MacCallum, N., Lee, J.: *GPT-4.1 Prompting Guide.* OpenAI Cookbook (2025). Available at: https://cookbook.openai.com/examples/gpt4-1_prompting_guide
10. EvalOps: Evaluation-Driven Development: Building LLM Features with Confidence.
    Available at: https://www.evalops.dev/blog/evaluation-driven-development

Metodologias Ágeis e Ensino Invertido na Engenharia: Um Caso de Aplicação Real

Miguel Correia[1], Ana Ribeiro[2], Bruno Costa[3], Patrick Costa[4], Alberto Pereira[5], and Marílio Cardoso[6]

Instituto Superior de Engenharia do Porto (ISEP),
Licenciatura em Engenharia de Sistemas, Porto, Portugal
{1231245[1],1230654[2],121247[3],1230881[4],abp[5],joc[6]}@isep.ipp.pt

Empresa Parceira: Tlantic

Resumo Este artigo apresenta um estudo de caso sobre a integração de metodologias ágeis, concretamente *Scrum*, e a aplicação do modelo de ensino invertido (*Flipped Learning*) no contexto do ensino superior. O objetivo central foi o desenvolvimento de um *Portal do Colaborador*, uma aplicação web funcional e escalável, denominada *SmarterHUB*, destinada à gestão interna de recursos humanos. O estudo decorreu em quatro semanas intensivas, combinando *learning by doing*, interdisciplinaridade e trabalho em equipa, sustentado pelos princípios do *Flipped Learning Framework*.

Palavras-Chave: Scrum · Ensino Invertido · Learning by Doing · Metodologias Ágeis · Autoeficácia · Aprendizagem Ativa

1 Introdução

Este estudo descreve a implementação de uma abordagem pedagógica baseada em metodologias ágeis e ensino ativo no contexto do ensino superior em Engenharia de Sistemas. A investigação insere-se na unidade curricular *Laboratórios de Sistemas I (LSIS1)* do Instituto Superior de Engenharia do Porto (ISEP) e centra-se na resposta a uma necessidade real apresentada pela empresa parceira **Tlantic**, que identificou a ausência de uma plataforma centralizada para a gestão de colaboradores, processos de formação e documentação interna, tarefas até então realizadas de forma manual e dispersa através de folhas de cálculo.

O desafio proposto consistiu no desenvolvimento de um **Portal do Colaborador**, designado *SmarterHUB*, capaz de digitalizar e automatizar os processos de recursos humanos da empresa. O sistema deveria integrar gestão de perfis e equipas, acompanhamento de formações, controlo de dados contratuais, *dashboards* de indicadores e geração automática de relatórios, assegurando simultaneamente acessibilidade, segurança e escalabilidade.

Para além da vertente técnica, o projeto foi concebido como uma experiência de aprendizagem prática, alinhada com metodologias de ensino ativo. A equipa

aplicou *Scrum* em sprints semanais e o modelo *Flipped Learning*, aproximando o trabalho de um ambiente profissional de engenharia de software, com contacto regular com o cliente e entrega de incrementos funcionais.

O *SmarterHUB* constituiu, assim, uma resposta a um problema empresarial concreto e uma imersão realista no ciclo de vida de um projeto de software, através de uma aplicação web moderna em PHP e MySQL que contribui para a modernização de processos internos.

2 Fundamentação Teórica: Flipped Learning em LSIS

O modelo de ensino invertido (*Flipped Learning*) tem ganho destaque no ensino superior, sobretudo em Engenharia, por potenciar a aprendizagem autónoma prévia e a aplicação prática em aula [1]. Valoriza-se a preparação individual através de vídeos e leituras, reservando o tempo presencial para atividades de interação, resolução de problemas e trabalho colaborativo [2].
Sustentada em princípios construtivistas, esta abordagem redefine o papel do docente, que assume funções de orientação e facilitação, promovendo autonomia, responsabilização e competências transversais essenciais à prática profissional.
Conforme indicado por Eppard e Rochdi [3], a metodologia articula-se com a Taxonomia de Bloom revista: os níveis cognitivos inferiores são trabalhados fora da aula, enquanto os superiores são explorados presencialmente, reforçando a aprendizagem ativa.
No âmbito da unidade *Laboratórios de Sistemas I (LSIS1)*, os estudantes dispõem de quatro semanas exclusivas para o projeto, exigindo estudo autónomo rigoroso e documentação contínua. Esta aprendizagem independente incidiu sobre tópicos como modelação relacional avançada, padrões *Model-View-Controller* (MVC), autenticação 2FA e a aplicação colaborativa da metodologia *Scrum*, incluindo papéis, eventos e artefactos.

3 Metodologia e Framework de Trabalho

A metodologia adotada no projeto combinou práticas de gestão ágil com uma organização estruturada do trabalho, de forma a garantir iteração rápida, alinhamento com os requisitos e desenvolvimento sustentado. Nesta secção descrevem-se os princípios orientadores da abordagem utilizada, bem como a forma como a equipa se organizou, distribuiu responsabilidades e operou ao longo das quatro semanas de desenvolvimento intensivo.

3.1 Estrutura e papéis

A equipa organizou-se segundo a metodologia *Scrum*, adotando ciclos curtos, foco no valor entregue e um conjunto de papéis, eventos e artefactos que garantem transparência e melhoria contínua.

Com quatro elementos, a estrutura foi adaptada ao contexto académico. O *Scrum Master*, Miguel Correia, assegurou a coordenação e a gestão de tempo, enquanto o *Product Owner*, Bruno Costa, comunicou com docentes e com a empresa parceira, gerindo prioridades e o *Product Backlog*. A empresa **Tlantic** manteve interação regular com o grupo, alinhando o desenvolvimento com necessidades reais, e os docentes atuaram como mediadores e validadores das entregas.
As responsabilidades distribuíram-se da seguinte forma:

- **Miguel Correia (Scrum Master):** coordenação, facilitação e acompanhamento do progresso;
- **Ana Ribeiro (Developer / Business Analyst):** análise funcional, conceção de *dashboards* e integração matemática/estatística;
- **Patrick Costa (Lead Developer):** arquitetura, base de dados e implementação das funcionalidades principais;
- **Bruno Costa (Product Owner / Developer):** priorização, gestão de requisitos e garantia de qualidade.

Para além das funções individuais, todo o grupo participou no controlo de qualidade. Cada tarefa desenvolvida tinha de ser revista pelos restantes membros, que validavam o código e a conformidade funcional antes de marcar *review done*. Este processo assegurou consistência técnica, partilha de conhecimento e qualidade das entregas.

3.2 Processo e ferramentas

O projeto desenvolveu-se em quatro sprints semanais, permitindo iteração, recolha de *feedback* e entregas funcionais progressivas validadas por docentes e pela empresa.
As principais cerimónias de *Scrum* foram:

- **Sprint Planning:** definição de objetivos e tarefas a partir do *Product Backlog*, com estimativas via *Planning Poker* [4];
- **Daily Meetings:** sincronização da equipa e identificação de obstáculos;
- **Sprint Review:** apresentação dos incrementos e recolha de *feedback*;
- **Sprint Retrospective:** análise do processo e proposta de melhorias [5].

O trabalho foi apoiado por **Microsoft Teams** e **Planner** para comunicação e organização, **GitHub** para controlo de versões [6] e gráficos *Burndown* e *Burnup* para monitorizar progresso e capacidade.
A natureza iterativa do processo exigiu adaptação constante a imprevistos técnicos e novas solicitações, aproximando o trabalho da dinâmica típica de equipas profissionais de engenharia de software.

4 Desenvolvimento Técnico do Sistema

O desenvolvimento do *SmarterHUB* (projeto inserido na unidade curricular de LSIS1) seguiu uma abordagem iterativa e orientada a requisitos reais definidos pela empresa parceira, integrando decisões arquiteturais, funcionalidades nucleares e componentes analíticas que reforçaram a maturidade do protótipo.

4.1 Arquitetura

O sistema, desenvolvido em **PHP 8** e **MySQL 8**, adota a arquitetura **MVC**. A aplicação organiza-se em três camadas:

1. **UI (View):** interface responsiva para diferentes perfis, construída em HTML5, CSS3 e JavaScript;
2. **BLL:** regras de negócio, validações e cálculos estatísticos;
3. **DAL:** acesso seguro à base de dados com *prepared statements*.

Para a componente visual utilizaram-se a *Canvas API* [7] e *CanvasJS* [8], permitindo gráficos interativos e personalizáveis. O desenvolvimento foi incremental, com revisões semanais e integração contínua.

4.2 Funcionalidades Principais

O sistema cumpriu todos os requisitos validados pela **Tlantic**, totalizando mais de cinquenta tarefas distribuídas pelos quatro sprints. Entre as funcionalidades centrais destacam-se:

- **Autenticação 2FA**;
- **Gestão de perfis, equipas e contratos**;
- ***Dashboards* de KPIs**;
- **Relatórios PDF** automáticos;
- **Importação/Exportação CSV**;
- ***Logs*** e permissões multinível.

4.3 Funcionalidades Extra e Inovação

- ***Chatbot*** integrado via `https://www.chatbase.co`;
- ***Dashboards*** personalizáveis para RH;
- **Relatórios PDF** com `fpdf`;
- **Mensagens internas** com histórico e notificações.

4.4 Componente Analítica

A análise exploratória utilizou métricas da biblioteca *Simple Statistics* [9], como correlações de Pearson e Jaccard. As visualizações - gráficos de dispersão, barras empilhadas e *boxplots* - foram produzidas com *CanvasJS* e *Canvas API*, apoiando a interpretação dos dados e a tomada de decisão.

5 Resultados e Impacto

5.1 Resultados Técnicos

O *SmarterHUB* foi entregue funcional no final da quarta *sprint*, ultrapassando o previsto no *Product Backlog*. No total, foram concluídas **97 tarefas**, incluindo requisitos obrigatórios e funcionalidades extra.

Testes de integração e usabilidade, acompanhados por *feedback* contínuo da **Tlantic**, evidenciaram uma interface clara, consistência visual e um tratamento de dados seguro. A aplicação apresentou estabilidade, bom desempenho e um sistema de autenticação fiável.
O trabalho revelou planeamento consistente, documentação completa e aplicação de práticas seguras e escaláveis, suportadas por relatórios técnicos e documentação complementar.

O desempenho da equipa recebeu reconhecimento académico e institucional: o projeto foi considerado o **melhor da Licenciatura em Engenharia de Sistemas**, alcançando **19 valores**, com distinção individual de **20 valores** para o Scrum Master. A **Tlantic** distinguiu igualmente o produto como **o melhor projeto da unidade curricular**, destacando a qualidade técnica e o rigor metodológico.

5.2 Impacto Educacional

O modelo pedagógico *Flipped + Scrum* utilizado no projeto fomentou autonomia, aprendizagem ativa e responsabilidade individual, incentivando a colaboração e o planeamento iterativo. Foram desenvolvidas competências técnicas (modelação de dados, segurança e visualização) e competências transversais (gestão de tempo, comunicação, pensamento crítico e tomada de decisão). As *sprint reviews* e *retrospectives* proporcionaram momentos estruturados de reflexão e melhoria contínua, reforçando a maturidade da equipa e o sentido de autocrítica.

5.3 Impacto Organizacional

Para a empresa **Tlantic**, o sistema desenvolvido representa uma solução que centraliza informação, automatiza processos e aumenta a fiabilidade dos dados. As funcionalidades adicionais aproximaram o protótipo de um produto potencialmente utilizável em contexto real, reforçando a transferência de conhecimento entre o meio académico e o empresarial.

6 Recolha e Análise dos Dados

A integração entre metodologias ágeis e *Flipped Learning* assume uma dimensão pedagógica significativa, contribuindo para o desenvolvimento de competências técnicas, cognitivas e sociais relevantes para a Engenharia. Para comprovar empiricamente esta eficácia, foi realizada uma análise quantitativa e qualitativa centrada na retenção de conhecimento, autonomia, autoeficácia e capacidade de adaptação ao contexto profissional.

6.1 Instrumentos de Avaliação

A recolha de dados recorreu a dois instrumentos complementares, permitindo analisar dimensões cognitivas e motivacionais:

1. **TRLA Teste de Retenção e Literacia Aplicada:** aplicado a 16 estudantes três meses após o projeto, avaliando retenção de conteúdos relacionados com *Scrum*, modelação relacional, segurança e metodologias ativas. As respostas objetivas foram convertidas numa pontuação percentual, calculada por:

$$\text{Percentagem} = \frac{\text{Respostas corretas}}{\text{Total de questões}} \times 100$$

   garantindo uniformidade e permitindo a análise estatística subsequente.
2. **ALSES Escala de Autoeficácia e Aprendizagem Ativa em Contexto Ágil:** aplicada ao grupo de desenvolvimento (4 elementos), composta por 12 itens de perceção sobre autonomia, motivação, aprendizagem independente e impacto das práticas ágeis. Utilizou-se uma escala *Likert* de 0 a 7, escolhida pela sua sensibilidade e capacidade de discriminar variações subtis nas perceções individuais.

6.2 Procedimentos de Recolha

Ambos os instrumentos foram aplicados no final do projeto, com o intuito de recolher evidências consolidadas da experiência formativa. Participaram:

- **16 estudantes do curso**, que realizaram a mesma unidade curricular, permitindo avaliar a retenção geral de conhecimento após três meses da conclusão do projeto;
- **4 elementos do grupo ABMP**, que responderam à escala ALSES, fornecendo perceções diretas sobre autoeficácia, motivação e aprendizagem ativa em contexto ágil.

6.3 Resultados e Discussão

Resultados do TRLA - Teste de Retenção e Literacia Aplicada O TRLA foi respondido por 16 estudantes, com os seguintes resultados individuais (em percentagem): 70, 90, 100, 90, 90, 90, 90, 90, 100, 80, 90, 100, 100, 100, 90, 90.

Tabela 1. Resultados descritivos do TRLA (n=16)

Estatística	Valor (%)	Interpretação
Média	91.25	Retenção muito elevada
Mediana	90.00	Consistência nos resultados
Desvio padrão	7.9	Baixa dispersão
Amplitude	30	Variabilidade natural reduzida

Os resultados demonstram uma retenção média de 91.25%, com 75% dos participantes a atingirem 90% ou mais. Esta consistência confirma que os conteúdos

abordados no projeto (*Scrum*, modelação relacional e segurança de aplicações) foram efetivamente assimilados e mantidos ao longo do tempo.

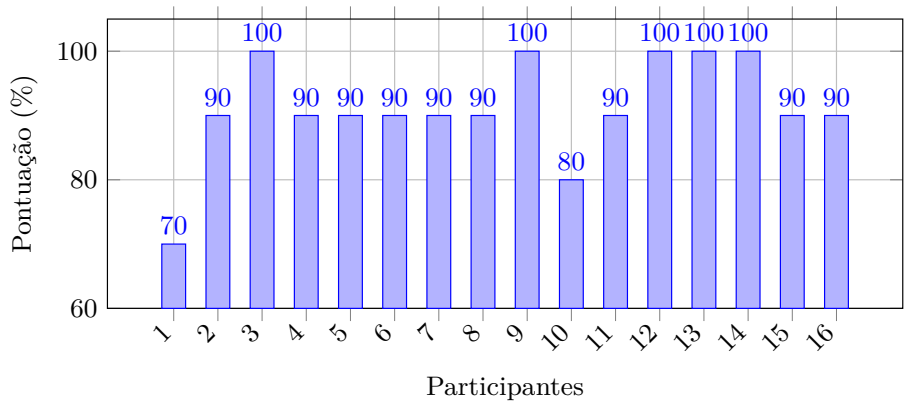


Fig. 1. Distribuição das pontuações individuais no TRLA (n=16).

A elevada concentração de valores acima dos 85% (ver Fig. 1) reforça a eficácia da abordagem baseada em *learning by doing*, evidenciando a transferência de conhecimento da prática para a teoria e vice-versa.

Resultados do ALSES - Autoeficácia e Aprendizagem Ativa O ALSES, aplicado internamente ao grupo ABMP (4 elementos), avaliou a perceção de autoeficácia e aprendizagem ativa em 12 dimensões, numa escala de 0 a 7. A Tabela 2 apresenta os resultados médios obtidos.

Tabela 2. Resultados médios do ALSES (n=4)

Dimensão Avaliada	**Média (07)**	**Interpretação**
Compreensão do Scrum	6.25	Elevado domínio
Planeamento e estimativa realista	5.25	Boa, mas perfectível
Foco e comunicação (Dailies)	6.75	Muito eficaz
Organização e disciplina ágil	6.75	Excelente consistência
Integração de feedback (Reviews)	6.25	Boa capacidade adaptativa
Reflexão e melhoria (Retrospectives)	5.25	Potencial a reforçar
Aprendizagem iterativa	7.00	Impacto máximo
Ensino invertido e autonomia	5.75	Boa adaptação
Responsabilidade individual	6.50	Forte sentido de accountability
Retenção via prática	6.50	Aprendizagem consolidada
Investigação própria	6.00	Boa capacidade exploratória
Motivação global	6.50	Altamente positiva
Média geral	**6.29**	**89.9% de eficácia**

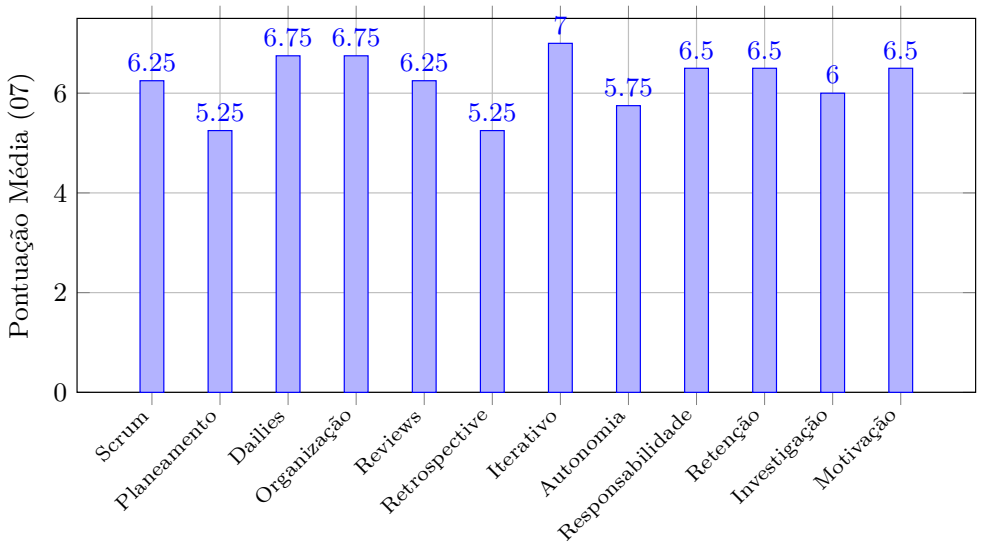


Fig. 2. Distribuição das médias por dimensão (ALSES).

A pontuação média global de 6.29 (89.9%) (ver Fig. 2) revela perceções muito positivas sobre a eficácia da metodologia. As dimensões com menor pontuação concentram-se no planeamento e reflexão, aspetos típicos de equipas em fase inicial de maturação ágil.

Comparação Geral dos Instrumentos A Tabela 3 sintetiza os resultados normalizados entre os dois instrumentos aplicados.

Tabela 3. Comparação geral dos instrumentos aplicados

Instrumento	Média Normalizada (%)	Dimensão Avaliada
TRLA	91.25	Retenção de conhecimento técnico
ALSES	89.9	Autoeficácia e aprendizagem ativa

As médias elevadas e a pequena diferença percentual entre os instrumentos evidenciam a coerência dos resultados: a metodologia *Scrum + Flipped Learning* influencia positivamente tanto a retenção cognitiva como o desenvolvimento de competências socio-profissionais.

6.4 Conclusão da Análise

Os resultados confirmam que metodologias de aprendizagem ativa aliadas a práticas ágeis favorecem retenção de conhecimento e desenvolvimento de competências profissionais. A convergência entre TRLA e ALSES evidencia ganhos cognitivos, motivacionais e técnicos, reforçando a eficácia do modelo adotado.

7 Discussão

A estrutura iterativa do *Scrum* promoveu metacognição, melhoria contínua e compreensão mais profunda do processo de desenvolvimento. O ensino invertido criou condições para aplicar conceitos, discutir soluções e partilhar conhecimento durante as sessões presenciais.

A literatura [2,3] sublinha o papel do docente como facilitador em ambientes colaborativos. No *SmarterHUB*, docentes, estudantes e empresa parceira contribuíram para um contexto de aprendizagem prática e orientada a desafios reais.

As restrições de tempo, a complexidade técnica e a evolução de requisitos transformaram-se em oportunidades de desenvolver competências essenciais: planeamento, comunicação, liderança técnica e adaptação - aspetos centrais na Engenharia de Sistemas.

8 Conclusões e Trabalho Futuro

O projeto mostrou que a combinação entre *Scrum* e *Flipped Learning* é eficaz em Engenharia, promovendo autonomia, responsabilidade e articulação consistente entre teoria e prática. Os resultados evidenciam melhorias na retenção, autoeficácia e desempenho técnico.

Modelos centrados no estudante potenciam competências valorizadas profissionalmente, como comunicação, pensamento crítico e resolução de problemas. No plano técnico, o desenvolvimento iterativo do *SmarterHUB* demonstrou ganhos de qualidade, adaptação e foco em necessidades reais.

Reconhecem-se limitações - nomeadamente a dimensão da amostra e a dependência de perceções - apontando para a necessidade de estudos mais amplos e métodos de avaliação complementares.

A **Tlantic** validou a utilidade do sistema e o rigor do processo, reforçando o potencial da abordagem adotada.

Como trabalho futuro, salientam-se:

- aplicação do modelo em outras unidades curriculares e contextos comparativos;
- estudos longitudinais sobre transferência de competências para o mercado de trabalho;
- diversificação de instrumentos de avaliação;
- evolução técnica da plataforma (API REST, testes automatizados, monitorização).

Em síntese, a articulação entre metodologias ágeis e ensino invertido revela-se uma via pedagógica robusta e alinhada com as exigências contemporâneas da Engenharia, abrindo oportunidades para investigação e aplicações futuras.

Agradecimentos

Os autores expressam o seu agradecimento à **Tlantic** pela colaboração e *feedback* contínuo em todas as fases do projeto, que garantiram relevância prática e contextualização real do trabalho desenvolvido.

O grupo reconhece igualmente o contributo do **corpo docente do ISEP**, em particular:

- Professor **Marílio Cardoso**, docente responsável pela unidade curricular LSIS1 (`joc@isep.ipp.pt`);
- Professor **Paulo Proença**, pelo apoio em desenvolvimento web e bases de dados (`prp@isep.ipp.pt`);
- Professor **Alberto Pereira**, pela orientação técnica e metodológica (`abp@isep.ipp.pt`);
- Professor **Pedro Guedes**, pelo suporte em integração de sistemas e engenharia de software (`pbg@isep.ipp.pt`).

A abertura à inovação pedagógica e ao trabalho colaborativo foi determinante para o sucesso do projeto, sintetizando de forma exemplar a ligação entre academia e indústria.

Referências

1. R. Brewer and S. Movahedazarhouligh, "Successful stories and conflicts: A literature review on the effectiveness of flipped learning in higher education," *Journal of Computer Assisted Learning*, vol. 34, no. 4, pp. 409–416, 2018.
2. J. Yarbro, K. M. Arfstrom, K. McKnight, and P. McKnight, "Extension of a review of flipped learning," Flipped Learning Network, Tech. Rep., 2014.
3. J. Eppard and A. Rochdi, "A framework for flipped learning," in *13th International Conference on Mobile Learning*, 2017, pp. 33–40.
4. P. Poker, "Planning poker," http://www.planningpoker.com, 2008, acedido em 15 de abril de 2016.
5. A. Przybyek, M. Albecka, O. Springer, and W. Kowalski, "Game-based sprint retrospectives: multiple action research," *Empirical Software Engineering*, vol. 27, no. 1, p. 1, 2022.
6. J. D. Blischak, E. R. Davenport, and G. Wilson, "A quick introduction to version control with git and github," *PLoS Computational Biology*, vol. 12, no. 1, p. e1004668, 2016.
7. MDN Web Docs, "Canvas api: Tutorial," https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API/Tutorial, 2025, acesso em: 29 de junho de 2025.
8. CanvasJS, "Canvasjs charts introduction," https://canvasjs.com/docs/charts/intro/, 2025.
9. Simple Statistics Contributors, "Simple statistics documentation," https://simple-statistics.github.io/docs/, 2025, acesso em: 29 de junho de 2025.

NetBox-Zabbix Plugin para Gestão e Organização de Data Center

Inês Costa[1,2], Ivo Pereira[1,3] e Daniel Alves de Oliveira[2]

[1] ISEP, Politécnico do Porto, 4249-015 Porto, Portugal
[2] E-goi, 4450-190 Matosinhos, Portugal
[3] INESC TEC, Faculdade de Engenharia, Universidade do Porto, Rua Dr. Roberto Frias, Porto, 4200-465, Portugal
1210814@isep.ipp.pt

Resumo. Este artigo descreve a modernização e centralização do inventário e da monitorização de infraestruturas realizada num estágio curricular de licenciatura. A arquitetura proposta integra o NetBox como sistema de inventário e DCIM (Data Center Infrastructure Management) e o Zabbix como plataforma central de monitorização e alarmística, com autenticação unificada baseada em LDAP (Lightweight Directory Access Protocol). Foi desenvolvido um plugin em Python para sincronização entre o NetBox e o Zabbix; a plataforma de monitorização inclui descoberta automática, templates personalizados, ações corretivas e alertas multicanal, bem como a centralização de tarefas previamente dispersas. A solução foi validada numa prova de conceito em ambiente real de escritório, evidenciando ganhos de visibilidade e eficiência operacional e reduzindo inconsistências entre inventário e monitorização. Embora ainda não esteja implantada no data center principal, a solução encontra-se preparada para escalar; trabalho futuro inclui sincronização em tempo real, novos dashboards e configuração de alta disponibilidade.

Palavras-chave: Gestão de Infraestruturas, Monitorização, Integração de Sistemas, Inventário, Automatização, NetBox, Zabbix, Python.

1 Introdução

O panorama atual evidencia fragmentação entre ferramentas de gestão e monitorização, sem integração nem fonte única de verdade, o que gera inconsistências de inventário, informação desatualizada e maior carga manual. Esta realidade conduz a operações reativas, dificultando a antecipação de incidentes por análise de padrões e aumentando o tempo médio de reparação (MTTR - Mean Time to Repair), com risco acrescido de indisponibilidade, impacto na satisfação dos clientes e na reputação. A dependência de processos manuais limita a escalabilidade e desvia tempo da equipa de iniciativas de maior valor. Face a estes desafios técnicos e riscos de negócio, impõe-se uma solução que privilegie centralização, automação e fiabilidade.

A infraestrutura em estudo suportava e monitorizava serviços críticos (e.g. bases de dados, websites específicos, hardware, testes a portas de rede, entre outros) utilizando

um conjunto de ferramentas de gestão de inventário, endereçamento IP, monitorização e dashboards isoladas. A falta de integração dificultava uma perspetiva atualizada do estado do sistema e aumentava a propensão a erro humano, sobretudo em tarefas de diagnóstico e de resposta a incidentes. Neste cenário, a modernização e consolidação da stack operacional tornaram-se prioritárias.

Apesar de funcionais de forma isolada, as ferramentas existentes apresentavam fragmentação (dados redundantes e desatualizados), subutilização de capacidades (ausência de descoberta/automação coordenada) e baixa escalabilidade organizacional (dificuldade em integrar novos serviços/processos de forma automática). Em particular, a inexistência de um "single source of truth" (SSoT) para inventário e de integrações por API comprometia: (i) a coerência entre inventário e monitorização; (ii) a rapidez na deteção e resposta; e (iii) a rastreabilidade das alterações.

Este projeto teve como objetivos:

1. Centralizar o inventário de recursos (equipamentos, redes, endereços IP, bastidores) e adotar um "source of truth";
2. Unificar a monitorização de ativos e a gestão de alertas;
3. Integrar inventário e monitorização por API, reduzindo inconsistências e esforço manual;
4. Automatizar a descoberta de novos ativos, aplicação de templates, ações corretivas e notificações multicanal;
5. Padronizar tarefas dispersas (e.g. crontabs) e documentação de operação;
6. Validar a abordagem numa prova de conceito em ambiente real, avaliando benefícios e limitações.

O projeto adotou uma abordagem iterativa e incremental, iniciando-se com a avaliação da stack aplicacional e a recolha de informações junto da equipa de Site Reliability Engineering (SRE) para identificar lacunas e redundâncias existentes. Após análise do estado da arte selecionaram-se alternativas open-source que permitiram a centralização das tarefas necessárias - NetBox (inventário/IPAM) e Zabbix (autodescoberta, monitorização e ações corretivas) - promovendo substituição gradual com preservação de dados, integração via APIs (adição automática de hosts, validação cruzada e alarmística centralizada) e consideração do Grafana para reforço de visualização. Em paralelo, a automação (scripts, templates de configurações, atualizações) reduziu a intervenção manual e o risco de erro humano. Foi produzida documentação técnica que facilitou a adaptação de equipa de SRE.

O artigo está organizado da seguinte forma: o capítulo 2 faz uma contextualização do estado da arte, descrevendo trabalhos relacionados; o capítulo 3 apresenta a arquitetura de solução; a implementação e a avaliação são descritas no capítulo 4; e, finalmente, no capítulo 5 são apresentadas as principais conclusões, incluindo limitações e trabalho futuro.

2 Trabalhos relacionados

A literatura recente mostra uma evolução marcada na gestão e monitorização de data centers, impulsionada pela maior complexidade dos sistemas e pela exigência de disponibilidade, eficiência e automação [1]. Estudos de caso evidenciam migrações de stacks fragmentadas para soluções integradas, como a adoção do Zabbix num banco de média dimensão, com ganhos na observabilidade e no tempo de resposta [2], e a transição do projeto Tor de Munin para Prometheus com Grafana, tirando partido de alertas mais avançados (Alertmanager) e de uma linguagem de consulta expressiva (PromQL) [3, 4]. Em paralelo, ferramentas modernas de DCIM, como o NetBox, consolidam a gestão de ativos e IPAM como source of truth e expõem APIs que facilitam integrações com a monitorização (e.g., sincronização NetBox↔Zabbix e enriquecimento de alertas) [5, 6]. Para agregação e visualização unificadas, o Grafana opera como "single pane of glass", integrando métricas de múltiplas origens e reduzindo inconsistências e erros [5, 7]. A automação com Ansible, alinhada com princípios de IaC e práticas DevOps, promove uniformidade de configurações e diminui a propensão a erro humano [8, 9]. No plano arquitetural, surgem propostas avançadas (e.g., Monalytics) que integram monitorização e análise para reduzir latências e custos [10], bem como abordagens distribuídas para IaaS (IaaSMon) com suporte a VMs e integração com plataformas como OpenStack [11]. Adicionalmente, iniciativas de AIOps aplicam aprendizagem automática para previsão de anomalias e otimização operacional [12]. Por fim, a integração da TI com a infraestrutura física, via DCIM e gestão remota, reforça a resiliência em cenários distribuídos e edge [1].

Em síntese, os trabalhos relacionados convergem em três eixos: (i) evolução para soluções de monitorização escaláveis e orientadas a métricas [3, 4]; (ii) centralização da gestão de ativos com suporte a automação e source of truth [5, 6, 8, 9]; e (iii) adoção de arquiteturas proativas/inteligentes (Monalytics, IaaSMon, AIOps) com foco na prevenção e resposta eficiente a falhas [10, 11, 12]. O projeto da empresa em questão posiciona-se neste panorama, aplicando estas práticas para modernizar e automatizar a sua infraestrutura. A escolha da utilização de NetBox (para gestão de inventário e endereçamento IP) e Zabbix (para monitorização centralizada) justifica-se por serem ferramentas open-source e flexíveis, cumprindo os requisitos do projeto.

3 Arquitetura da Solução

A arquitetura é descrita em três níveis complementares, descritos de seguida.

3.1 Arquitetura Lógica de Nível 1 – Vista Lógica Global

A vista lógica de Nível 1 (Fig. 1) apresenta o "Sistema de Gestão e Monitorização de Data Center" como o componente central da arquitetura. Este sistema é responsável por integrar ferramentas díspares através de desenvolvimentos personalizados, nomeadamente um plugin NetBox-Zabbix e scripts de automação, resolvendo assim a fragmentação de ferramentas anterior. O Administrador de TI/SRE interage com este sistema,

que orquestra três serviços externos essenciais: 1) O NetBox, estabelecido como a SSoT para o inventário, onde a solução executa operações de leitura/escrita e expande a interface nativa; 2) O Zabbix, como sistema de monitorização, cuja configuração de hosts e templates é totalmente automatizada pela solução; e 3) O servidor LDAP, que fornece autenticação centralizada e segura para ambas as plataformas. Esta arquitetura garante a transição para uma solução integrada, coesa e robusta para a gestão da infraestrutura.

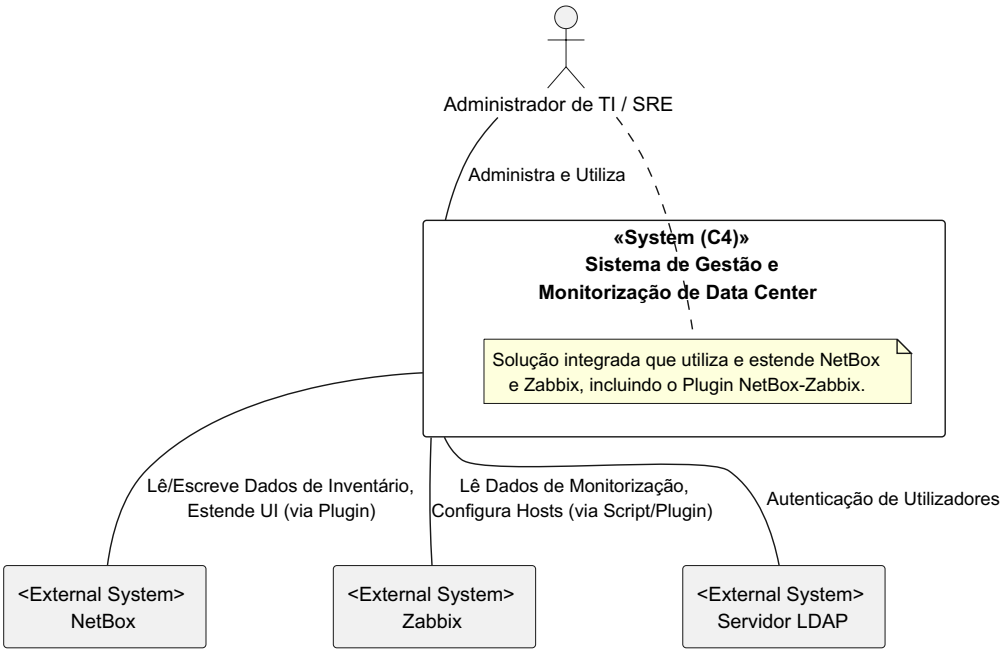


Fig. 1. Vista Lógica de Nível 1

3.2 Arquitetura Lógica de Nível 2 – Decomposição por Camadas/Módulos

A arquitetura lógica de Nível 2 (Fig. 2) decompõe o sistema central em três containers funcionais que promovem uma solução coesa. O primeiro, o NetBox (Sistema de Inventário), atua como a "Fonte da Verdade" (SSoT) e é composto pelo NetBox Core App e pelo Plugin NetBox-Zabbix. Este plugin permite ao Administrador/SRE importar e visualizar hosts descobertos no Zabbix diretamente no NetBox. O segundo container, o Zabbix (Sistema de Monitorização), formado pelo Zabbix Server e Agents, gere a recolha de métricas e alertas, interagindo com o plugin NetBox-Zabbix através da sua API para facilitar a sincronização. Por fim, o Servidor LDAP opera como um sistema externo que centraliza a autenticação de utilizadores para ambas as plataformas, NetBox e Zabbix. Esta arquitetura modular, baseada em APIs e ferramentas open-source, substitui a fragmentação de sistemas anterior por uma solução integrada, flexível e escalável.

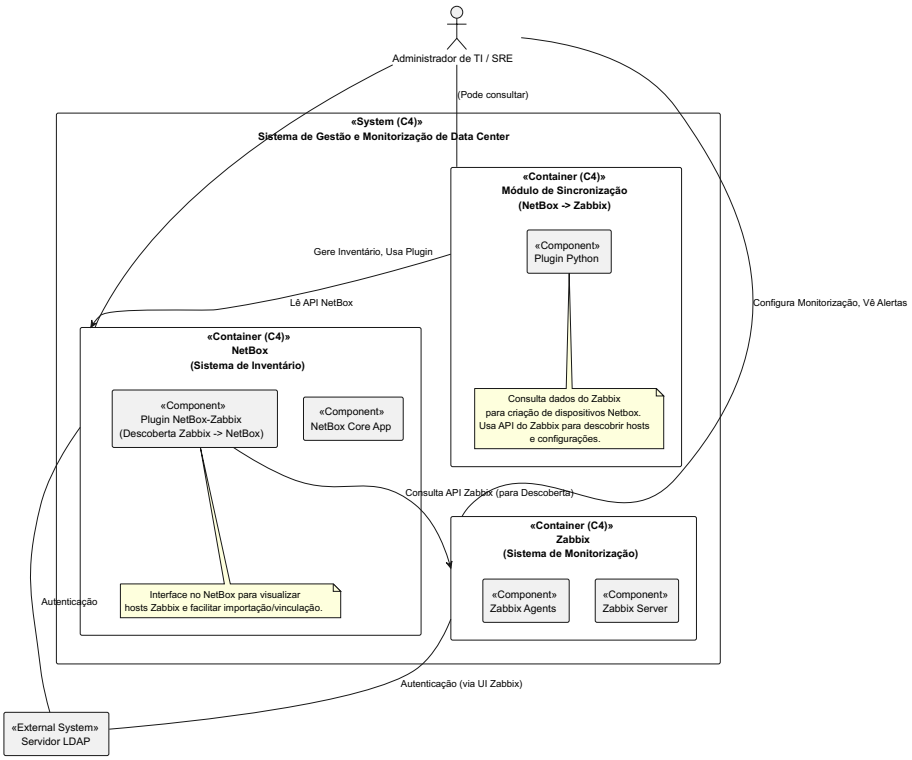


Fig. 2. Vista Lógica de Nível 2

3.3 Arquitetura Lógica de Nível 3 - Design Interno da Integração (Plugin)

A Fig. 3 apresenta a arquitetura interna detalhada do Plugin NetBox-Zabbix, um componente essencial para assegurar a integração e sincronização entre o sistema de inventário NetBox e a monitorização no Zabbix. Este plugin é estruturado de forma modular, com vários componentes internos claramente definidos, cada um com responsabilidades específicas e interações bem estabelecidas.

O ponto central do plugin é o componente Plugin Core (__init__.py), que serve como entrada principal e gere o ciclo de vida do plugin dentro do NetBox Core Framework. Este módulo é responsável pela inicialização e integração geral dos restantes componentes com o NetBox.

A comunicação com o sistema de monitorização é realizada através do Zabbix API Client (api_request.py), que implementa a interface IZabbixAPIAdapter. Este componente gere todas as requisições à API JSON-RPC do Zabbix, garantindo uma comunicação eficiente e estruturada para obter e enviar dados necessários às operações do plugin.

Para representar internamente os hosts descobertos no Zabbix, o plugin usa os Data Models (models.py), nomeadamente a entidade DiscoveredHost. Este modelo de dados, ligado diretamente à base de dados do NetBox através do seu ORM, fornece a estrutura e métodos necessários para manipular os dados relacionados com os hosts a serem sincronizados.

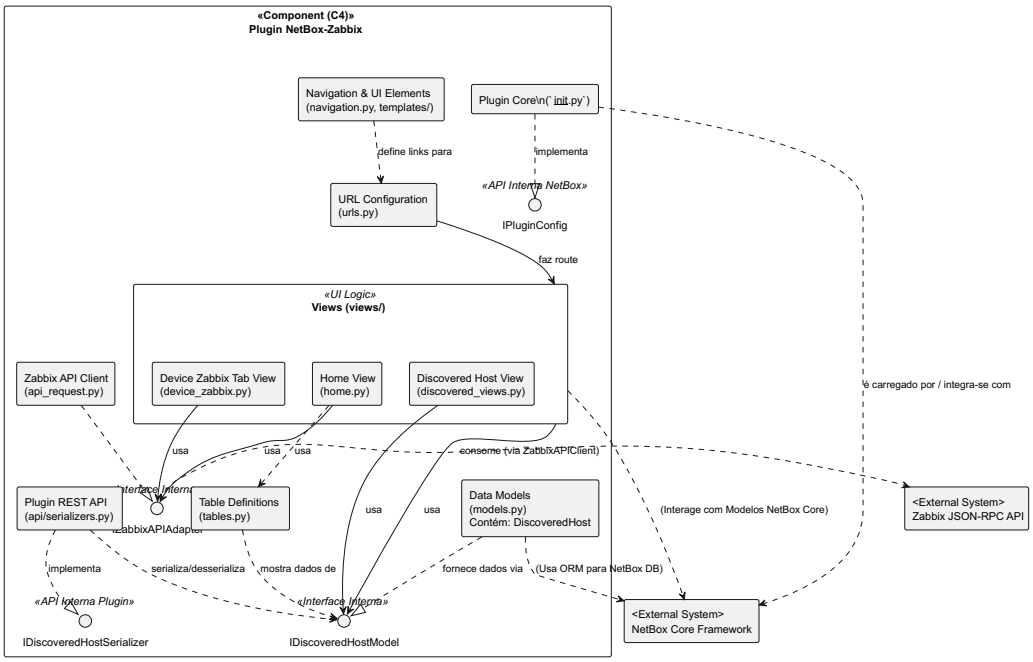


Fig. 3. Vista Lógica de Nível 3 – NetBox-Zabbix Plugin

A interação direta com o utilizador acontece através das Views (views/). Estas são responsáveis pela lógica da interface, incluindo a página principal (Home View), a página de visualização e importação de hosts descobertos (Discovered Host View), e uma aba específica nos detalhes dos dispositivos NetBox para apresentar informações provenientes do Zabbix (Device Zabbix Tab View). As Views recolhem os dados dos Data Models, utilizam o Zabbix API Client para obter informações externas e são disponibilizadas ao utilizador através da configuração de URLs (urls.py).

A configuração dos endpoints da aplicação é feita pelo componente URL Configuration (urls.py), que mapeia claramente cada URL para as Views correspondentes. Já o agrupamento e apresentação visual da interface são suportados pelo componente de Navigation & UI Elements (navigation.py, templates/), que define a forma como o plugin aparece no NetBox e fornece os templates necessários para exibição na interface web.

Os dados exibidos nas interfaces são organizados pelo módulo de Table Definitions (tables.py), estruturando claramente a apresentação das informações provenientes dos Data Models em tabelas visuais consistentes.

Finalmente, o plugin disponibiliza uma interface programática através do componente Plugin REST API (api/serializers.py). Este permite a serialização e desserialização dos dados relacionados com os hosts descobertos, facilitando interações futuras com outros sistemas ou componentes internos do NetBox.

O fluxo principal de interação dentro do plugin inclui duas funcionalidades fundamentais. Primeiro, a descoberta e importação de hosts do Zabbix para o NetBox: o utilizador acede à Discovered Host View, o plugin utiliza o Zabbix API Client para obter

a lista atualizada de hosts disponíveis no Zabbix, e permite ao utilizador importá-los diretamente para o inventário do NetBox através dos Data Models. Segundo, a navegação direta entre NetBox e Zabbix: quando o utilizador consulta detalhes de um dispositivo, uma aba específica permite visualizar rapidamente informações relevantes provenientes do Zabbix sem necessidade de trocar manualmente entre interfaces.

Em resumo, a modularidade desta arquitetura garante flexibilidade e facilidade na manutenção do plugin.

3.4 Componentes do NetBox-Zabbix Plugin

O Diagrama de Componentes (Fig. 4) ilustra a arquitetura modular do NetBox-Zabbix Plugin. O Plugin Core atua como orquestrador central, interagindo com o NetBox Core Framework. A comunicação externa é gerida por um Zabbix API Client dedicado, que implementa uma interface (IZabbixAPIAdapter) para abstrair as chamadas à API JSON-RPC do Zabbix. O módulo Data Models utiliza o ORM do Django para persistir dados na base de dados do NetBox, com destaque para o modelo DiscoveredHost. A lógica de interface (Views) fornece os pontos de interação do utilizador, incluindo a Home View (descoberta), a Discovered Host View (gestão/importação) e a Device Zabbix Tab View (integrada na página do dispositivo NetBox). Módulos auxiliares de URL Configuration e Navigation & UI Elements asseguram a integração visual. Adicionalmente, uma Plugin REST API interna expõe os dados programaticamente, garantindo a modularidade e extensibilidade da solução.

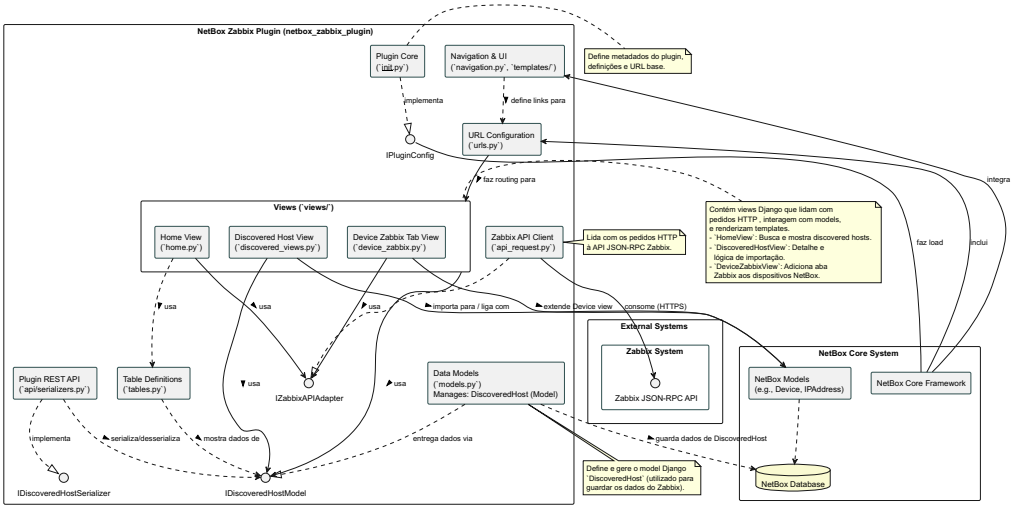


Fig. 4. Diagrama de Componentes – NetBox-Zabbix Plugin

4 Implementação e Avaliação da Solução

Este capítulo descreve em detalhe a implementação da solução definida anteriormente, abordando as tecnologias utilizadas, configuração dos sistemas, integração entre componentes e a validação da solução, com foco na centralização da informação, automatização, escalabilidade e facilidade de manutenção.

4.1 Descrição da Implementação

A implementação da arquitetura foi realizada utilizando uma stack tecnológica open-source composta por NetBox, Zabbix, Python, LDAP (FreeIPA) e NGINX.

- NetBox (Inventário e DCIM): Implementado como "Fonte da Verdade" (SSoT) num container Proxmox. A estruturação do inventário foi realizada por inserção manual e complementada por um plugin de migração (Python) desenvolvido para importar dados de rede do sistema legado LibreNMS.
- Integração NetBox-Zabbix: Foi desenvolvido um plugin Python personalizado que utiliza as APIs JSON-RPC e REST para garantir a consistência de dados. O plugin permite a importação de hosts do Zabbix para o NetBox e a visualização de hiperligações diretas para os dashboards Zabbix a partir da interface do NetBox.
- Zabbix (Monitorização Centralizada): Substituiu um conjunto de ferramentas fragmentadas (Monit, Munin, etc.). A implementação destacou-se pela automação do deployment do Zabbix Agent 2 através de scripts Bash, que configuram os agentes para auto registo, monitorização de systemd e execução de ações corretivas remotas. A solução utiliza LLD para descoberta automática e ações corretivas para remediação de incidentes (e.g.: reinício de serviços).
- Autenticação e Acesso (LDAP e NGINX): O acesso às interfaces do NetBox e Zabbix foi unificado através de integração com o FreeIPA (LDAP). O NGINX foi utilizado como reverse proxy para ambas as aplicações, configurado com HTTPS e regras de segurança.
- Suporte Legacy: Foi garantida a compatibilidade com sistemas operativos mais antigos (e.g.: CentOS 6) e desenvolvidos templates de monitorização específicos para serviços baseados em SysV Init.

4.2 Testes e Validação

Embora não tenha sido implementada uma suíte de testes automatizada, a solução foi submetida a uma validação funcional pragmática em ambiente operacional, focada em quatro pilares:

- Validação da Precisão das Métricas: Verificação manual dos itens de monitorização do Zabbix contra a execução de comandos nativos nos hosts (e.g.: df -h). As falhas detetadas foram validadas de forma cruzada com logs de sistema e com as ferramentas legadas (M/Monit) para assegurar a fiabilidade e prevenir falsos positivos.
- Validação do Ciclo de Alerta e Remediação: Simulação controlada de falhas, incluindo a paragem intencional de serviços (e.g.: systemctl stop nginx) e o esgotamento de recursos (usando stress e fallocate). Validou-se a ativação de triggers, a entrega de notificações (SMS/Chat) e a execução de ações corretivas automáticas (e.g.: reinício de serviço).
- Validação da Integração (Plugin NetBox-Zabbix): Testes focados no fluxo de sincronização, confirmando a comunicação API, a listagem de hosts descobertos na interface do NetBox, a importação correta de dados e o funcionamento das hiperligações de acesso rápido ao Zabbix.

- Validação da Autenticação Centralizada (LDAP): Confirmação de que o acesso às interfaces do NetBox e Zabbix era corretamente gerido pelo diretório LDAP, validando logins bem-sucedidos e a aplicação correta do mapeamento de grupos para os perfis de permissão internos de cada aplicação.

4.3 Avaliação da Solução

A avaliação da solução demonstrou ganhos operacionais e técnicos significativos, validando a abordagem de integração. Os principais resultados incluem:

- Centralização do Inventário: A implementação do NetBox como SSoT eliminou a dispersão e redundância de dados, assegurando um inventário de ativos físicos e lógicos com elevada precisão.
- Automação de Processos: O plugin de integração NetBox-Zabbix automatizou o ciclo de vida dos hosts (criação/atualização) entre o inventário e a monitorização. Adicionalmente, as ações corretivas automáticas no Zabbix reduziram o tempo de resposta a incidentes (MTTR) e a dependência de intervenção manual.
- Monitorização e Resposta a Incidentes: O Zabbix centralizou a monitorização, proporcionando uma plataforma unificada para deteção de falhas, triggers personalizadas e mecanismos de alerta proativos (chat/SMS), minimizando o tempo de inatividade.
- Reforço da Segurança: A integração com um servidor LDAP centralizado para ambas as plataformas (NetBox e Zabbix) unificou a gestão de acessos e permissões, simplificando a administração e reforçando a conformidade com as políticas de segurança.
- Manutenção e Escalabilidade: A adoção de ferramentas open-source robustas e o design modular do plugin garantem a sustentabilidade da solução e facilitam a extensibilidade futura, como a implementação de sincronização automática de ativos.

5 Conclusões

Este trabalho detalhou a implementação de uma solução integrada de gestão de infraestrutura, unificando o NetBox como SSoT para inventário e o Zabbix para monitorização centralizada. Os objetivos do projeto foram alcançados: o NetBox centralizou o inventário de ativos e um plugin NetBox-Zabbix foi desenvolvido para facilitar a migração de dados. A plataforma Zabbix consolidou a monitorização, a alarmística e as ações corretivas automáticas, absorvendo scripts e cronjobs anteriormente fragmentados.

As principais limitações identificadas incluem a ausência de sincronização em tempo real entre as plataformas, a falta de uma configuração de alta disponibilidade (HA) e a inexistência de testes formais. Como trabalho futuro, propõe-se a evolução do plugin para suportar sincronização em tempo real, a implementação de HA e a expansão da solução para o ambiente de data center, acompanhada pelo desenvolvimento de uma suíte de testes formais, que possibilitem uma avaliação robusta.

Referências

1. Data Center Dynamics: The resurgence of DCIM: Navigating the future of data center management.
   Disponível em: https://www.datacenterdynamics.com/en/opinions/the-resurgence-of-dcim-navigating-the-future-of-data-center-management/
2. Zabbix Blog: Zabbix migration in a mid-sized bank environment. Disponível em: https://blog.zabbix.com/zabbix-migration-in-a-mid-sized-bank-environment/13040/
3. anarcat (blog): Replacing Smokeping with Prometheus. Disponível em: https://anarc.at/blog/2020-06-04-replacing-smokeping-prometheus/
4. StackShare: Munin vs Prometheus — What are the differences? Disponível em: https://stackshare.io/stackups/munin-vs-prometheus
5. NetBox Labs: Is a Network Source of Truth Essential for Automation? Disponível em: https://netboxlabs.com/blog/do-you-need-a-source-of-truth-for-network-automation-not-at-first/
6. Zabbix Blog: NetBox as Home CMDB and Integrated with Zabbix. Disponível em: https://blog.zabbix.com/netbox-as-home-cmdb-and-integrated-with-zabbix/29324/
7. DZone: Introduction to Grafana, Prometheus, and Zabbix. Disponível em: https://dzone.com/articles/introduction-to-grafana-prometheus-and-zabbix
8. Syed Asif: Ansible and NetBox — Ansible for Network Automation. Disponível em: https://medium.com/@sydasif78/ansible-and-netbox-896f14d991d5
9. Red Hat: What is Infrastructure as Code (IaC)? Disponível em: https://www.redhat.com/en/topics/automation/what-is-infrastructure-as-code-iac
10. Wang, C., Schwan, K., Talwar, V., Eisenhauer, G., Hu, L., Wolf, M.: A Flexible Architecture Integrating Monitoring and Analytics for Managing Large-Scale Data Centers. In: 8th IEEE International Conference on Autonomic Computing (ICAC 2011), pp. 141–150. IEEE (2011)
11. Gutierrez-Aguado, J., Alcaraz Calero, J.M., Diaz Villanueva, W.: IaaSMon: Monitoring Architecture for Public Cloud Computing Data Centers. *Journal of Grid Computing* 14(2), 283–297 (2016). https://doi.org/10.1007/s10723-015-9357-4
12. Dong, W.: AIOps Architecture in Data Center Site Infrastructure Monitoring. *Computational Intelligence and Neuroscience* 2022, 1988990 (2022). https://doi.org/10.1155/2022/1988990

Session 5B

"Automação Inteligente e Apoio à Decisão com LLMs"

Chair: F. Jorge Duarte **Room:** H207 **Time:** 15:15 - 16:15

Paper	Presenter
Market Research: Um Sistema Automatizado para Prospeção de Clientes usando Web Crawling e Large Language Models	Marco Cunha
Advancing Recruitment: Fair and Efficient Resume Screening with LLMs	Liliana Novais
Differential Diagnosis of Adult Neurodevelopmental Disorders: Systematic Review and Development of a Rule-Based Expert System	Micila Sumaria Medeiros Pereira Magalhães
NutriScan – Nutrition Analysis System	João Serra

Market Research: Um Sistema Automatizado para Prospeção de Clientes usando Web Crawling e Large Language Models

Marco Cunha[1][0009-0004-7097-0923] and Emanuel Silva[2][0000-0002-5046-8942]

[1,2] Instituto Superior de Engenharia do Porto
{1212243,ecs}@isep.ipp.pt

Abstract. A pesquisa de mercado manual para identificar novos clientes business-to-business é um processo demorado e trabalhoso. Este artigo apresenta o design, implementação e avaliação de um sistema automatizado de market research desenvolvido para a SISTRADE, uma empresa de consultoria de software. O sistema visa otimizar a prospeção comercial através da automação das tarefas de pesquisa e análise de potenciais clientes na web. Utiliza técnicas de web crawling para identificar e extrair dados de websites de empresas e integra-se com Large Language Models para classificar automaticamente a sua relevância com base numa pesquisa inicial. A solução adota uma arquitetura Onion e é implementada com recurso a um stack tecnológico moderno, incluindo Vue.js para o frontend, uma camada de integração em .NET/C# e um backend em Python para a lógica de processamento principal, com a persistência de dados a ser gerida pelo SQL Server. O sistema oferece uma interface intuitiva para iniciar pesquisas parametrizadas e rever um histórico completo de resultados. A avaliação através de testes unitários, de integração e de aceitação confirmou que o sistema cumpre com sucesso todos os requisitos, fornecendo uma ferramenta robusta e eficiente para a geração de leads.

Keywords: Market Research, Web Crawling, Web Scraping, Large Language Models, Classificação Automática de Empresas, Inteligência Artificial.

1 Introdução

No cenário empresarial competitivo atual, a capacidade de identificar e atrair novos clientes de forma eficiente é crucial para o crescimento. Para empresas business-to-business (B2B) como a SISTRADE, este processo, conhecido como prospeção comercial, envolve tradicionalmente uma pesquisa manual, que é frequentemente ineficiente e não escala bem com o crescimento exponencial da web. A necessidade estratégica de otimizar este processo levou ao desenvolvimento deste projeto.

O principal problema abordado é a identificação manual de novos clientes B2B. O objetivo foi criar um sistema robusto, capaz de automatizar a recolha, análise e armazenamento de dados relevantes de websites de empresas. Esta abordagem automatizada permite a análise detalhada de websites para identificar oportunidades de negócio,

descobrir informações-chave como contactos e localização da empresa e fortalecer a presença da empresa no mercado. O foco principal deste trabalho é apresentar uma solução arquitetural, mas também a integração de inteligência artificial moderna, especificamente Large Language Models (LLMs), para a classificação e extração de dados, visando aumentar a eficácia das campanhas de marketing da empresa com uma abordagem mais personalizada e estratégica.

2 Estado da arte

O desenvolvimento de um sistema automatizado de prospeção de clientes assenta na interseção de três domínios principais: o Web Crawling para a descoberta de conteúdo, o Web Scraping para a extração de dados, e a Classificação de Relevância para a análise e atribuição de relevância a esses dados. Esta secção apresenta uma análise do estado da arte nestas áreas, contextualizando a abordagem do projeto.

2.1 Descoberta de dados (Web Crawling)

A base de qualquer sistema de análise de mercado na web é a sua capacidade de obter dados de forma autónoma tal como exemplificado na Fig. 1. O Web Crawling é o processo de descoberta, onde um agente automatizado (crawler) navega pela web para encontrar novos conteúdos [1]. Existem várias estratégias, como o Focused Crawling, que direciona a navegação para domínios ou temas específicos, aumentando a pertinência dos dados recolhidos [2]. Outra abordagem também utilizada neste contexto é o Incremental Crawling, que otimiza o processo ao atualizar apenas as páginas que sofreram alterações desde a última visita, reduzindo custos computacionais.

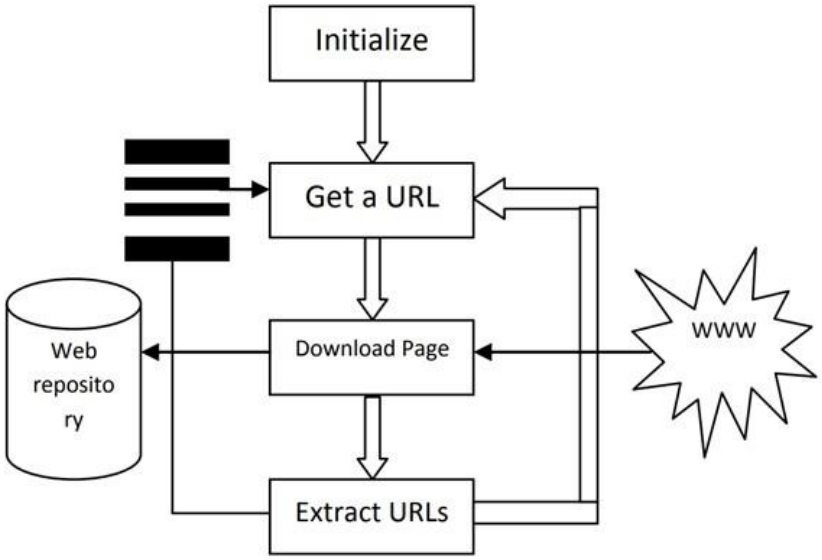


Fig. 1. – Exemplo de um processo de Web Crawling

2.2 Extração de dados (Web Scraping)

Este é o processo de extração de dados não estruturados de páginas web e a sua transformação em formatos estruturados [3]. As técnicas variam, incluindo o DOM Parsing, que utiliza um navegador embutido para interagir com páginas dinâmicas, e o Uso de Expressões Regulares (Regex Matching), que se baseia na identificação de padrões de texto específicos no código da página para extrair os dados desejados [4]. Embora alguns websites disponibilizem APIs que permitem o acesso direto e estável aos seus dados de forma controlada, essas interfaces nem sempre abrangem todas as informações visíveis no site ou podem impor restrições de uso. O web scraping oferece maior flexibilidade, permitindo a extração de dados de praticamente qualquer página da web.

2.3 Classificação de Relevância com Inteligência Artificial

A classificação de registos é o processo que atribui uma pontuação de relevância aos dados. Técnicas como a Aprendizagem Supervisionada, que utiliza algoritmos como Support Vector Machines (SVM) treinados com dados previamente rotulados para aprender padrões, demonstram alta precisão nestes processos.

No entanto, a vanguarda da classificação de textos é atualmente representada pelos LLMs. Utilizando arquiteturas transformer, os LLMs conseguem compreender nuances linguísticas e contextos complexos com uma capacidade sem precedentes [5]. As abordagens baseadas em IA, demonstram resultados consistentemente superiores em precisão e adaptabilidade, quando comparadas com as abordagens mais tradicionais, tornando-se o padrão para sistemas avançados de classificação de registos [4].

2.4 Ferramentas e Trabalhos Relacionados

Existem no mercado ferramentas que abordam partes isoladas deste problema. O BuiltWith[1], por exemplo, é útil para visualizar informações de contacto e outros dados relevantes de uma empresa presentes num website, para além de identificar as tecnologias que este utiliza [6]. Já o Firecrawl [2]especializa-se num workflow de crawling e scraping otimizado para extrair conteúdo limpo, pronto a ser processado por LLMs [7]. O sistema desenvolvido neste projeto diferencia-se por integrar estas várias funcionalidades num workflow coeso e de ponta a ponta, adaptado às necessidades específicas da prospeção de clientes.

2.5 Síntese Comparativa

O Web Crawling trata da descoberta eficiente de páginas, enquanto o Web Scraping se dedica à extração estruturada dos seus dados. A Classificação de Relevância, hoje dominada por modelos de IA e LLMs, avalia a importância desses dados com elevada

[1] https://builtwith.com/
[2] https://www.firecrawl.dev/

precisão. Ferramentas existentes cobrem apenas partes do processo, ao passo que o sistema proposto integra todas as etapas num único workflow.

3 Arquitetura e design do sistema

Para garantir a escalabilidade, manutenibilidade e uma clara separação de responsabilidades, o design do sistema segue os princípios da Onion Architecture. Este estilo arquitetural enfatiza uma abordagem domain-centric, colocando a lógica de negócio principal no centro e garantindo que as camadas externas dependem das camadas internas, e não o contrário [8]. A arquitetura é representada usando uma abordagem do modelo C4 para descrever o sistema em diferentes níveis de abstração.

3.1 Visão de alto nível do sistema

A um nível alto de granularidade (C4 Nível 2), o sistema é decomposto em quatro containers principais, como mostrado na Fig. 2. Esta estrutura isola responsabilidades e facilita o desenvolvimento e deployment independentes.

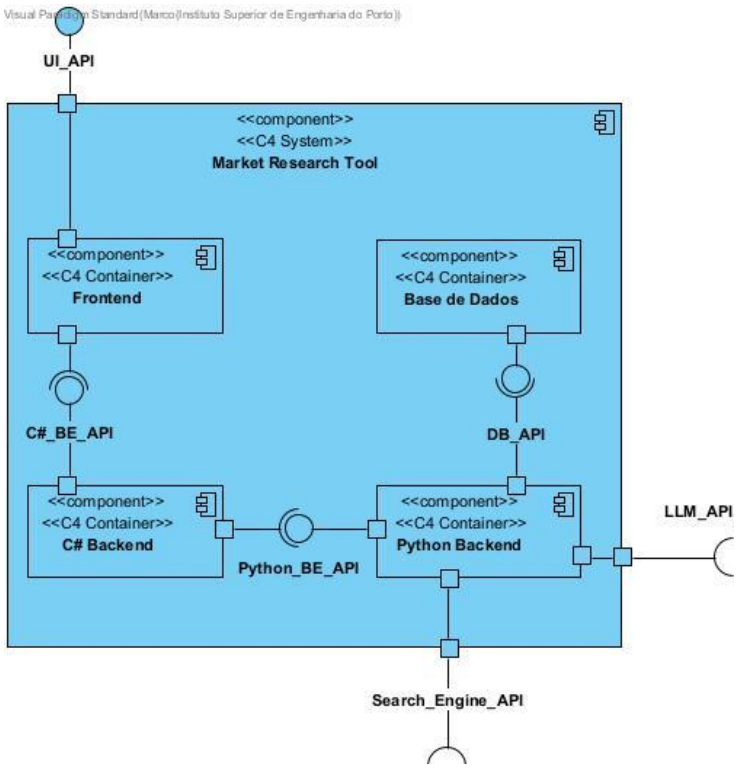


Fig. 2. - Arquitetura de Alto Nível do Sistema

Frontend: Uma single-page application (SPA) em Vue.js que fornece a interface gráfica para os utilizadores configurarem e iniciarem pesquisas, bem como visualizarem os resultados.

C# Backend: Este container atua como uma camada de passthrough ou ponte. Recebe pedidos do frontend e encaminha-os para o backend Python. Este design foi um requisito específico para garantir a compatibilidade e integração futuras com o sistema ERP existente da empresa, baseado em .NET.

Python Backend: Este é o núcleo do sistema, responsável por toda a lógica de processamento principal. Gere os pedidos de pesquisa iniciais, executa os processos de web crawling e scraping, comunica com APIs externas (Search Engines, LLMs) e interage com a base de dados.

Base de Dados: Uma instância do Microsoft SQL Server é usada para a persistência de dados. Armazena toda a informação relacionada com as pesquisas, incluindo os parâmetros definidos pelo utilizador, o estado e os dados extraídos e classificados das empresas.

3.2 Lógica Principal: Python Backend

O Backend Python está estruturado internamente de acordo com as camadas da Onion Architecture (Fig. 3), garantindo que a lógica principal permaneça independente de frameworks e tecnologias externas.

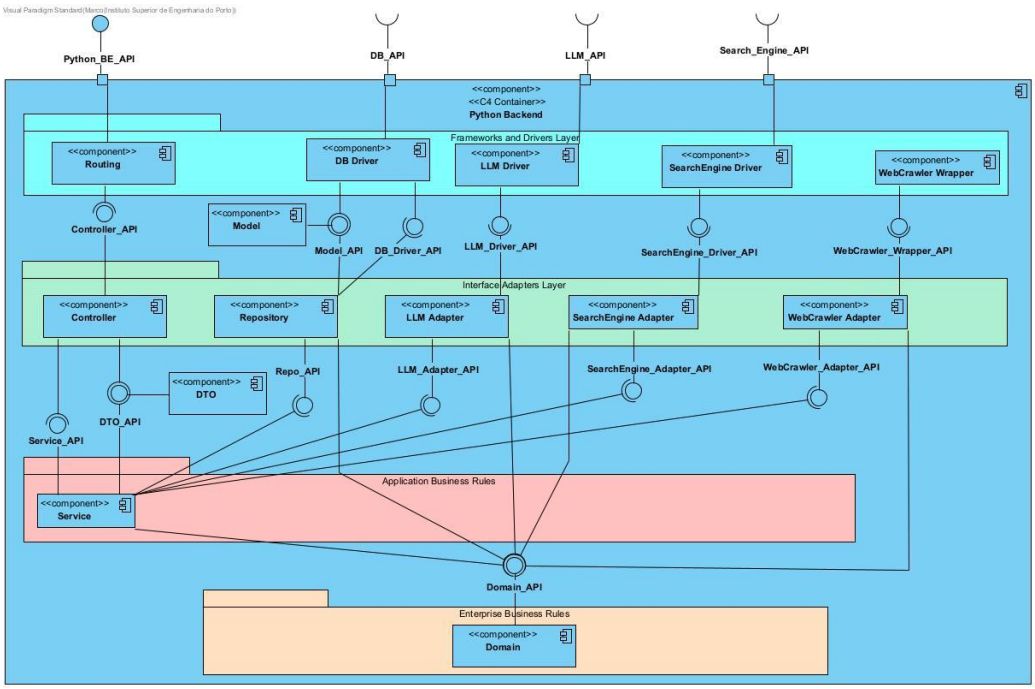


Fig. 3. - Arquitetura em Camadas do Backend Python

Enterprise Business Rules (Dominio): No núcleo encontram-se as entidades de domínio, que encapsulam a lógica de negócio e as regras essenciais da aplicação.

Application Business Rules (Serviços): Esta camada contém os serviços específicos da aplicação que orquestram as funcionalidades, como iniciar uma nova pesquisa ou

obter o histórico de pesquisas. Depende da camada de domínio e define interfaces para as camadas externas.

Interface Adapters: Esta camada adapta os dados do formato mais conveniente para as agências externas (como a base de dados ou APIs web) para o formato mais conveniente para as camadas internas (domínio e serviços). Inclui controladores, repositórios e adaptadores para serviços externos como o LLM e o web crawler.

Frameworks and Drivers: A camada mais externa consiste nas tecnologias que interagem com o mundo exterior, como a web framework, os drivers da base de dados e os wrappers para bibliotecas externas como o Scrapy.

4 Metodologia de implementação

A funcionalidade do sistema é construída em torno de um workflow de várias etapas que transforma uma simples query do utilizador numa lista de potenciais clientes classificados. Este processo é suportado por um stack tecnológico cuidadosamente selecionado.

4.1 Workflow de Aquisição e processamento de Dados

O processo principal começa quando um utilizador inicia uma pesquisa parametrizada e termina com o armazenamento dos dados classificados. As etapas-chave são:

- Obtenção Inicial de URLs: Com base na query do utilizador (ex: "empresas de artes gráficas em Portugal"), o sistema consulta um ou mais motores de busca (ex: Google, Bing) para obter uma lista de URLs iniciais. Isto é conseguido através de APIs oficiais ou simulando a interação de um utilizador;
- Web Crawling: O sistema usa a framework Scrapy para realizar um crawl focado a partir das URLs iniciais. O crawler é configurado para navegar dentro dos domínios permitidos e identificar páginas relevantes, como páginas "Sobre Nós" e "Contactos", que provavelmente contêm informações empresariais valiosas;
- Extração de Dados: Uma vez identificadas as páginas relevantes, o sistema faz o scrape do seu conteúdo para extrair dados não estruturados, incluindo nome da empresa, emails de contacto, números de telefone, moradas e texto descritivo;
- Classificação baseada em LLM: A informação extraída de cada website é compilada e enviada para um LLM interno da empresa através de uma API. O LLM é instruído a analisar o conteúdo no contexto da query original do utilizador e a devolver uma pontuação de relevância (0-100), juntamente com dados limpos e estruturados (ex: uma lista de emails válidos);
- Persistência de dados: Os dados processados e classificados de cada empresa são armazenados na base de dados e ligados ao histórico da pesquisa inicial para recuperação posterior.

4.2 Stack Tecnológico

A seleção das tecnologias baseou-se tanto no ecossistema existente na SISTRADE como nos requisitos específicos do projeto.

Frontend: O Vue.js foi escolhido pela sua baixa curva de aprendizagem e arquitetura baseada em componentes, ideal para criar interfaces de utilizador reativas e de fácil manutenção.

Backend: Foi utilizada uma abordagem com dois backends. O .NET (C#) serve como uma ponte segura para os sistemas internos da empresa. O Python foi selecionado para a lógica principal devido ao seu vasto ecossistema de bibliotecas para web crawling e análise de dados, destacando-se a framework Scrapy.

Base de Dados: O Microsoft SQL Server foi utilizado por ser o sistema de gestão de base de dados padrão na empresa.

5 Resultados e Avaliação

O sistema implementado fornece uma interface web totalmente funcional e foi rigorosamente testado para garantir que cumpria todos os objetivos do projeto.

5.1 Interface e Funcionalidade do Sistema

O frontend apresenta aos utilizadores duas funcionalidades principais: (1) iniciar uma nova pesquisa e (2) visualizar o histórico de pesquisas anteriores. O painel de pesquisa (Fig. 4) permite aos utilizadores definir a query de pesquisa e parâmetros como a profundidade do crawl e que motores de busca utilizar.

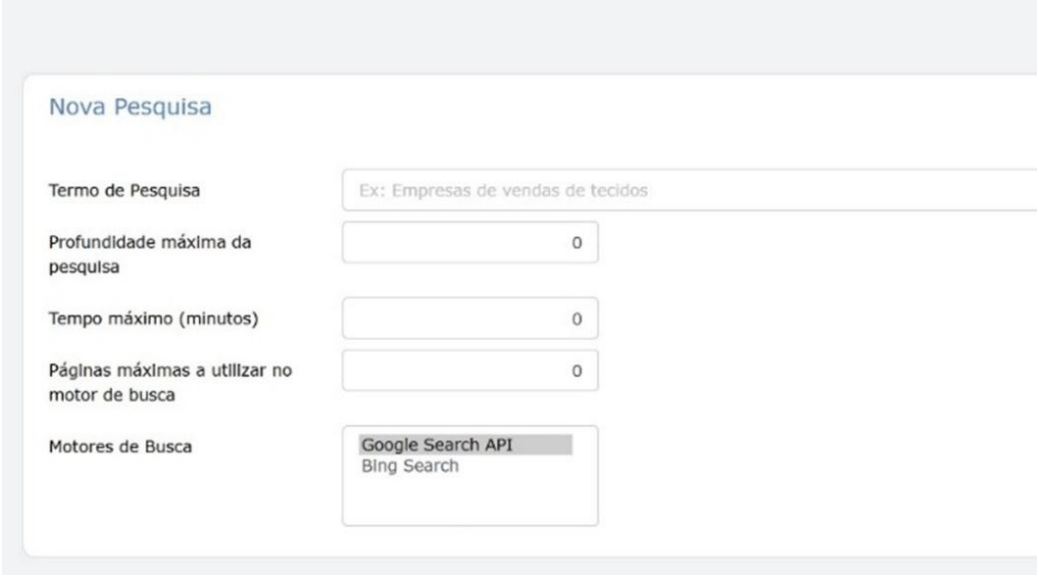


Fig. 4. - Interface do Utilizador (nova pesquisa)

Assim que a pesquisa é concluída, os resultados são exibidos na secção de histórico. Os utilizadores podem ver uma lista das empresas encontradas, cada uma com a sua pontuação de relevância calculada (Fig. 5). Clicar num resultado revela informações detalhadas, incluindo o URL do website da empresa, emails, números de telefone e moradas extraídos. Esta interface também inclui opções de filtragem avançada para o histórico de pesquisas.

Fig. 5. - Interface do Utilizador (histórico de pesquisa)

5.2 Validação do Sistema

A qualidade e correção do sistema foram validadas através de uma estratégia de testes multinível:

- Testes Unitários: Foram escritos testes automatizados em Python para verificar componentes individuais de forma isolada, alcançando uma cobertura de código de 94%. Foram usados Mocks para simular dependências externas, como a base de dados e a API do LLM;
- Testes de Integração: A ferramenta Postman foi utilizada para testar as interações entre o frontend, o backend C# e o backend Python, garantindo que os endpoints da API se comportavam como esperado sob várias condições (ex: filtros combinados);
- Testes de Aceitação: Foram realizadas sessões de teste semanais com os supervisores da empresa para validar que a funcionalidade da aplicação cumpria os requisitos de negócio e as expectativas dos utilizadores.

A avaliação concluiu que todos os objetivos pré-definidos foram totalmente alcançados. O sistema demonstrou um desempenho eficiente e fiável, integrando com sucesso todas as tecnologias especificadas.

6 Conclusões e trabalho futuro

Este artigo apresentou um sistema automatizado de pesquisa de mercado projetado para otimizar a prospeção de clientes. O desenvolvimento e a implementação foram bem-sucedidos, resultando numa aplicação funcional que responde às necessidades identificadas pela SISTRADE. O projeto demonstra a viabilidade técnica da combinação de técnicas de web crawling com Large Language Models para criar ferramentas robustas de business intelligence.

Uma limitação significativa do sistema atual é a sua dependência de uma API de LLM interna que só consegue processar um pedido em paralelo. Isto limita a escalabilidade do sistema, tornando-o adequado principalmente para uso individual, em vez de um uso concorrente e intensivo.

O desenvolvimento futuro focar-se-á em melhorar a experiência do utilizador e a eficiência do sistema. As melhorias planeadas incluem a implementação de uma funcionalidade para interromper pesquisas em andamento; permitir que os utilizadores refaçam uma pesquisa em websites específicos; adicionar uma funcionalidade para gerar automaticamente rascunhos de emails de contacto com base no termo de pesquisa e nos dados da empresa; e introduzir um sistema de notificação para alertar os utilizadores quando pesquisas mais longas terminarem. Estas melhorias tornarão a aplicação mais interativa, escalável e alinhada com as necessidades dinâmicas dos seus utilizadores.

Agradecimentos. O autor gostaria de agradecer à SISTRADE - Software Consulting, S.A. pela oportunidade de estágio e por proporcionar um ambiente acolhedor e à equipa de supervisão, pela sua orientação inestimável, feedback e apoio fundamental ao longo deste projeto.

References

1. Khder, M.: Web Scraping or Web Crawling: State of Art, Techniques, Approaches and Application. International Journal of Advances in Soft Computing and its Applications 13(3), 145–168 (2021)
2. M. Ahuja, J. Singh, and V. nica, "Web Crawler: Extracting the Web Data," *International Journal of Computer Trends and Technology*, vol. 13, pp. 132–137, Jul. 2014, doi: 10.14445/22312803/IJCTT-V13P128.
3. Sandeep Shreekumar, Satyan Mundke, and Murlidhar Dhanawade, "IMPORTANCE OF WEB SCRAPING IN E- COMMERCE BUSINESS," *NCRD's Technical Review : e-Journal*, vol. 7, no. 1, 2022.
4. Manushi Weerasinghe, "Enhancing Web Scraping with Artificial Intelligence," 2024.
5. Kostina, A., Dikaiakos, M., Stefanidis, D., Pallis, G.: Large Language Models For Text Classification: Case Study And Comprehensive Review. arXiv:2501.08457 (2025)
6. BuiltWith Technology Lookup. https://builtwith.com/, last accessed 2025/10/21
7. Firecrawl Playground. https://www.firecrawl.dev/playground, last accessed 2025/10/21
8. Martin, R.C.: Clean Architecture: A Craftsman's Guide to Software Structure and Design. Prentice Hall (2017)

Advancing Recruitment: Fair and Efficient Resume Screening with LLMs

Liliana Novais[1,2,3,4 [0000-0002-2347-6779]], Vitor Rocio[1,4 [0000-0002-3314-898X]], Paulo Moura Oliveira[4,6 [0000-0003-4283-1243]] and Arnaldo Santos[1,4,5 [0000-0001-5139-6728]]

[1] Universidade Aberta, Lisboa, Portugal
[2] Instituto Superior de Engenharia do Porto Porto, Portugal
[3] SEG Automotive Portugal, Vila do Conde, Portugal
[4] INESC TEC, Porto, Portugal
[5] Digimedia, Aveiro, Portugal
[6] Universidade de Trás-os-Montes e Alto Douro, Vila Real, Portugal
2302227@estudante.uab.pt

Abstract. Traditional talent acquisition encounters significant sustainability problems marked by increasing application quantities and the intrinsic dangers of subjective human bias. This article, which is still being worked on, uses the Design Science Research (DSR) technique to create CurAIEval, a new IT tool for automated, fairness-aware resume screening. Based on a rigorous examination of the literature, the study finds a fundamental gap between technical optimization and ethical governance. To close this gap, the suggested architecture combines Large Language Models (LLMs) for semantic analysis with a required "Fairness-by-Design" auditing engine. The system uses a Human-in-the-Loop (HITL) protocol to automate parsing of large amounts of data while still letting people make the ultimate judgments. Expected outcomes encompass the creation of a workable prototype that substantially alleviates administrative burdens and the validation of a Balanced KPI Framework. This approach is meant to quantify both operational efficiency (like time-to-hire) and algorithmic fairness (like disparate impact) at the same time. It gives a validated technique to automate hiring in a responsible and fair way.

Keywords: LLM-Based Recruitment, Algorithmic Fairness, Resume Classification, Design Science Research.

1 Introduction

Strategic management of human capital is a key part of an organization's performance and a major source of its competitive edge [6]. Talent acquisition, or the capacity to find, attract, and hire talented people, is at the heart of this. But the first step of screening and choosing candidates is still a major problem in modern Human Resource Management (HRM). This process is sometimes very time-consuming, expensive, and manual in many companies, which means that highly skilled HR professionals must spend too much time on routine duties instead of strategic ones[8].

The digitization of job platforms and the trend to remote work after the pandemic have made this problem worse by making talent pools more global and increasing the number of applications by a huge amount. For example, SEG Automotive Portugal gets over 170 applications for each job opening, which means that it can take up to two months to choose one [15]. These kinds of delays cost a lot of money and make it more likely that you'll lose top applicants to competitors that can move faster [5].

In addition to being inefficient, manual screening also has big legal and moral risks. The procedure is quite open to unconscious bias from people, and recruiters who are under a lot of pressure may use cognitive shortcuts, which can lead to unfair results based on aspects that don't have anything to do with merit [19]. Recent progress in Artificial Intelligence (AI), particularly Large Language Models (LLMs) developed from transformer architectures [22], provides strong tools for semantic analysis and automated screening. However, the uncritical adoption of these technologies poses a "dual-use dilemma": while they provide scalability, they also risk automating and exacerbating historical biases if not meticulously regulated [13].

This study used the Design Science Research (DSR) approach [10, 17] to introduce CurAIEval, a socio-technical IT artifact, in response to the identified challenge. This approach is meant to improve hiring by using automation to make things more efficient and built-in fairness rules to make sure everyone is treated fairly.

This paper discusses the first steps of the DSR cycle, which include finding the problem and making a rough architectural plan for the CurAIEval artifact. The proposed design is based on a thorough systematic examination of the literature, although it is still a work-in-progress conceptual model. The next steps in the DSR process will be to technically implement the prototype and show it in a real-life high-volume recruitment setting. In the end, the research will end with a thorough empirical evaluation utilizing a "Balanced KPI Framework" to see how well the system can cut down on time-to-hire while also limiting disparate impact [12]. This will show that the artifact is useful in real-world situations.

2 Research Methodology

This article presents a work-in-progress whose primary objective is to design, develop, and validate a novel IT artifact to solve a practical organizational problem. Therefore, the Design Science Research (DSR) methodology was adopted [17]. DSR is uniquely suited for this work, as it focuses on creating and evaluating innovative artifacts that address real-world challenges while contributing to the theoretical knowledge base [10].

The Main Research Question (MRQ) is, '*How can contemporary AI systems enhance recruitment processes by simultaneously addressing efficiency constraints, evaluator subjectivity, HR workload, and unconscious biases while ensuring equitable selection of optimally qualified candidates?'*

To perform a systematic literature review, the MRQ is operationalized through four specific Research Questions (RQs) that the artifact must address:

- RQ1 (Frameworks): What implementation frameworks enable effective deployment of LLMs and chatbots for automated resume screening while maintaining alignment with organizational-specific hiring criteria?
- RQ2 (Balancing): Which automation strategies (e.g., scoring thresholds, human-in-the-loop protocols) most effectively balance HR workload reduction with candidate selection quality?
- RQ3 (Ethical Design): What methodological approaches best identify and mitigate unconscious biases in AI-assisted recruitment to promote diversity and equitable candidate evaluation?
- RQ4 (Metrics): Which quantitative metrics (KPIs) most accurately assess both procedural efficiency and selection fairness in AI-enhanced recruitment systems?

3 Research Contributions

This study provides unique insights at the convergence of Human Resource Management (HRM) and Artificial Intelligence (AI). Utilizing the Design Science Research (DSR) methodology [10, 17], these contributions are classified into completed contributions—originating from the problem explication and the systematic literature review—and anticipated contributions—to be achieved through the design, demonstration, and evaluation of the proposed artifact.

3.1. Completed Contributions: Foundational and Analytical

The preliminary research phase yielded three key analytical outcomes. First, the SLR identified a critical "Structural Gap" between technical optimization frameworks focused on efficiency [18] and ethical governance models centered on disability justice [14, 21]. Second, the study synthesized a scholarly consensus for Human-in-the-Loop (HITL) protocols, restricting AI to high-volume parsing while retaining human decision authority [13, 23]. Finally, we defined "Fairness-by-Design" architectural principles, including pre-processing data sanitization [1] and the integration of allocational harm metrics like RABBI directly into scoring pipelines[9].

3.2. Expected Contributions: Practical and Methodological

The subsequent DSR phases will deliver three tangible contributions:

- **The CurAIEval Artifact:** A functional prototype implementing "Fairness-by-Design" principles. Unlike theoretical models, it technically decouples fairness audits from semantic matching, offering a scalable framework for bias-aware screening that resolves manual scaling challenges [2].
- **Verified "Balanced KPI" Framework:** Addresses the deficiency in evaluation metrics [12] by integrating operational efficiency (e.g., time-to-hire) with Selection Fairness (e.g., disparate impact). This dashboard enables practitioners to assess the long-term effects of automation, as suggested by [11].

- **Empirical Verification:** Validates the artifact in a high-volume industrial setting. This case study provides a framework for addressing the socio-technical friction between algorithmic efficiency and organizational culture in HR.

4 Literature Review & Problem Justification

A full Systematic Literature Review (SLR) was conducted to precisely delineate the objectives for the proposed solution (DSR Step 2). This research delineated the present condition of AI in recruitment while also critically examining the shortcomings of current commercial and academic solutions, so underscoring the need for an innovative artifact.

4.2. The Systematic Review Process and Bibliometric Analysis

The review adhered to the PRISMA 2020 framework to guarantee methodological transparency and reproducibility [16]. The search technique focused on five leading academic databases: Scopus, Web of Science, IEEE Xplore, ACM Digital Library, and Semantic Scholar. The search parameters were strictly limited to peer-reviewed literature published from January 2020 to January 2025. This temporal limitation was intentional, aimed at encapsulating the paradigm shift brought about by the transformer architecture and the post-pandemic digital transformation of HR. A multi-stage screening approach, which included automated deduplication and manual quality assessment, reduced an initial pool of 735 records to a final corpus of 83 high-quality research.
A bibliometric analysis of keyword co-occurrence was performed to illustrate the intellectual structure of this area (see Fig. 1).

Fig. 1. Keywords Associations

Fig. 1 demonstrates that the network analysis uncovers distinct groupings of information that infrequently engage with one another. The visual distinction between the technical clusters (centered on "Machine Learning" and "Optimization") and the sociotechnical clusters (centered on "Algorithmic Fairness" and "Ethics") offers actual proof of the disciplinary silos obstructing the area.

4.2. The "Structural Gap" and Its Causes

The synthesis of these clusters unveiled a significant bifurcation, termed the "Structural Gap." The research landscape is distinctly polarized:

1. **Technical Optimization (The "Green" Cluster):** Research concentrated on mathematical programming and predictive analytics to enhance efficiency and minimize costs [18].
2. **Ethical Governance (The "Red" Cluster):** Research concentrated on normative ethics, disability justice, and the sociological implications of automation (Tilmes, 2022; Mujtaba, 2019)[14, 21].

This analysis indicates that this disparity arises from underlying disciplinary differences. Computer science research frequently regards justice as a quantitative restriction to be fulfilled within a cost function, while social science research challenges the inherent validity of assessing human potential, sometimes failing to provide technical alternatives. As a result, solutions are often designed to be either efficient or equitable, but seldom at the same time.

4.3. The Imperative for a Novel Artifact vs. Adaptation

A crucial inquiry emerges in Design Science Research: Why create a novel artifact (CurAIEval) instead of modifying pre-existing commercial technologies (e.g., LinkedIn Recruiter, standard ATS)? The literature presents three compelling arguments against adaptation:

- **The "Black Box" Problem:** Commercial products that are already on the market are "black boxes" that only work with certain software. Their algorithms are hard to understand, so it's hard to check their decision-making or see if they are making previous biases worse [24]. For a serious academic study, there needs to be a "white box" architecture that lets you see and change how efficiency (LLM embeddings) and ethics (fairness restrictions) work together.
- **Reactive vs. Proactive Design**: Many commercial solutions see fairness as something to fix after the fact or as a UI feature. The research, on the other hand, says that "Fairness-by-Design" is needed to effectively reduce bias. This means putting protections deep into the data input and vectorization layers [1]. This basic architectural need can't be added to old systems that were only developed for speed.
- **The Metrics Deficit:** Right now, most systems are just good at "Time-to-Hire." There is a clear shortage of solutions that natively incorporate balanced dashboards that show both "Allocational Harm" and processing speed[9]. Making CurAIEval lets you test these new, balanced measures in a safe setting.

4.4. Robust Fairness: Beyond Legal Compliance

Finally, the evaluation requires a more nuanced definition of "Fairness" than what is already offered in conventional hiring instruments. Early methods used "Fairness

through Unawareness," which meant getting rid of protected class labels like "gender." This didn't work because of proxy bias, which happens when models learn to discriminate based on features that are related, like zip codes, language patterns, or university affiliations [4].

The CurAIEval artifact is thus required by the necessity to progress beyond just legal compliance to Algorithmic Equity. This means dealing with Allocational Harm, which is when certain populations are systematically left out of economic opportunities. The literature necessitates systems that implement active mitigation strategies, including counterfactual testing (evaluating how a resume would be ranked if the gender were altered) and intersectional analysis (ensuring equity for subgroups, such as Black women, rather than merely broad categories)[3]. Because "off-the-shelf" LLMs have built-in biases, the only way to safely deploy them is to make a custom artifact with a "Fairness Engine."

5 Proposed Artifact: *CurAIEval* Objectives

The CurAIEval system is the main IT tool for this project. It was made to turn the theoretical ideas found in the literature review into a working solution. Five interconnected goals set in the problem-framing phase influence its development:

1. Bias Mitigation (RQ3): Fight implicit bias by making algorithms that put job-related skills first, which will lead to fair hiring.
2. Process Optimization (RQ1/RQ2): Use LLMs for semantic matching to improve prediction validity beyond just using keywords
3. Strategic Repositioning (RQ2): Automate low-value administrative tasks so that HR professionals can focus on high-value interactions.
4. Performance Transparency (RQ4): Set up balanced Key Performance Indicators (KPIs) that look at both how well the process works (like time to hire) and how fair it is (like disparate impact).
5. Ethical Compliance: Make sure that algorithms follow the law so that the system can be audited and defended.

5.1. Artifact Architecture and Design

CurAIEval is designed as a modular socio-technical system that is run by "Fairness-by-Design" to make these goals happen. Instead of using retroactive ethical patches, fairness constraints are built into the architecture itself. We used Unified Modeling Language (UML) to model the system and show three important points of view.

5.1.1. Functional View: Operationalizing Human Sovereignty

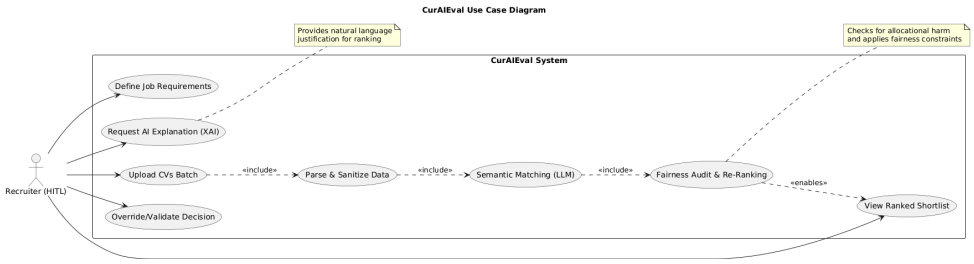


Fig. 2. Use Case Diagram for CurAIEval

The functional design (Fig. 2) rejects full automation in favor of a Human-in-the-Loop (HITL) protocol, aligning with the consensus that AI lacks the nuance required for final hiring decisions. The recruiter initiates automated tasks but retains sole authority over validation. Crucially, the "Fairness Audit" is modeled as a mandatory <<include>> relationship, preventing the bypass of bias checks for speed. Additionally, an "Explainability Request" use case addresses "black box" trust issues by providing natural language justifications for scoring.

5.1.2. Structural View: Decoupling Ethics from Intelligence

The Component Diagram (Fig. 3) talks about scalability by architecturally separating the Matching Service from the LLM API. This means that backend model modifications (such as going from GPT-3.5 to LLaMA) may be made without having to change the whole system. Most importantly, the Fairness Engine is set apart as an independent auditor. This engine maintains mathematical fairness restrictions (such as demographic parity) on the output, no matter how the semantic model behaves. This is a hard stop against allocational harm, as suggested by Delecraz et al. and Zhou[7, 25].

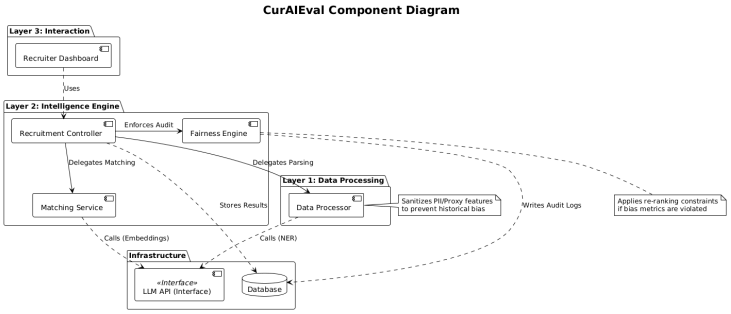


Fig. 3. Component Diagram for CurAIEval

5.1.3. Behavioral View: The Pre-Emptive Fairness Loop

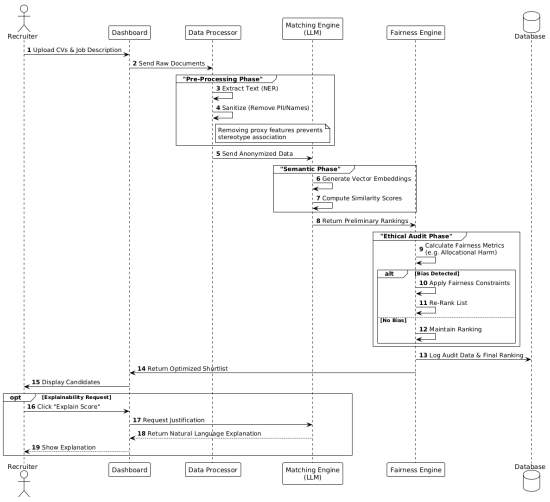


Fig. 4. Sequence Diagram for CurAIEval

The Sequence Diagram (Fig. 4) shows how data flows to get rid of the risk of "proxy bias." Before semantic embedding, a Sanitization stage removes PII (names, gender markers) to keep the LLM from encoding preconceptions. The system also uses the re-ranking logic that Gan et al. suggested. If the Ethical Audit Phase finds that a protected group is being unfairly left out, the Fairness Engine automatically starts re-ranking algorithms. This makes sure that efficiency never comes at the expense of fairness, which is what balanced metrics need.

6 Discussion and Conclusion

This study pinpoints a significant operational and ethical impediment in contemporary Human Resource Management (HRM). As established, traditional manual recruiting is increasingly unsustainable due to the globalization of talent pools and the sheer number of digital applications, resulting in significant administrative expenses and lengthy timescales [5]. At the same time, manual screening is still legally risky and open to unconscious human biases that undermine the fairness of the selection process [8]. This study, using a comprehensive Systematic Literature Review (SLR), identified a notable "structural gap" in the current academic landscape: a division between research solely concentrating on technical optimization [18] and research addressing ethical governance [21]. There are not many alternatives right now that are designed to be both operationally efficient and verifiably fair.

This work introduces CurAIEval, an IT artifact created through the Design Science Research approach [10], to address this gap. CurAIEval is not a "black box" automation tool; instead, it is thought of as a socio-technical system that follows a "Fairness-by-Design" philosophy. The system's goal is to automate high-volume parsing and semantic matching while tightly protecting human sovereignty for ultimate, context-aware decision-making by following the academic agreement on Human-in-the-Loop (HITL) protocols[13, 23]. The suggested architecture radically changes the role of the HR

professional from an administrative data processor to a strategic partner. This lowers legal risks and makes the hiring process more accurate[20].

Preliminary Design and Limitations: It is vital to note that the architectural design shown in this work is still in the early stages. The conceptual model does a good job of separating the "Fairness Engine" from the "Semantic Matcher" such that efficiency doesn't take precedence over fairness. However, the artifact has not yet been fully coded or put into use in a real setting. The current design is based on theoretical validation from the SLR instead of real-world data from field testing. As a result, the proposed algorithmic interventions, such as preprocessing data sanitization and postprocessing re-ranking, are still only theoretical ideas that need to be tested against real-world data noise and organizational complexity.

Work in the Future: The next steps in the Design Science Research cycle, Demonstration and Evaluation, will shape the direction of this research. The next step will be to work on the technical side of the CurAIEval prototype, going from architectural plans to a working piece of software that can work with current HR information systems. We will use this prototype in the real-world setting of our industrial partner, SEG Automotive Portugal, to test high-volume screening scenarios.

The most important next step is to use the proposed Balanced KPI Dashboard to test the artifact in real life. Future study will meticulously evaluate the system's effectiveness not just against efficiency criteria, such as time-to-hire and cost reduction, but also against fairness measurements, including the Disparate Impact Ratio and Allocational Harm indices [9]. Future studies will ascertain whether these automated outcomes effectively diminish the "metrics gap" identified by [12] by contrasting them with a manual baseline, thereby establishing a validated framework for the responsible and equitable implementation of large language models in recruitment.

References

1. Abdelhalim, E.: A framework of diversity, equity, and inclusion safeguards for chatbots. 67, 5, (2024). https://doi.org/10.1016/j.bushor.2024.03.003.
2. Alzyoud, A.A.Y.: Navigating the Future: The Role of Artificial Intelligence in Shaping Recruitment Practices. 2024 ASU Int. Conf. Emerg. Technol. Sustain. Intell. Syst. ICETSIS. (2024). https://doi.org/10.1109/ICETSIS61505.2024.10459627.
3. Armstrong, L. et al.: The Silicon Ceiling: Auditing GPT's Race and Gender Biases in Hiring - Proceedings of the 4th ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization. Proc. 4th ACM Conf. Equity Access Algorithms Mech. Optim. (2024). https://doi.org/10.1145/3689904.3694699.
4. Barriuso, M.I.: Unveiling Gender Biases in Recruitment: A Natural Language Processing Approach. J. Glob. Econ. Manag. Bus. Res. (2024). https://doi.org/10.56557/jgembr/2024/v16i18634.
5. Cappelli, P.: The practical utility of the latest research on the time to hire. Hum. Resour. Manag. J. 19, 4, 385–397 (2009).
6. Cascio, W.F.: Managing Human Resources: Productivity, Quality of Work Life, Profits. McGraw-Hill Education (2019).

7. Delecraz, S. et al.: Making Recruitment More Inclusive: Unfairness Monitoring With A Job Matching Machine-Learning Algorithm - 2022 IEEE/ACM INTERNATIONAL WORKSHOP ON EQUITABLE DATA & TECHNOLOGY (FAIRWARE 2022). 2022 IEEEACM Int. Workshop EQUITABLE DATA Technol. FAIRWARE 2022. 34–41 (2022). https://doi.org/10.1145/3524491.3527309.
8. Dessler, G.: Human Resource Management. Pearson Education (2020).
9. Gan, C. et al.: Application of LLM Agents in Recruitment: A Novel Framework for Resume Screening. ArXiv Prepr. (2024).
10. Hevner, A.R. et al.: Design science in information systems research. MIS Q. 28, 1, 75–105 (2004).
11. Hu, Q.: Unilever's Practice on AI-based Recruitment. Highlights Bus. Econ. Manag. (2023). https://doi.org/10.54097/hbem.v16i.10565.
12. Joshi, A. et al.: Strategic Adoption of Artificial Intelligence for Human Resource Management Practices Transforming Healthcare Sector. 3, 3, 151–163 (2024). https://doi.org/10.58818/ijems.v3i3.133.
13. Lukaszewski, K.M.: Will the use of AI in human resources create a digital Frankenstein? Organ. Dyn. 53, 1, (2024). https://doi.org/10.1016/j.orgdyn.2024.101033.
14. Mujtaba, D.F.: Ethical Considerations in AI-Based Recruitment. 019 IEEE Int. Symp. Technol. Soc. ISTAS. 2019, (2019). https://doi.org/10.1109/ISTAS48451.2019.8937920.
15. Novais, L.: CurEval - Curriculum Evaluation. Instituto Superior de Engenharia do Porto (2023).
16. Page, M.J. et al.: The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. BMJ. 372, n71 (2021). https://doi.org/10.1136/bmj.n71.
17. Peffers, K. et al.: A design science research methodology for information systems research. J. Manag. Inf. Syst. 24, 3, 45–77 (2007).
18. Pessach, D.: Employees recruitment: A prescriptive analytics approach via machine learning and mathematical programming. Decis. Support Syst. 134, (2020). https://doi.org/10.1016/j.dss.2020.113290.
19. Roxburgh, E., Hansen, K.: Bias in Recruitment and Selection. Cent. Labour Employ. Work. 0, 1–3 (2015).
20. Schmidt, F.L., Hunter, J.E.: The validity and utility of selection methods in personnel psychology: Practical and theoretical implications of 85 years of research findings. Psychol. Bull. 124, 2, 262–274 (2016).
21. Tilmes, N.: Disability, fairness, and algorithmic bias in AI recruitment. Ethics Inf Technol. 24, (2022). https://doi.org/10.1007/s10676-022-09633-2.
22. Vaswani, A. et al.: Attention Is All You Need. Adv. Neural Inf. Process. Syst. NIPS. 30, 5998–6008 (2017).
23. Vivek, R.: Enhancing Diversity and Reducing Bias in Recruitment Through AI: A Review of Strategies and Challenges. 2, 4, 0101–0118 (2023). https://doi.org/10.47813/2782-5280-2023-2-4-0101-0118.
24. Waller, M. et al.: Bias Mitigation Methods: Applicability, Legality, and Recommendations for Development. J. Artif. INTELLIGENCE Res. 81, 1043–1078 (2024).
25. Zhou, R.: Empirical Study and Mitigation Methods of Bias in LLM-Based Robots. Acad. J. Sci. Technol. (2024). https://doi.org/10.54097/re9qp070.

Differential Diagnosis of Adult Neurodevelopmental Disorders: Systematic Review and Development of a Rule-Based Expert System

Micila Sumaria Medeiros Pereira Magalhães[1], Rodrigo Bernado Domingos Rodrigues[2], Mário António Afonso Cardoso[3], Fábio José Dias Girão[4], and Gonçalo Fernandes[5]

ISEP, Politécnico do Porto, Rua Dr. António Bernardino de Almeida, 4249-015, Porto, Portugal
{1231812,1250549,1191568,1201146,1201496}@isep.ipp.pt

Abstract. Neurodevelopmental Disorders (NDDs) in adults — most notably Autism Spectrum Disorder (ASD) and Attention-Deficit/ Hyperactivity Disorder (ADHD) — remain challenging to diagnose due to overlapping symptoms and comorbidities. This study examines artificial intelligence (AI) approaches applied to adult DDx, which converge on two main directions: rule-based systems for workflow management (CDSS) and machine-learning (ML) models aimed at efficient triage and classification. Based on this analysis, we propose a transparent, rule-based expert system for adults, complemented by optional ML components to support prioritization within a dual-professional workflow (Neurology + Psychology). The system seeks to improve diagnostic consistency and explainability while maintaining clinical oversight and ensuring full decision traceability. The contribution is an explainable and pragmatic model that combines expert rules with lightweight ML to provide reliable and auditable decision support for adult NDD DDx.

Keywords: Neurodevelopmental Disorders (NDD); Differential Diagnosis (DDx); Rule-Based Systems; Business Process Management (BPM); Clinical Decision Support Systems (CDSS); Explainability (XAI); Machine Learning (ML); Adults.

1 Introduction

The adult DDx of NDDs—notably autism spectrum disorder (ASD) and attention-deficit/hyperactivity disorder (ADHD) — is challenging due to symptom overlap, comorbidities, and heterogeneous developmental trajectories. Consequently, there is demand for approaches that standardize the diagnostic pathway, shorten time to diagnosis, and support decisions with traceability and clinical explainability. Recent literature converges on two broad families of AI support for adult DDx: (i) rule-based clinical decision support systems (CDSS) integrated with business process management (BPM), which operationalize staged workflows (e.g., triage →

interviews/tests → consensus meeting → report) and generate automatic summaries; and (ii) machine-learning (ML) pipelines that select minimal variable subsets (instrument items, neuropsychological tests, social cognition measures) for classification and case prioritization [1–5].
In practice, a rule-based CDSS for adult ADHD has been deployed in a real clinical service and coupled to BPM, reporting qualitative gains in standardization and process transparency—yet without published quantitative diagnostic metrics (e.g., area under the curve (AUC), sensitivity, specificity) and without external validation [3]. In parallel, clinical ML studies show that abbreviated triage can retain performance close to longer protocols: in adolescents/adults with suspected ASD, models using a few items from the Autism Diagnostic Observation Schedule, Module 4 (ADOS-4) achieve AUC $\approx$ 0.82–0.87, near 11/31-item versions, highlighting nonverbal social cues (e.g., eye contact, directed facial expressions) and the quality of reciprocal social communication as discriminative axes [4, 2]. For DDx among ASD, early psychosis, and social anxiety, different algorithms—RF, Lasso/Elastic Net, and Bayesian Additive Regression Trees (BART)—that combine social cognition (e.g., Reading the Mind in the Eyes Test; RMET), executive function, attention/memory, and mood (e.g., Depression Anxiety Stress Scales; DASS-21) report AUC $\approx$ 0.72–0.92 depending on the task [1]. Scope reviews confirm the potential of artificial intelligence (AI) while highlighting persistent gaps: scarcity of public datasets (outside some imaging corpora), limited external validation, underused Explainable Artificial Intelligence (XAI), incipient interoperability (rare Electronic Health Record integration via HL7-FHIR), and limited impact evaluation (time to diagnosis, cost-effectiveness, clinician acceptance). Pragmatic trials (e.g., DECIDE-AI/CONSORT-AI), multimodality, and explainable, workflow-ready solutions are recommended [5].
This paper is organized into five main sections: introduction, theoretical framework, state of the art, presentation of the project Rule-Based Expert System for Adult NDD Differential Diagnosis, discussion, and conclusions.

2 State of the Art

There is increasing interest in integrating domain expertise and AI to enhance clinical decision-making for the differential diagnosis (DDx) of neurodevelopmental disorders (NDDs) in adults. This section summarizes the main approaches, reports representative evidence, and highlights open gaps supported by the recent literature.

2.1 State of the Art Methodology

To identify and analyze the main scientific evidence on the use of **Artificial Intelligence (AI)** applied to the **differential diagnosis of Neurodevelopmental Disorders (NDD)** in adults, a specific research protocol was developed, based on an **adapted PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses)**

model. This protocol guided all methodological stages, from the formulation of the research questions to the final screening and inclusion of studies, ensuring methodological rigor, transparency, and replicability.

Step 1 – Definition of Focus and Research Questions. The review was structured around four key dimensions:

- **Population:** adults ($\geq$ 19 years) with suspected or confirmed NDDs, including Autism Spectrum Disorder (ASD), Attention Deficit Hyperactivity Disorder (ADHD), Dyslexia, Developmental Coordination Disorder (DCD), and Tic/Tourette syndromes.
- **Intervention/Technology:** AI-based systems applied to differential diagnosis, such as machine learning (ML), deep learning (DL), expert systems, neural networks, natural language processing (NLP), neuro-symbolic models, Bayesian inference, decision trees, fuzzy logic, and clinical decision support (CDS) systems.
- **Context:** clinical, psychological, and neurological settings, as well as computational validation studies with clinical intent.
- **Outcomes:** diagnostic performance (sensitivity, specificity, AUC, F1-score), diagnostic time reduction, workflow efficiency, explainability (XAI), and clinical integration.

The research questions were structured as follows:

1. What is the current state of the art of AI in the differential diagnosis of NDDs in adults?
2. How have AI systems been designed and implemented for this purpose (workflows, data inputs, metrics, and clinical integration)?
3. Which paradigms, architectures, and business rules have been used (expert systems, Bayesian models, ML/DL, NLP, neuro-symbolic or hybrid models, ontologies, and coded screening guides)?
4. What gaps remain in this field (bias, generalization, interpretability, clinical validation, interoperability, and ethical–legal compliance)?

Rigorous inclusion and exclusion criteria were applied to systematically filter the literature and retain only the most relevant and methodologically robust studies. This process ensured that the selected references were high-quality, representative, and suitable to support the contributions of this work.

A total of **13 records** were identified: 9 from Web of Science, 2 from IEEE Xplore, and 2 from PubMed. The inclusion criteria were:

- Adult population ($\geq$ 19 years);
- Application of AI to differential or screening diagnosis of NDDs;
- Accuracy, validation, scoping/systematic reviews, or prototype studies with quantitative evaluation;
- Publication years between 2020 and 2025, in English, Portuguese, or Spanish.

The exclusion criteria were:

- Pediatric-only studies without adult sub-analysis;
- AI applied exclusively to treatment (non-diagnostic);
- Reports without quantitative evaluation;
- Case series with $<$ 10 participants;
- Opinion papers or studies without empirical data.

All screening steps were conducted independently by two reviewers to ensure reliability and consistency in study selection.

2.2 AI Support Families for Adult NDD Differential Diagnosis

Rule-Based CDSS Integrated with BPM. **Definition and process:** Knowledge-based CDSS encode clinical rules and orchestrate workflows (tasks, roles, deadlines), usually with *gates* that require completed steps before the consensus meeting and report issuance.
Representative example: an adult ADHD CDSS integrated with BPM (Serena Business Manager) that produces automatic summaries and coordinates a multidisciplinary team; qualitative gains in standardization/efficiency were reported, but no AUC/sensitivity/specificity have been published and no external validation was conducted [3].

ML Pipelines for Triage and DDx. **Definition and process:** pipelines comprising pre-processing (recoding, imputation, normalization), feature selection, training/validation (k-fold, hold-out), and testing, yielding classifiers that support DDx. **ASD in adolescents/adults:** radial Support Vector Machine (SVM) and Round Forrest (RF) with minimal ADOS-4 item subsets achieve AUC $\approx$ 0.82–0.87, close to longer algorithms; nonverbal social signals and reciprocity quality are central; external validation is lacking and circularity is possible (the reference diagnosis includes instruments also used as predictors) [4, 2]. **DDx ASD vs. early psychosis vs. social anxiety:** BART, RF, and Lasso/Elastic Net, combining social cognition, executive function, attention/memory, and mood, report AUC $\approx$ 0.72–0.92, without clinical deployment or external validation [1]. **ADHD synthesis:** a thematic review confirms the predominance of ML (SVM, RF) over deep learning (DL) in adults, reliance on private datasets, methodological heterogeneity, scarce XAI, and rare EHR/HL7-FHIR integration; it recommends pragmatic trials (DECIDE-AI/CONSORT-AI) and multimodality (including wearables) [5].

2.3 Research Questions and Synthesis of the Evidence

RQ1: What is the state of the art of AI for adult NDD differential diagnosis? Two axes prevail: rule-based CDSS already in operation (process standardization and transparency, but no published diagnostic metrics and no external validation) and ML pipelines with promising performance (for ASD, AUC $\approx$ 0.82–0.87 using few ADOS-4 items; for ASD–psychosis–anxiety DDx, AUC $\approx$ 0.72–0.92), yet mostly without clinical deployment and without external validation [3, 4, 2, 1]. Reviews emphasize potential and gaps (datasets, XAI, integration) [5].
RQ2: How have AI systems been designed and implemented for adult DDx (workflows, inputs, metrics, clinical integration)? CDSS+BPM: chained workflows (triage $\rightarrow$ assessments $\rightarrow$ consensus $\rightarrow$ report), business rules with gates, and automatic summaries; real service integration, but no published AUC/sensitivity/specificity and no external validation [3]. ML (research): rigorous preprocessing, feature selection (e.g., RFE, Boruta, Lasso), k-fold/holdout validation, and metrics such as AUC/accuracy/

sensitivity/specificity; inputs combine clinical instruments (e.g., ADOS-4, utism Diagnostic Interview-Revised (ADI-R), Adult ADHD Self-Report Scale (ASRS), Diagnostic Interview for ADHD in adults (DIVA-5)), neuropsychological tests, social cognition (e.g., RMET), and mood (DASS21); EHR/HL7 FHIR integration is rare [4, 2, 1, 5].

RQ3: Which paradigms, architectures, and business rules are used? Paradigms: rule-based (expert systems) in real settings; classical ML (SVM, RF, Lasso/Elastic Net, BART) dominating adult clinical studies; DL is a minority in this adult slice; Natural Language Processing (NLP), neurosymbolic, and ontology-based approaches are scarcely represented [3, 4, 2, 1, 5]. Business rules: explicit in CDSS (workflows, gates, mandatory scales) and implicit in ML pipelines via minimal item subsets and cutoffs (e.g., Youden's J), which can be encoded as clinical triage/referral rules [3, 2, 4].

RQ4: Which gaps remain (data, generalization, bias/fairness, interpretability, clinical validation, interoperability, ethics)? *Data and generalization:* predominantly single-site samples, high-functioning bias, lack of external validation, and risk of circularity (reference diagnosis includes instruments used as predictors) [4, 2]. *Bias and fairness:* subgroup analyses by sex, age, and comorbidities are rare; fairness is not systematically assessed [2, 4]. *Explainability (XAI):* limited use of SHapley Additive exPlanations (SHAP), Local Interpretable Model-agnostic Explanations (LIME), attention, or counterfactuals; variable-importance narratives predominate [1, 4, 2]. *Clinical validation and impact:* pragmatic trials (DECIDE-AI/CONSORT-AI), time to diagnosis, workload, and cost-effectiveness are seldom measured; the operational CDSS lacks published diagnostic metrics [3, 5]. *Interoperability and ethics:* EHR/HL7 FHIR integration is rare; privacy and consent are described at a high level, with little evidence of privacy-by-design architectures [3, 5].

3 Project: Rule-Based Expert System for Adult NDD Differential Diagnosis

3.1 Overview and Motivation

This project aims to build a *C*DSS for the DDx of NDDs in **adults**, with an initial focus on ASD and ADHD. The motivation is to standardize the diagnostic pathway, reduce uncertainty and assessment time, and provide traceable clinical evidence, without replacing professional judgment. The initial project presentation outlined core DDx challenges (overlapping symptoms, multidimensional assessment, limited consultation time) and the goal of *supporting clinical reasoning* with structured knowledge.

3.2 Scope and Design Decisions (Minimum Viable Product (MVP))

Following supervisory meetings, the team scoped the *MVP* to **adults** (≥ 19 years) with **two professionals** in the minimal viable workflow:

Neurologist and **Psychologist**. The objective is to structure an objective digital triage, standardized psychological testing, and a neurological report with organic exclusion, while preserving traceability and auditability of evidence.

3.3 Design Principles

- **Explainability and traceability:** clear rules, human-readable rationales attached to the report, and source logging (reports, scales, attachments).
- **Minimal viable flow:** Neurology $\leftrightarrow$ Psychology, with progressive inclusion of other specialties as clinically required.
- **Standardization:** Yes/No/Likert questions and recognized instruments to reduce ambiguity.
- **Complementarity:** the system supports—not replaces—professional judgment.

3.4 Inputs and Outputs (Concise Data Dictionary)

Priority inputs (Adults). Functional self-report (work/study/daily life); developmental history when remembered; symptoms across multiple contexts; comorbidities/medical conditions (sleep, anxiety, depression, vitamin deficits); evidence uploads (reports, exams). Items should be preferably objective (Yes/No/Likert).

System outputs. (i) **Triage sheet** with score and initial recommendations; (ii) **Structured psychology report**; (iii) **Neurology report** documenting organic exclusion and justifications; (iv) **Feedback plan** (referrals, accommodations, follow-up).

3.5 Logical Architecture and Knowledge Engineering Method

Knowledge acquisition via structured interviews focused on clinical heuristics; **modeling** as *IF* $\rightarrow$ *THEN* rules; **knowledge base** organized by entities (symptom, context, scale, criterion, evidence); **rule engine** (e.g., Drools/Prolog) for inference; **workflow orchestration** via BPM (e.g., BPM and Notation (BPMN)), with automatic document generation. This pipeline (interview $\rightarrow$ rules $\rightarrow$ prototype) follows the project presentation.

3.6 Assisted Workflow (High-Level BPMN)

1. **Structured digital triage** (Yes/No/Likert) + evidence upload $\rightarrow$ creation of the triage sheet and attachments.
2. **Psychology:** focused interview and *standardized tests* (executive functions, attention, social pragmatics) $\rightarrow$ **psychology report**.
3. **Neurology:** history, neurological exam, organic exclusion, integration of findings $\rightarrow$ **neurology report**.
4. **Feedback and plan** (referrals, accommodations, follow-up).

3.7 Business Rules (Decision Engine, MVP Version)

RB–1 Route to Neurology: Moderate/high screening signals + current functional complaint and *no recent medical assessment* ⇒ schedule Neurology; Psychology collects *baseline*.

RB–2 Prioritize Psychological Testing. High ASD/ADHD suspicion with occupational impact ⇒ standardized psychological battery + summary for Neurology.

RB–3 Pause and Reassess: Active comorbidities that mimic symptoms (severe anxiety, sleep disorders, vitamin deficits) ⇒ treat/stabilize before issuing the diagnostic report.

RB–4 Neurology Report. Convergence of triage + psychology report and *organic exclusion* ⇒ issue the neurology report; refer to Psychiatry when necessary.

3.8 Knowledge Representation and Application Approach

The modelling of expert knowledge in the system is structured through a rule-based representation approach, in which knowledge is encoded as a chain of questions organised in a decision-tree structure. To ensure clinical coherence and reflect how different specialties interpret clinical signs, a hybrid model was adopted in which the knowledge base is organised by specialty domains. These domains distinguish neurological and psychiatric axes and map functional areas such as:

- Cognitive and motor
- Sensory and emotional
- Behavioural and social

This organisation allows the questions to capture specific indicators that contribute to the probabilistic construction of neurodevelopmental disorders (NDDs). Tacit clinical knowledge, obtained through structured interviews, was formalised into explicit $IF \rightarrow THEN$ rules. The knowledge base is therefore organised into entities (such as symptom, context, scale, criterion, and evidence), preparing it to be processed by a rule engine (e.g., Drools/Prolog) that generates inferences and orchestrates the workflow via BPMN.

The inference of a possible diagnosis is obtained through a structured scoring system that quantifies the weight of each symptom within the differential diagnosis. This system is based on a questionnaire organised in a decision-tree structure.

To increase screening accuracy and reduce the likelihood of an incorrect diagnosis, the decision flow is initially segmented by a question that divides the system into two main sections:

- **Section 1:** Focused on Dyslexia and Dyspraxia, associated with motor and visual difficulties.
- **Section 2:** Focused on ADHD, Autism, Bipolar Disorder, Tourette Syndrome, and OCD, where emotional regulation and behavioural patterns are more evident.

Table 1. Variables and scoring definitions.

Variable	Definition / Calculation	Scale / Limits
Response scale	Specialist or patient indicates symptom frequency or severity.	0 = Never; 1 = Rarely; 2 = Sometimes; 3 = Often.
Scoring calculation	Each response (0–3) is multiplied by its question weight and summed to produce the condition score.	Score = response × weight.
Maximum total score	Maximum value per condition, used to compute the relative probability across diagnoses.	ADHD 10; Autism 10; Bipolar 10; Tourette 5.6; OCD 6.5; Dyslexia 5.5; Dyspraxia 5.5.

The inference process is straightforward: the patient's response is converted into a numerical value, which is then multiplied by the weight assigned to each question for a specific condition. The total sum of these weighted values for each condition allows the system to estimate the relative probability of each diagnosis.

Scoring and Decision Engine Calculation The inference calculation relies on a standardised response scale and predefined maximum scoring thresholds for each condition.

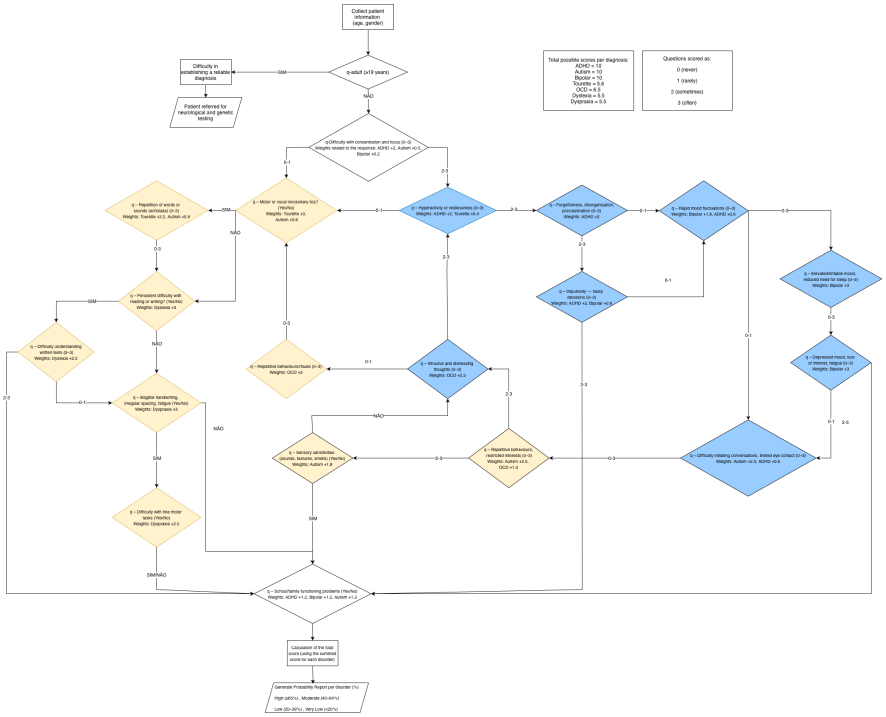


Fig. 1. Fluxograma do sistema

Example of scoring calculation Question: *"Difficulty concentrating and focusing"*
Expert's answer: 2
Application of weights:

$$ADHD: 2 \times 2 = 4, \quad Autism: 2 \times 0.3 = 0.6, \quad Bipolar: 2 \times 0.2 = 0.4.$$

For yes/no questions, the score for a given condition is added only if the answer is "yes". The system sums the values of all responses, resulting in a total score per diagnosis, which is then interpreted to estimate the relative probability of each condition.
The sequential and interconnected questions that compose the system guide healthcare professionals through a logical evaluation pathway, allowing the gradual identification of patterns associated with the different disorders analysed. In the system diagram (Fig.1), the questions highlighted in yellow represent items related to neurology—centred on brain, motor, and sensory functioning—while the questions in blue refer to the psychiatric domain, focusing on emotional, behavioural, and mood regulation aspects. This visual and logical structure ensures an integrated analysis, supporting clinical decision-making in a more consistent manner.

R1 (Age below 18): If the answer to this question is "yes", the system advises the clinician to discontinue the use of the tool and instead refer the patient for neurological and genetic evaluation.
R2 (Attention difficulties): When the response indicates a high frequency of attention-related difficulties, the weight attributed to ADHD is substantially increased.
R3 (Mood-related symptoms): If the patient reports marked or extreme fluctuations in mood, the weight associated with Bipolar Disorder is proportionally adjusted.
This system enables a preliminary inference that is quick and easy to apply, providing an initial diagnostic orientation.

The weights associated with each question were reviewed by several healthcare professionals who offered highly divergent opinions. Some argued that the system should not rely on weighted scoring across questions but were unable to propose an alternative approach; others stated that the weights were too inflated, while some suggested they were too low. A middle ground was sought to balance the contribution of each response to the different conditions.

We acknowledge that there are cases in which certain responses lead the system to assign diagnostic probabilities approaching 100%, which is unrealistic given the nature of this type of screening tool. Unfortunately, it was not possible to further adjust the weights or add additional questions to refine these scenarios due to time constraints and the unavailability of specialists from the relevant medical domains.

Example of System Output for a Use Case: In this use case, the patient answered *"No"* to the question *"Age between 0–18 years?"*.

This makes the patient eligible for diagnostic processing, as the system only evaluates individuals aged 19 or older.

The evaluation proceeds as follows:

1. **Difficulty concentrating and focusing (0–3)**: The patient answered **0**, which does not contribute points to ADHD, Autism, or Bipolar Disorder.
2. **Involuntary motor or vocal tics?** The patient answered **Yes**, adding:
   - Tourette: +3
   - Autism: +0.6
3. **Repetition of words or sounds (echolalia) (0–3)**: The patient answered **1**, adding:
   - Tourette: +2.2
   - Autism: +0.9
4. **Persistent difficulty in reading or writing (Yes/No)**: Answered **Yes**, activating the Dyslexia weight (+3) and triggering the follow-up question.
5. **Difficulty understanding texts (0–3)**: The patient answered **2**, resulting in:
   $$2 \times 2.5 = 5 points for Dyslexia,$$
   reaching the maximum score for this condition (5.5 points).
6. **School or family functioning problems (Yes/No)**: Answered **No**, so no additional points are added for ADHD, Bipolar Disorder, or Autism.

The system then sums all accumulated weights to generate the final diagnostic profile.

Condition	Accumulated Score
Tourette	5.2
Autism	1.5
Dyslexia	7.5

Condition	Maximum Score
Tourette	5.6
Autism	10
Dyslexia	5.5

Condition	Likelihood (%)
Tourette	92.8%
Autism	15%
Dyslexia	100%

Each group of questions was designed to capture specific indicators that, when analysed together, contribute to the probabilistic construction of neurodevelopmental disorders (NDD). This structure serves as the foundation for the weighting calculations and inference rules used in the expert system, enabling the estimation of relative probabilities across overlapping conditions (e.g., ADHD, ASD, dyslexia, dyspraxia, Tourette syndrome, bipolar disorder, or anxiety), with full clinical transparency and traceability. As shown in Fig. 2, we present the system interface with the final probability report, illustrating how the aggregated scores are translated into interpretable diagnostic likelihoods.

Fig. 2. System result screen

3.9 Roles and Responsibilities

Psychology: structured data collection, test administration, report with evidence and limitations. **Neurology:** history and examination, exclusion of organic causes, diagnostic decision and report. **Other specialties:** on demand, depending on case complexity.

3.10 Interoperability, Ethics, and Compliance

Interoperability: prepare data fields compatible with minimal HL7-FHIR profiles (Questionnaire, Observation, DiagnosticReport) to enable future EHR integration.
Ethics: obtain consent with a clear purpose statement (decision support, not replacement), enforce data minimization, and maintain versioning of rules with complete audit trails.
Governance: define policies for clinical attachments and role-based access control aligned with ethical use of knowledge gathered during interviews.

4 Discussion

4.1 Synthesis of Evidence

Across the reviewed literature, two complementary avenues emerge for adult NDD differential diagnosis (DDx): (i) rule-based clinical decision support systems (CDSS) embedded in business process management (BPM), which provide process standardization, explicit business rules, and auditable summaries; and (ii) machine learning (ML) pipelines that learn compact, discriminative sets of variables for triage and DDx [3, 1, 4, 2, 5]. The former demonstrates feasibility and acceptability in routine services, but lacks published diagnostic accuracy and external validation

[3]. The latter achieves promising AUCs (ASD vs. non-ASD with minimal ADOS-4 items $\approx$ 0.82–0.87; ASD vs. early psychosis vs. social anxiety $\approx$ 0.72–0.92), yet remains largely pre-deployment and single-site, with limited explainability and interoperability [1, 4, 2, 5].

4.2 Implications for Our Project (Adult, Neuro + Psychology, Rule-Based Core)

Guided by this evidence and our stakeholder meetings, we centered the MVP on adults and a two-professional workflow (Neurology + Psychology). We adopt a *rule-based core* (transparent business rules, gates, and auditable justifications) orchestrated by BPM for end-to-end traceability—aligned with strengths shown by real-world CDSS [3]. To capture the performance benefits reported by ML studies while preserving explainability, we incorporate *lightweight, optional* ML components for prioritization/triage using compact feature sets (e.g., minimal ADOS-4 items and brief neuropsychological/social cognition measures) [4, 2, 1]. Concretely, the project's business rules (RB–1..RB–4) encode safe prioritization (e.g., medical red flags $\Rightarrow$ Neurology first), standardized psychological testing when functional impact is high, and temporary deferral in the presence of destabilizing comorbidities—with all decisions logged and justified in the report. The reviewed literature highlights that the DDx of NDDs in adults is inherently challenging due to symptom overlap and the need for a multispecialty team. Historically, adult diagnosis requires the integration of neurological findings (organic exclusion) and psychiatric assessment (emotional and behavioural regulation). Our Minimum Viable Product (MVP) focuses on adults (19 years) and establishes a minimal workflow involving a Neurologist and a Psychologist, with the proposed system serving as an alternative to integrate these professionals. This project is currently in the development and testing phase, with clinical rules formalised into code (rule engine prototypes) and validated in collaboration with specialists in Neurology and Psychology, ensuring that the inferences follow recognised clinical criteria. The aim is to provide reliable and auditable decision support, helping to standardise the diagnostic pathway and reduce assessment time. Experts have considered the system to be a potential support tool for professionals working in the NDD field, particularly valuable for triage and for those on the frontline of diagnosis, given the challenge of meeting rising demand. The system aims to integrate the key specialties, organising the knowledge base into neurological and psychiatric axes, which facilitates navigation through abstract variations and the inherent complexity of DDx.

4.3 Methodological Considerations and Threats to Validity

The evidence base is affected by single-site datasets, potential circularity (ground truth includes instruments also used as predictors), and scarce external validation [4, 2]. Few works examine fairness (sex/age/comorbidity subgroups), and XAI remains underused beyond variable-importance

summaries [1, 2]. These issues motivate our evaluation plan to (i) predefine endpoints and analysis (including decision-curve analysis and calibration), (ii) report subgroup performance, and (iii) separate model development from reference adjudication to mitigate circularity. It is acknowledged that the evidence base for AI in DDx is affected by single-site samples and limited external validation. As the project is in a prototype stage with initial testing, it remains aware of these threats and is motivated to pursue an evaluation plan that includes separating model development from reference adjudication, thereby mitigating circularity. The central claim of the system's utility lies in its potential to reduce diagnostic time and streamline the process of filtering potential cases without loss of quality. The workflow assisted by business rules (RB–1..RB–4) encodes safe prioritisation, which translates into earlier treatment initiation and follow-up, ultimately improving patients' quality of life as early as possible.

4.4 Clinical Integration, Interoperability, and Ethics

Real-world impact requires integration with clinical workflows and records. Prior literature rarely reports HL7-FHIR/EHR integration [5], hence our design specifies a minimal interoperability profile (Questionnaire, Observation, DiagnosticReport) and audit trails for rule versions and evidence attachments. Privacy-by-design measures include purpose limitation (decision support, not automation), role-based access, and data minimization (objective Yes/No/Likert items when feasible). Each inference is accompanied by a human-readable rationale; ML-assisted recommendations expose feature attributions and uncertainty where applicable.

5 Conclusion

This work reviewed the state of the art in the use of artificial intelligence (AI) for the Differential Diagnosis (DDx) of Neurodevelopmental Disorders (NDDs) in adults. Two complementary paths were identified: (i) rule-based Clinical Decision Support Systems (CDSS) integrated into Business Process Management (BPM), which provide standardisation and auditable summaries; and (ii) Machine Learning (ML) pipelines that show promising performance for triage. Our project positions itself within this context, presenting an explainable and pragmatic model that uses a rule-based core orchestrated through BPM, complemented by lightweight and optional ML components for prioritisation. The project is currently under development and testing, built on expert knowledge to address the growing demand and the challenge of multidisciplinary assessment of NDDs in adults. The central contribution of this hybrid system, focused on integrating the Neurology and Psychology workflow, lies in its potential to reduce time and variability in the triage and case-filtering process. By ensuring traceability and explainability of decisions, the system has the capacity to accelerate the initiation of treatment and follow-up, contributing to improved quality of life for adults with NDDs.

References

1. Demetriou, E., Park, S., Ho, N., Pepper, K., Song, Y., Naismith, S., Thomas, E., Hickie, I., Guastella, A.: Machine learning for differential diagnosis between clinical conditions with social difficulty: autism spectrum disorder, early psychosis, and social anxiety disorder. Frontiers in Psychiatry **11**, 545 (2020). https://doi.org/10.3389/fpsyt.2020.00545
2. Kamp-Becker, I., Tauscher, J., Wolff, N., Küpper, C., Poustka, L., Röpke, S., Roessner, V., Heider, D., Stroth, S.: Is the combination of ADOS and ADI-R necessary to classify ASD? rethinking the "gold standard" in diagnosing ASD. Frontiers in Psychiatry **12**, 727308 (2021). https://doi.org/10.3389/fpsyt.2021.727308
3. Kemppinen, J., Salmisaari, T., Korpela, J., Polkko, J., Elfvengren, K., Tuominen, M.: A clinical decision support system for adult ADHD diagnostics process. In: Proceedings of the 46th Hawaii International Conference on System Sciences (HICSS). pp. 2614–2625. IEEE, Wailea, HI, USA (2013). https://doi.org/10.1109/HICSS.2013.30
4. Küpper, C., Stroth, S., Wolff, N., Hauck, F., Kliewer, N., Schad-Hansjosten, T., Kamp-Becker, I., Poustka, L., Roessner, V., Schultebraucks, K., Röpke, S.: Identifying predictive features of autism spectrum disorders in a clinical sample of adolescents and adults using machine learning. Scientific Reports **10**, 4805 (2020). https://doi.org/10.1038/s41598-020-61607-w
5. Loh, H., Ooi, C., Barua, P., Palmer, E., Molinari, F., Acharya, U.: Automated detection of ADHD: current trends and future perspective. Computers in Biology and Medicine **146**, 105525 (2022). https://doi.org/10.1016/j.compbiomed.2022.105525

NutriScan – Nutrition Analysis System

Hugo Ribeiro[1], Filipe Soares[2], Gonçalo Mendes[3], João Serra[4], Mariana Neves[5] and Tiago Gonçalves[6]

Instituto Politécnico do Porto, Porto, Portugal
[1]1141033@isep.ipp.pt, [2]1220823@isep.ipp.pt,
[3]1250421@isep.ipp.pt,[4]1201350@isep.ipp.pt,
[5]1211264@isep.ipp.pt,[6]1140331@isep.ipp.pt

Abstract. Growing public awareness of the connection between diet and health has increased the need for accessible and comprehensible nutritional information. To address this, we developed NutriScan, a web-based expert system designed to provide real-time nutritional analysis of food products. The system integrates barcode recognition with a knowledge base powered by Drools and Prolog inference engines, enabling intelligent reasoning over nutritional data. NutriScan offers detailed product evaluations and scoring. Through personalized user profiles, it identifies potential allergens, generates tailored alerts, and suggests healthier alternatives. The architecture combines both Java (Drools) and Prolog inference back-end components to explore the impact of two different technologies over AI techniques. The system demonstrates the integration of symbolic AI through Prolog-based logical inference and a structured knowledge database, showcasing how expert systems can deliver transparent, rule-driven nutritional analysis and decision support.
While both Drools and Prolog can be applied to rule-based reasoning, their underlying mechanisms differ substantially: Prolog employs backward chaining for logic-based inference, facilitating complex reasoning and knowledge representation, whereas Drools applies forward chaining to enable efficient, scalable rule evaluation with greater implementation clarity.
Overall, NutriScan leverages expert system principles and AI reasoning to support informed and health-conscious consumer decisions. The successful development and validation of NutriScan highlight the effectiveness of combining distinct inference paradigms to create intelligent, user-oriented decision-support tools.

Keywords: AI software, Expert systems, Knowledge Base, Drools, Prolog

1 Introduction

In recent years, growing public awareness of the relationship between diet and health has highlighted the importance of providing consumers with accessible and reliable nutritional information [19]. Poor nutrition and unhealthy diets are major contributors to rising global rates of obesity, diabetes, and heart disease, highlighting the need for better nutritional awareness [23] and decision-making tools. Despite the increasing

availability of food labeling systems, many individuals still struggle to interpret complex nutritional data and evaluate the health implications of their food choices [8]. This challenge has created a demand for intelligent, user-oriented tools capable of simplifying the process of understanding and comparing food products.

To address this need, NutriScan was developed – a web-based expert system that performs real-time nutritional analysis of food items. The application integrates barcode recognition with artificial intelligence reasoning mechanisms implemented using Drools and Prolog inference engines. By combining a structured knowledge base with symbolic AI techniques, NutriScan provides accurate and personalized evaluations, as well as alerts tailored to each user's dietary profile and restrictions.

This project aims, not only to create a functional application that supports consumers to have real-time information about the nutrition, but also to apply AI techniques over software engineering, to use different programming languages for implementing an AI solution and to measure the impact of different technologies.

2 State of the Art

Artificial Intelligence (AI) is a branch of computer science, that uses algorithms, data and hardware to simulate human intelligence, make decisions, discover patterns or perform some sort of action [10]. Currently, it is composed by the following AI techniques [15]: Machine Learning, Deep Learning [10], Natural Language Processing [18], Expert Systems [14] Computer Vision, Robotics [18], Reinforcement Learning [1].

Each subject focuses on a specific dimension of intelligence, such as perception in computer vision [18], reasoning in expert systems [14], or adaptation in reinforcement learning [1] and their integration within software applications allows for the development of highly autonomous and context-aware solutions.

2.1 Expert Systems

Expert systems are a branch of Artificial Intelligence, that can imitate and reproduce the flow of reasoning and decision making executed in human experts' brains to derive optimal solutions [14]. These systems, consists in a knowledge base, inference engine and user interface [2].

Knowledge bases are structured repositories designed to store, manage and retrieve information, enabling systems to solve complex problems through reasoning and inference [20]. Information is present in the knowledge base using facts and rules. **Facts** are represented by basic, unambiguous pieces of information considered true within a specific domain [9], such as "Window is open" or "Room is occupied". **Rules** are the are conditional statements that define relationships between facts and

guide the system's reasoning process [9]. As an example, "IF window is open and room is not occupied THEN close the window".

The **inference engine** applies logical reasoning to the knowledge base to derive conclusions, distinguishing expert systems from other computer systems [2]. The **explanation module** provides users with justifications or reasoning paths that led to a certain conclusion, increasing transparency and trust in the system's decisions [6]. The **user interface** allows interaction between the user and the system, enabling input of data and interpretation of outputs in an intuitive way [7].

By using the knowledge and validation of expert humans on a given subject, it becomes easier to achieve more accurate results and derive optimal solutions [14] within AI systems. It can be used in areas like medicine, business, computer science, law, defense, education, mathematics, engineering and more [3].

2.2 Programming Languages

AI techniques can be applied through various resources, especially programming languages. Languages such as Prolog, Java, and Python can be used to develop intelligent systems capable of reasoning, learning, and problem-solving [13].

Prolog is a declarative language, that allows facts and rules to be represented uniformly, making it useful for rule-based expert systems. Its goal-driven inference uses depth-first search with backtracking, and features like uncertainty handling or explanations can be added by extending predicates. Expert systems can be fully implemented in Prolog without a host language, as demonstrated by both small and large-scale applications [14].

However, Prolog has limitations: its rigid control strategies can make some inferences inefficient, and procedural constructs, like "assert" or "cut", can reduce program clarity, by making it less declarative. Knowledge base structuring is limited, and mixing domain and control knowledge can reduce modularity, complicating system extensions such as explanation mechanisms [14].

Java is an object-oriented programming language, that is widely used for building applications. Its fusion with artificial intelligence offers a powerful synergy, enabling developers to create intelligent systems and applications with efficiency, robustness and scalability [21].

Java provides many AI libraries and tools, offering many solutions for programmers to implement AI algorithms and techniques [21]. Among other tools, there is available **Drools**, a Business Rules Management System (BRMS) and a rules engine [5]. It is a rule-based approach that allows to implement an Expert System, with a knowledge base holding the rules; internal types, and processes, representing the system's knowledge; the working memory stores the facts; and these facts against the rules that infers the facts [21].

By the other hand, Java shows to have runtime overhead and performance concerns, and it tends to have more complexity and verbosity when comparing to other languages, like Python [13].

2.3 Nutrition Apps

The advances in information technology have made nutrition applications more accessible and widely used. Artificial intelligence enhances these apps by enabling features like automated food recognition and personalized dietary recommendations, making diet monitoring more convenient and tailored to individual needs [22]. This personalization aims to optimize nutrition plans, potentially improving health outcomes for users [22].

Nutrition applications are described to be a positive asset for consumers, having the potential to allow them to have more informed decisions regarding their food intake [12] and support disease treatment and control, like diabetes [16]. Some applications were confirmed to be a major improvement in monitoring diet quality and weight control and, preventing the development of diet associated chronic diseases [12].

Nevertheless, there are still several challenges that must be addressed, such as the lack of standardization of food databases and the difficulty in accurately recognizing foods and measuring nutrient content [16].

3 Project

To address the need of consumers for nutritional advising, it was created NutriScan, a web application, that provides an intelligent system to allow consumers to evaluate in real-time food products and its nutritional value, having into account their profile and food restrictions.

NutriScan offers a **smart scanning** supported by mobile devices, with automatic recognition of barcodes, integration with standard product databases and offline processing of previously searched products. With **consumer profiles**, the application considers the restriction management, of allergies, intolerances, and dietary preferences; provides critical alerts when detecting hazardous ingredients; holds contextual analysis. Its **analysis system**, provides a nutritional analysis, taking sugars, salt, fats, additives and nutritional value in consideration. It identifies food properties like vegan, organic, and gluten-free products. It used visual indicators, as traffic light color system and pictograms for quick user interpretation and alarm.

To sustain the previous functional requirements, it was defined the following technical goals for this project:

- Implement a robust barcode scanning system.

- Develop a Drools/Prolog rules engine for intelligent product classification.
- Implement a personalized profile system with dietary restriction management.

3.1 Application Architecture

The architecture of NutriScan project is composed by the following components:

- **Java backend server** – is a Java 1.8 API server application that handles requests from UI, requests to Prolog and Drools and database connections.
- **Prolog backend server** – is a Prolog API server application that handles Prolog logical inference system.
- **Drools backend server** – is a Java application that uses Drools to handle rules engine.
- **UI application** – is the user interface from the application, written in React, with the main feature allowing end users to interact with out internal back-end APIs.

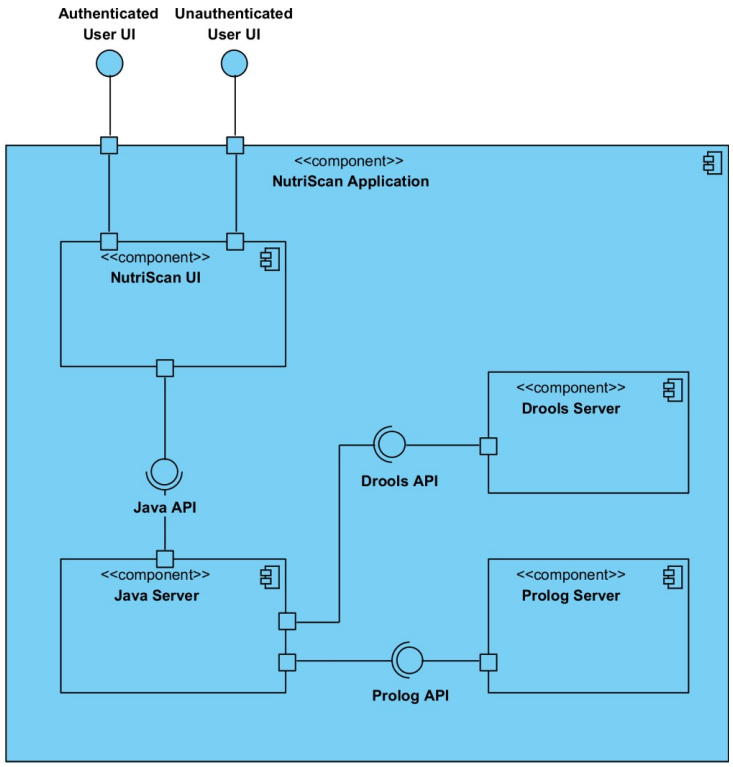


Fig. 2. Level 2 Logical View Component Diagram.

Using this architecture, we can divide the application into different components according to their technological requirements. The front-end is naturally separated due to its UI nature and React stack. A main Java server responsible for managing

communication between the UI and the various back-end components. With this AI-oriented approach, we are able to maintain two distinct models, which is an essential aspect of this project's goal, to study and compare the results of Java and Prolog implementations.

By using C4 with 4+1 model, NutriScan application architecture could be represented. C4 model, divides software architecture into different levels: context (level 1); containers (level 2); components (level 3); and code (level 4) [4]. 4+1 model divides software in logical view, process view, development view, physical view and uses cases [11].

3.2 Expert System

For building NutriScan it was used an expert system, to ensure that the knowledge base reflects accurate, context-specific expertise, allowing the system to simulate human-like judgment more effectively.

Interacting with experts is essential for a rule engine, as they provide the base knowledge that shapes its facts, rules, and inference. Interviews were run with three experts from different backgrounds (marketing and nutrition).

Beatriz Almeida and Flávio Gart, are two mentors with experience as directors in marketing agencies, support the NutriScan project by validating the application features, advising about project's potential for successful market adoption and user engagement, validating the application features and business decisions.

Daniela de Sousa, a nutritionist specializing in Clinical and Sports Nutrition, provides essential nutritional expertise to NutriScan. She helps shape the app's health recommendations and dietary analysis features, by providing the nutrition information and the rules to infer with the existing knowledge base.

3.3 Rule Implementation

After having the meetings and discussions with the experts, it was defined two major groups of rules to implement in Drools and Prolog, as it follows:

Core Rules: General Nutritional Assessment. These rules evaluate products based solely on their nutritional composition and ingredient profile, independently of the user's health data. They identify general positive and negative nutritional attributes, such as: High or low content of salt, sugar, fat (total/saturated), and energy density; Positive nutritional features such as low fat, low sugar, high fiber, or protein source. Each rule corresponds to a clear nutritional threshold derived from expert guidelines.

Personalized Alerts: User-Specific Warnings. These rules integrate user health profiles (e.g., hypertension, diabetes, pregnancy, childhood) to generate personalized or critical alerts. Examples include: Alerts for high salt, sugar, or fat intake in users

with related conditions; Critical warnings for allergens such as gluten; Specific advisories for children (e.g., artificial colorants, added sugars), pregnant women (artificial sweeteners), and sensitive groups (polyols or low energy foods).

4 Results and Discussion

This section presents the results obtained from the development and evaluation of the NutriScan application.

4.1 Application

The built application offers a clean and modern view for users, with the proposed features implemented. It allows users to create a profile with detailed information about their food restrictions, properly scanning a product barcode, select different analysis tools and give user food suggestions and alerts.

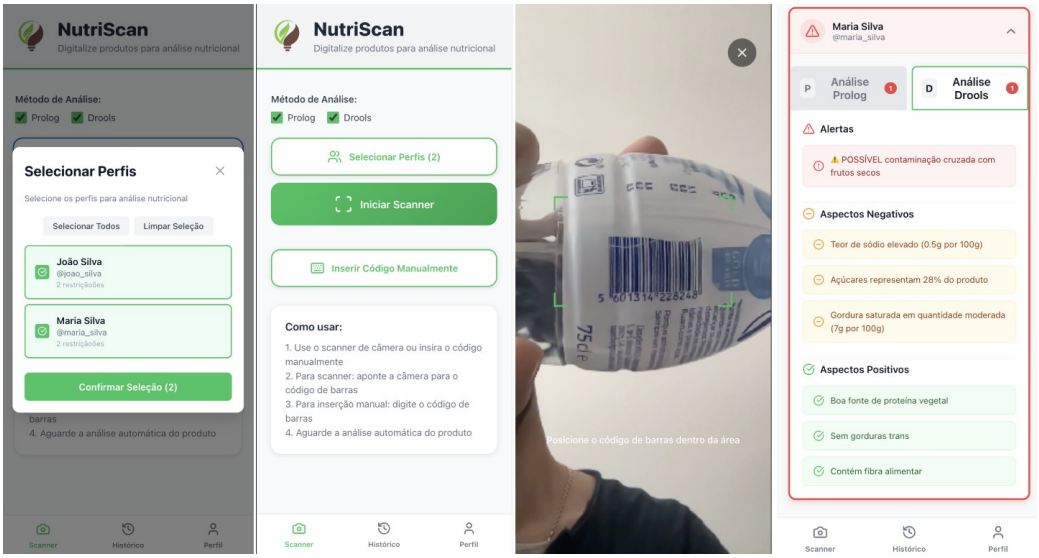


Fig. 3. NutriScan user interface and its features. From left to right: profile selection; analysis and scanning menu; barcode scanning view; analysis results view.

Login. Is the homepage of the NutriScan that allows users to authenticate on the website. Only visible for non authenticated users.

Account Creation. Allows non registered users to create an account. Only visible for non authenticated users.

Create and Edit Profile. This view allows users to create and edit profiles, to fill their name, gender, birth date, if it is pregnant or breastfeeding. It is in this section that users insert information about is allergies, diseases like hypertension, diabetes or celiac disease.

Profile Management. Allows users to manage multiple profiles from the same account, with options to list, create, see details, remove and update profile options.

Analysis Selection. This view holds the feature of selecting analysis methods (Prolog and/or Drools), selecting profiles for analysis, a button for scanning using the camera or the option to insert manually the bar-codes.

Barcode scanning. It shows the camera view, in order to start finding and scanning a code, from a product packaging.

Product analysis result. This view shows details about the scanned product (if it is found) and displays its description, nutritional value and the profile analysis for each one of the profiles. There is a tab separating Prolog and Drools, and inside of each one, it shows the alerts for the found food ingredient restrictions and allergies, positive and negative aspects according to nutritional value.

Historic. Allows to consult previous analysis done by the users.

4.2 Test Case

To validate the application, a complete acceptance test was conducted, covering all features from user registration to the history of analysis results, following a *happy path* scenario.

1. A user account was registered by entering an email address and password, after which authentication was successfully completed.

2. Within the NutriScan application, a consumer profile was defined representing a three-month-old female with barley allergy.

3. The analysis parameters were then configured, by selecting the previously created profile, both the Drools and Prolog inference models, and a product was scanned using its barcode.

4. The scanned product corresponded to a chocolate cereal, and the system displayed its nutritional information per 100g: 385 kcal, 24.8g of sugar, 0.2g of salt, and 4.6g of fat.

5. The analysis produced the following results in both inference models:

    - **Alerts:** not recommended for children under 12 months due to added sugars; contains barley, which is contraindicated when allergic to this ingredient.

    - **Negative aspects:** high sugar content.

- **Positive aspects:** high fiber content; low salt; low fat.

6. Finally, it was verified that this analysis was correctly stored in the user's history list.

These results confirm that both inference engines, Drools and Prolog, generated consistent outputs, and that the system's core functionalities (registration, analysis, and data persistence) operated as expected.

5 Conclusion

This study presented the development of a nutrition-focused application that integrates an expert system using Prolog for logical inference and Drools for rule-based decision management.

The application was successfully developed to provide personalized nutritional guidance based on individual user profiles. The results demonstrate the potential of expert systems, to enhance decision support and validate specialist knowledge in nutrition-related applications.

Regarding Drools and Prolog implementations, they can be used for similar goals (rule-based reasoning or expert systems), but their underlying nature and reasoning processes are fundamentally different. Prolog is a logic programming language that uses backward chaining to infer conclusions from a set of facts and logical rules, making it well suited for complex reasoning and knowledge representation. In contrast, Drools is a forward-chaining rule engine designed for efficient rule evaluation. While Prolog emphasizes declarative logic and inference, Drools focuses on scalable rule execution and its implementation seems to be have more readability.

5.1 Limitations and Future Work

The current knowledge base is constrained by the existing scope, which may not cover all nutritional scenarios or individual variations. Additionally, the system's recommendations depend on the quality of user-provided data and have not yet been clinically validated. Future work should focus on expanding the rule set and conducting broader evaluations with expert and user feedback.

References

1. Aggarwal, C., Reinforcement learning. In: *Artificial Intelligence*, pp. 343–384. Springer, Cham (2021).
2. Asemi, A.: Intelligent libraries: a review on expert systems, artificial intelligence, and robot. Library Hi Tech 39(2), 412–434 (2021).
3. Bavakutty, M., Salih, M.: Research on Library Computerisation. 1st edn. Ess Ess Publications (2006)
4. C4 Model, https://c4model.com, last accessed 2025/10/08

5. Drools Expert User Guide, Chapter 1: The Rule Engine, https://docs.drools.org/5.2.0.M2/drools-expert-docs/html/ch01.html, last accessed: 2025/10/25.
6. Expert Systems: Definition, Components, and Applications, Studylib. https://studylib.net/doc/25943845/ai-unit-5, last accessed 2025/10/28.
7. Füßl, A. and Nissen, V. "Design of explanation user interfaces for interactive machine learning using the example of a knowledge graph-based approach to explainable process analysis," Requirements Engineering, vol. 30, no. 1, pp. 81–108 (2025).
8. Grunert, K., Wills, J.: *A Review of European Research on Consumer Response to Nutrition Information on Food Labels*. Journal of Public Health, 15(5), 385–399 (2007).
9. Gupta, I., Nagpal, G.: Chapter 5: Expert Systems. In: *Artificial Intelligence and Expert Systems*, pp. 71–94. Mercury Learning and Information, Berlin/Boston (2020).
10. Janiesch, C., Zschech, P., Heinrich, K.: Machine learning and deep learning. *Electronic Markets* 31, 685–695 (2021).
11. Kruchten, P.: Architectural Blueprints—The "4+1" View Model of Software Architecture. *IEEE Software* 12(6), 42–50 (1995).
12. Kong, N., Moy, F., Ong, S., Tahir, G., Loo, C.: MyDietCam: Development and usability study of a food recognition integrated dietary monitoring smartphone application. Digit. Health 9, 1-18 (2023).
13. Marini, D., Khamlichi, A., El Alaoui, I.: Comparative Analysis of Four Programming Languages for Machine Learning. Int. J. Information and Systems Sciences 30(6), 1069–1077 (2023).
14. Matsuzaka, Y., Yashiro, R.: AI-Based Computer Vision Techniques and Expert Systems. AI 4, 289–302 (2023).
15. Mukhamediev, R., Popova, Y., Kuchin, Y., Zaitseva, E., Kalimoldayev, A., Symagulov, A., Levashenko, V., Abdoldina, F., Gopejenko, V., Yakunin, K., Muhamedijeva, E., Yelis, M.: *Review of Artificial Intelligence and Machine Learning Technologies: Classification, Restrictions, Opportunities and Challenges.* Mathematics, 10(15), 2552 (2022).
16. Rahman, L., Papathanail, I., Brigato, L., Spanakis, E., Mougiakakou, S.: *Food Recognition and Nutritional Apps*. (2023).
17. Rossi, G.: *Uses of Prolog in Implementation of Expert Systems*. *New Generation Computing*, 4(3), 321–329 (1986).
18. Ruiz, P., Fusco, J., Glossary of Artificial Intelligence Terms for Educators. Digital Promise (2024). https://digitalpromise.org/wp-content/uploads/2024/12/CIRCLS-Glossary-of-Artificial-Intelligence-Terms-for-Educators.pdf, last accessed 2025/10/28.
19. Silva, P., Araújo, R., Lopes, F., and Ray, S., Nutrition and Food Literacy: Framing the Challenges to Health Communication. Nutrients 15(22), 4708 (2023).
20. Sriram, R., Knowledge-Based Expert Systems: An Overview. In: *Intelligent Systems for Engineering*, pp. 1–22. Springer, London (1997).
21. Vyas, B.: Java-Powered AI: Implementing Intelligent Systems with Code. Journal of Science & Technology 4(6), 1–12 (2023).
22. Wang, X., Sun, Z., Xue, H., An, R.: Artificial Intelligence Applications to Personalized Dietary Recommendations: A Systematic Review. *Healthcare (Basel)* 13(12), 1417 (2025).
23. World Cancer Research Fund International and NCD Alliance: *The Link Between Food, Nutrition, Diet and Non-Communicable Diseases*. World Cancer Research Fund International, London (2014). Available at: https://www.wcrf.org/research-policy/library/link-between-food-nutrition-diet-ncds/

Session 6A

"Soluções Avançadas para Monitorização, Controlo e Análise em Azure"

Chair: Alexandre Gouveia **Room:** H202 **Time:** 16:45 - 17:30

Paper	Presenter
Governance Azure: Solução Integrada de Monitorização e Controlo de Custos	David Mendonça
Sistema de Manutenção Proativa e Resposta a Incidentes de Segurança	Diogo Nunes
Power BI Conversacional com RAG	João Fernandes

Governance Azure: Solução Integrada de Monitorização e Controlo de Custos

David Tavares Mendonça[1] and F. Jorge Duarte[1]

ISEP – IPP, Porto
1211572@isep.ipp.pt, fjd@isep.ipp.pt

Resumo A crescente complexidade dos ambientes cloud impõe desafios significativos à governança, resultando frequentemente numa fragmentação entre a monitorização operacional e o controlo financeiro. Embora as ferramentas nativas sejam robustas, a dispersão da informação obriga a uma alternância ineficiente entre interfaces, dificultando a correlação entre métricas de desempenho e o seu impacto orçamental. Este artigo propõe uma solução arquitetural integrada para ambientes Microsoft Azure, desenhada para centralizar a observabilidade de recursos e a gestão de custos. Através de uma implementação de referência fundamentada em *Clean Architecture*, demonstra-se como a centralização numa interface de APIs heterogéneas e a abstração de linguagens de consulta complexas (KQL) permitem otimizar a gestão de custos e recursos no Azure. Os resultados mostram redução do esforço na consulta técnica e financeira, através de visualizações interativas e filtros permitindo a utilizadores com diferentes níveis técnicos obterem *insights* acionáveis.

Keywords: Governança · Gestão de custos · Monitorização de métricas · Microsoft Azure · ASP.NET Core · React · KQL · Azure AD · Clean Architecture · Web API

1 Introdução

A adoção generalizada da computação em nuvem transformou radicalmente o paradigma de gestão de infraestruturas de TI (Tecnologias da Informação). Contudo, esta transição trouxe consigo desafios significativos de governança e controlo orçamental. Estudos recentes indicam que a gestão ineficiente de recursos *cloud* é uma das principais causas de desperdício financeiro em grandes organizações, impulsionando a emergência de práticas de *FinOps* para alinhar operações tecnológicas e responsabilidade financeira [1]. A complexidade inerente aos modelos de faturação dinâmica e à elasticidade dos recursos exige, portanto, estratégias robustas de *Cloud Cost Management and Optimization* (CCMO) que transcendam a simples monitorização técnica [2].

Apesar da disponibilidade de ferramentas nativas no ecossistema **Microsoft Azure**, a informação crítica encontra-se frequentemente fragmentada. Esta dispersão obriga os administradores a uma alternância ineficiente entre interfaces e exige um domínio técnico avançado de linguagens de consulta, como **KQL**, que

é a linguagem de consultas predominante em logs, métricas e telemetria, para correlacionar métricas operacionais com o seu impacto financeiro. Enquanto a literatura académica tem feito avanços nos estudos de *Machine Learning* para a deteção automática de anomalias de custos [3] e controlo baseado em políticas [4], persiste uma lacuna na disponibilização de plataformas unificadas que reduzam a carga cognitiva dos operadores humanos na gestão diária.

Para utilizadores não técnicos, a barreira de entrada para compreender a relação entre o desempenho de um recurso e o seu custo permanece elevada. O presente artigo propõe uma solução arquitetural integrada para ambientes **Azure**, desenhada para centralizar a observabilidade e a gestão de custos. O objetivo é demonstrar a relevância da centralização de métricas e custos numa única interface que oferece visualizações interativas e intuitivas, bem como consultas KQL pré definidas e personalizáveis de forma a acelerar o processo de tomada de decisões operacionais e financeiras e acabar com a dificuldade em gerir depósitos de informação em serviços dispersos.

2 Estado da Arte

O presente capítulo enquadra a solução proposta no panorama atual da *governance* em ambientes *cloud*.

2.1 Estudos académicos

A adoção generalizada de serviços *cloud* impulsionou o desenvolvimento de estratégias de *governance* mais sofisticadas [2].

Neste contexto, o conceito de *CloudOps* evoluiu para incorporar práticas de responsabilidade financeira, culminando na metodologia *FinOps*. Esta abordagem cultural e operacional visa alinhar as equipas de engenharia, finanças e gestão em torno da otimização de custos em larga escala [1]. Modelos teóricos como o CARES (*Cost, Automation, Reliability, Elasticity, Security*) sistematizam estes princípios, promovendo uma gestão equilibrada onde o controlo financeiro não compromete a fiabilidade ou a elasticidade dos serviços [1].

Paralelamente à evolução dos modelos de gestão, a investigação tem explorado a aplicação de técnicas de inteligência artificial como catalisadores na monitorização e otimização financeira [3]. Estudos sobre alocação de recursos evidenciam que a distribuição eficiente de recursos deve considerar simultaneamente restrições de custo e eficiência energética, adotando algoritmos de otimização para ajustar dinamicamente a infraestrutura [5].

Especificamente na deteção de padrões de consumo, os estudos académicos destacam o uso de *Machine Learning* para identificar anomalias de custos e prever variações orçamentais. Abordagens como o *AWS PredSpot* demonstram a eficácia de modelos preditivos na antecipação de flutuações de preços em instâncias *spot* [6]. Contudo, embora estes modelos matemáticos sejam robustos na deteção, persiste a necessidade de interfaces de *governance* que traduzam estas previsões em políticas de controlo acionáveis por operadores humanos [4].

Este desfasamento entre a sofisticação algorítmica e a usabilidade operacional motiva a arquitetura centralizada proposta neste artigo.

2.2 Soluções existentes

A gestão eficaz de ambientes *cloud* exige a correlação contínua entre métricas operacionais e o seu impacto financeiro. No ecossistema Microsoft Azure, esta observabilidade é assegurada por serviços nativos robustos, nomeadamente o **Azure Monitor** para telemetria [7], o **Log Analytics** para consulta de registos [8] e o **Cost Management** para controlo orçamental [9]. Embora estas ferramentas ofereçam granularidade técnica elevada, operam frequentemente em depósitos de informação. A literatura identifica esta fragmentação como um obstáculo crítico, obrigando os gestores a alternar entre múltiplos portais e a dominar linguagens de consulta complexas (como KQL) para obter uma visão holística do estado do sistema [2].

Para mitigar esta dispersão, o mercado evoluiu para a oferta de *Cloud Management Platforms* (CMPs) comerciais, tais como o Turbo360 ou o CloudHealth [10] [11]. Estas soluções agregam dados e centralizam a governança, validando a necessidade de interfaces unificadas [1]. Contudo, a adoção destas ferramentas introduz barreiras significativas: além dos custos de licenciamento, apresentam frequentemente curvas de aprendizagem acentuadas e interfaces densas.

Consequentemente, identifica-se a necessidade de soluções que não só integrem dados, mas que também reduzam a dificuldade da sua análise. A solução *Governance Azure* proposta distingue-se precisamente por priorizar a experiência do utilizador aliada com várias formas de visualizar os dados, com diferentes granulariadades facilitando a extração de *insights* relevantes para a gestão.

2.3 Tecnologias utilizadas

A implementação da solução baseia-se em tecnologias que equilibram robustez, escalabilidade e integração com o ecossistema Azure. O *backend* foi desenvolvido em **ASP.NET Core (.NET 9)**, framework *open-source*, que oferece suporte nativo a injeção de dependências e autenticação com **Azure AD** [12]. O sistema de autenticação utiliza **Azure Active Directory (Entra ID)** para garantir controlo de acessos baseado em papéis (RBAC) e segurança na comunicação entre camadas, sendo preferido neste contexto face a alternativas *cloud*. Foram consideradas alternativas a estas tecnologias, mas não foram adotadas por razões de adequação ao contexto.

Para o *backend* foi avaliado **Spring Boot** (**Java**); contudo, **.NET** oferece integração nativa com **Azure** e com as respetivas *APIs*, o que justifica a sua escolha.

Para a camada de apresentação, adotou-se o modelo de *Single Page Application* (SPA). Embora **Angular** e outras alternativas tenham sido consideradas, **React** revelou-se o mais interessante por ser um *framework* revelou-se mais interessante devido à vasta gama de bibliotecas de visualização de dados disponível [13].

Do ponto de vista de identidade, soluções de *IAM* de outros *cloud providers* (como **AWS** ou **Google Cloud**) foram descartadas por quebrarem a coerência tecnológica e por introduzirem fricção na integração com os serviços **Azure**. Assim, a combinação **ASP.NET Core**, **React** e **Azure AD** alinha-se com as necessidades funcionais e não funcionais do projeto.

3 Análise e Desenho da Solução

3.1 Requisitos funcionais e não funcionais

O processo de análise conduziu à definição de requisitos que orientaram o design funcional e técnico do sistema.

Os principais requisitos **funcionais** identificados foram:

- **RF01:** Permitir a autenticação de um utilizador pertencente ao *tenant* da organização na aplicação através do **Azure AD**.
- **RF02:** Permitir a listagem e pesquisa de recursos **Azure** associados à subscrição do utilizador, de forma a gerir os recursos presentes.
- **RF03:** Permitir a consulta de métricas de desempenho de um recurso específico, para acompanhar o seu estado e atividade.
- **RF04:** Permitir a execução de consultas *KQL* (pré-definidas ou personalizadas) sobre *logs* para analisar métricas operacionais, podendo ajustar a *query* às necessidades.
- **RF05:** Permitir a consulta de custos da subscrição com filtros e agrupamentos dinâmicos, facilitando a compreensão do consumo e das tendências financeiras.

Complementarmente, foram definidos requisitos **não funcionais** que garantem a qualidade da solução e simplicidade de utilização:

- **Usabilidade** foco na simplicidade, clareza e fluidez de utilização, sem dependência de conhecimentos técnicos, design visual moderno e coerente.
- **Fiabilidade**- a aplicação deverá gerir de forma controlada, os erros resultantes da falta de dados.
- **Desempenho** utilização de paginação de resultados, de forma a carregar um número limitado de recursos por página para não comprometer a *performance*.
- **Suportabilidade** documentação robusta que visa facilitar a manutenção e evolução do projeto.

3.2 Arquitetura da Solução

A arquitetura da solução foi desenhada sob os princípios da **Clean Architecture**. Esta arquitetura visa assegurar baixo acoplamento, alta coesão e testabilidade. O *Backend* implementa esta arquitetura distribuída por quatro camadas lógicas fundamentais:

1. Camada de apresentação: Contém os *controllers* e a exposição dos *endpoints* REST, servindo como ponto de entrada para o *Frontend*;
2. Camada de aplicação: Responsável pela orquestração de casos de uso e transformação de modelos de dados através de **DTOs**, dissociando as entidades de domínio dos contratos externos;
3. Camada de domínio: O centro da arquitetura, onde residem as regras de negócio e as interfaces de serviço, isoladas de detalhes de implementação externa;
4. Camada de infraestrutura: Implementa as interfaces definidas no *Core*, concretizando as integrações com os serviços Azure.

Foram aplicados princípios **SOLID** e padrões **GRASP** para reforçar a clareza estrutural e a independência entre componentes. Na Fig. 1 é apresentado o diagrama da *vista lógica de nível 2*, que ilustra a arquitetura da solução e a forma como os principais componentes interagem entre si.

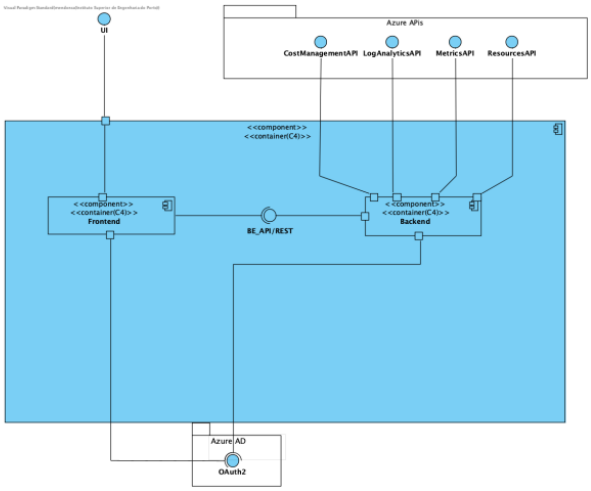


Fig. 1. Diagrama de componentes C4 Vista lógica Nível 2 (UML)

As principais interações dos componentes representados no diagrama acima são: O Frontend (React SPA) autentica-se no Azure AD (OAuth2/MSAL) e, de posse do *token*, invoca a API REST do Backend (ASP.NET Core). O Backend aplica regras de negócio e orquestra chamadas às Azure APIs para ter gestão de recursos e custos, métricas e análise de logs. Serviços transversais (autenticação, *logging*, DI e DTOs) suportam ambos os componentes, garantindo observabilidade e integração consistente.

4 Implementação da Solução

4.1 Segurança: Autenticação e Autorização

A identidade é garantidada através de *OAuth 2.0* com **Azure AD**. No *frontend*, **MSAL.js** obtém e renova *Bearer tokens*, consumido pela WebAPI para invocar as APIs da Azure. No *backend*, o acesso é concedido por políticas baseadas em *roles* do Azure, refletindo diretamente no que o utilizador pode consultar e visualizar.

4.2 Backend e Integrações Azure

A interação com o ecossistema Azure é centralizada na camada de infraestrutura, onde serviços dedicados encapsulam a complexidade da comunicação HTTP e a gestão de autenticação. A arquitetura assegura a cobertura das funcionalidades forneceidas pelas APIs do Azure escolhidas.

A Azure Resource Manager API expõe e gere os recursos; a Azure Monitor API inspeciona-os e produz métricas que suportam alertas operacionais; e a Log Analytics API consulta *workspaces* para obter entradas de log, também ligadas aos mesmos recursos. Em paralelo, a Cost Management API explora custos e origina *cost entries* associados a subscrições, grupos ou etiquetas, permitindo análises por centro de custo.

A Fig. 2 ilustra a implementação deste serviço.

```
    public async Task<string> GetAvailableMetricsAsync(string resourceId)
    {
        var token = await
_authService.GetAccessTokenAsync("https://management.azure.com/.default");
        _httpClient.DefaultRequestHeaders.Authorization = new
AuthenticationHeaderValue("Bearer", token);

        var url =
$"https://management.azure.com{resourceId}/providers/microsoft.insights/metricDefinitions?
api-version=2018-01-01";

        var response = await _httpClient.GetAsync(url);
        if (!response.IsSuccessStatusCode)
        {
            var error = await response.Content.ReadAsStringAsync();
            throw new Exception($"Erro ao obter métricas disponíveis:
{response.StatusCode} - {error}");
        }

        return await response.Content.ReadAsStringAsync();
    }
```

Fig. 2. Invocação da API do Azure Monitor.

4.3 Frontend e Visualização

O **Frontend** em **React** organiza a navegação por domínios funcionais (recursos, métricas, custos e *logs*) com acesso controlado por rotas protegidas. As interfaces foram desenhadas para facilitar a exploração rápida de dados através de tabelas com pesquisa e gráficos interativos.

A análise financeira é suportada pelo painel de **Sumário de Custos**, onde o utilizador pode aplicar filtros por intervalo temporal, granularidade e agrupamento. Os resultados são renderizados visualmente, permitindo comparar períodos ou observar a distribuição de gastos por categoria através de gráficos circulares, conforme exemplificado na Fig. 3.

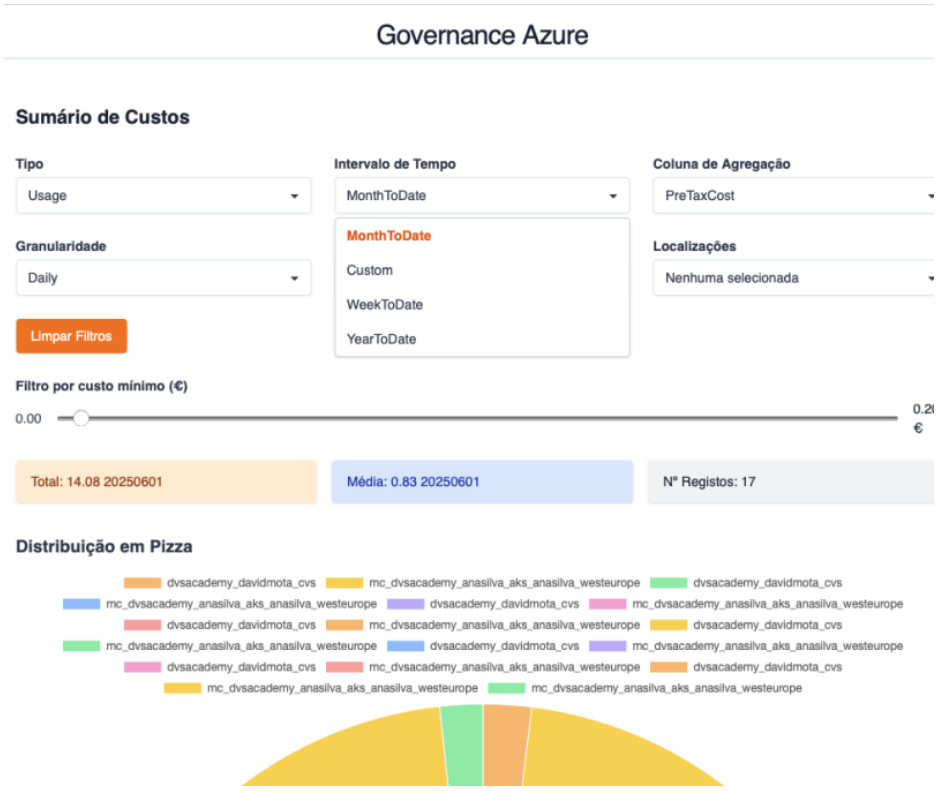


Fig. 3. Visualização interativa da distribuição de custos por recurso.

Para a monitorização operacional, a interface de **Inspeção de Métricas** permite selecionar o recurso, a métrica e a função de agregação desejada. O sistema projeta um gráfico de linhas para análise de tendências, complementado por *KPIs* automáticos (média, máximo e mínimo) e funcionalidade de exportação.

Por fim, a complexidade da consulta de registos é gerida num editor dedicado (Fig. 4), que permite a execução de *queries* KQL e a personalização da consulta pré definida para ajuste às necessidades do utilizador.

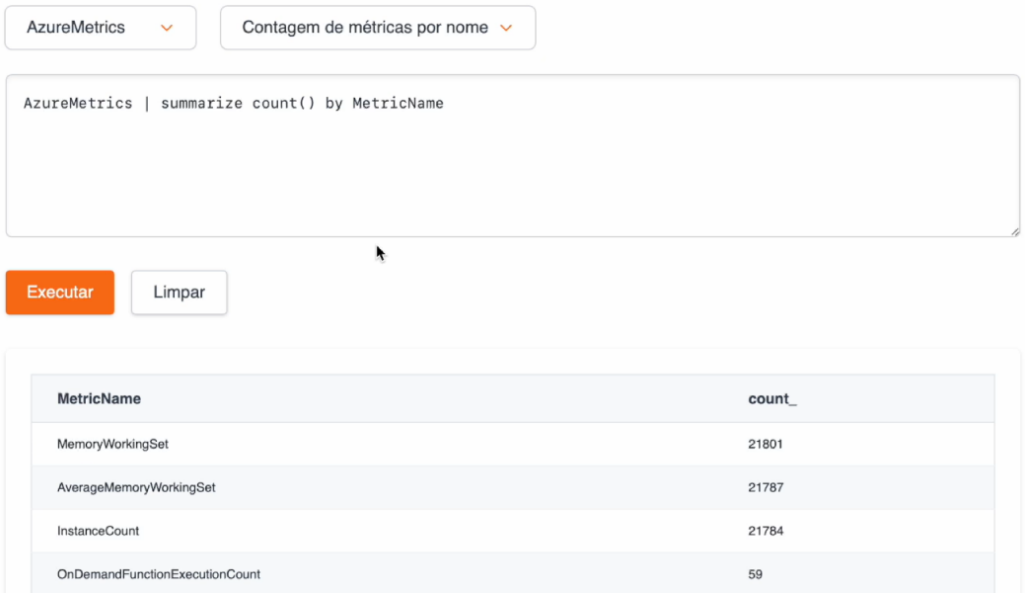


Fig. 4. Interface da camada de abstração de consultas KQL

4.4 Testes da aplicação

Os *endpoints* foram testados com **Postman** e cobertos por testes unitários focados na lógica de serviços. Os restantes cenários foram validados pela UI de forma a verificar se os resultados eram os esperados, assegurando uma experiência consistente sobre dados operacionais e financeiros provenientes das APIs Azure. A robustez da solução foi garantida através da execução de testes unitários, de integração e *end-to-end* (*E2E*). Os testes unitários verificaram regras de negócio e transformação de dados, enquanto os de integração validaram a comunicação entre camadas e o consumo das APIs Azure. Os testes *E2E* reproduziram fluxos reais, desde a autenticação até à visualização de métricas e custos, assegurando o correto funcionamento do sistema como um todo. No total, 112 testes unitários atingiram 77% de cobertura do código, de forma a cobrir os principais fluxos críticos e deixando por testar código que não tem lógica de negócio isolável.

O processo de integração contínua, implementado via GitHub Actions, automatizou a execução dos testes e o *build* do projeto, permitindo deteção precoce de problemas.

4.5 Avaliação e Resultados da Solução

A validação da solução proposta foi conduzida segundo uma metodologia multidimensional, desenhada para aferir a robustez arquitetural e a eficácia operacional da centralização de dados. A estabilidade do sistema foi corroborada pelos testes descritos na secção anterior.

Do ponto de vista da engenharia, a eficácia da adoção da *Clean Architecture* foi notória. Verificou-se que a utilização consistente de objetos de transferência de dados (DTOs) assegurou um desacoplamento efetivo entre o domínio e a infraestrutura, facilitando a manutenção evolutiva sem regressões. Simultaneamente, os testes de segurança ratificaram a integridade do controlo de acessos (RBAC), demonstrando que o mecanismo de autenticação garante o isolamento estrito de *tenants* e preserva a confidencialidade dos dados financeiros durante todo o ciclo de vida das transações HTTP.

Em cenário de produção, a API evidenciou tempos de resposta compatíveis com os requisitos de interatividade, suportando uma navegação fluida e a renderização dinâmica de visualizações complexas. Mais relevante, a centralização de métricas e custos numa interface unificada permite uma grande redução de esforço na correlação entre os indicadores técnicos e impacto orçamental. Assim, os resultados evidenciam a legitimidade e a viabilidade da abordagem proposta.

5 Conclusões

O projeto comprovou a viabilidade de uma solução integrada de governança em **Azure**, focada na monitorização de recursos e no controlo de custos. Os objetivos foram alcançados, incluíram: autenticação via Azure AD; listagem e pesquisa de recursos; consulta de métricas com diferentes agregações; execução de *queries KQL*; e análise de custos com filtros e agrupamentos.

A centralização de métricas, *logs* e custos numa única interface eliminou a necessidade de navegação entre múltiplos serviços, facilitando a leitura conjunta dos indicadores técnicos e do impacto financeiro. A experiência de utilização revelou-se fluida, com visualizações interativas e mecanismos de filtragem intuitivos, mantendo coerência com a identidade da organização.

Em suma, a solução cumpre o propósito definido: simplifica a monitorização e gestão de recursos em Azure, acelera a obtenção de *insights* e apoia a tomada de decisão. Os resultados sustentam a adoção desta abordagem como base sólida para evoluções futuras no contexto de governança em *cloud*.

Identifica-se como limitação funcional a ausência de um sistema automatizado de alertas que notifique os administradores, em tempo real, quando determinados limiares orçamentais são ultrapassados.

Como linhas de investigação futura, perspetiva-se colmatar esta lacuna e integrar modelos de Inteligência Artificial para a deteção automática de anomalias, permitindo evoluir para uma governança pró-ativa e preditiva.

Referências

1. W. A. Al-Sarayreh e M. A. Al-Qudah, "The Role of FinOps in Large-Scale Cloud Cost Optimization," *ResearchGate*, 2024.
2. S. R. Islam, *et al.*, "Cloud Cost Management and Optimization," *ResearchGate*, 2024.
3. J. Smith e A. Doe, "Machine Learning for Intelligent Cloud Cost Anomaly Detection," *ResearchGate*, 2024.
4. K. Patel e R. Singh, "Policy-Based Cloud Cost Control through Intelligent Automation," *ResearchGate*, 2024.
5. S. K. Singh, *et al.*, "Cloud Resource Demand Prediction using Machine Learning in the Context of QoS Parameters," *ResearchGate*, 2021.
6. M. Al-Roomi, *et al.*, "AWS PredSpot: Machine Learning for Predicting the Price of Spot Instances in AWS Cloud," *ResearchGate*, 2022.
7. Microsoft Docs. (2023). *What is Azure Monitor?* Disponível em: https://learn.microsoft.com/en-us/azure/azure-monitor/overview
8. Microsoft Docs. (2023). *What is Log Analytics in Azure?* Disponível em: https://learn.microsoft.com/en-us/azure/azure-monitor/logs/log-analytics-overview
9. Microsoft Docs. (2023). *Azure Cost Management and Billing documentation.* Disponível em: https://learn.microsoft.com/en-us/azure/cost-management-billing/
10. Turbo360. (2025). *Azure Cost Management Tool to Analyze & Optimize Cost Spend.* Disponível em: https://turbo360.com/azure-cost-analysis
11. IBM Cloudability. (2025). *Cloud Cost Management & Optimization Solutions.* Disponível em: https://www.apptio.com/products/cloudability/
12. Daily.dev. (2024). *Backend Frameworks List Choosing the Right One*. Disponível em: https://daily.dev/blog/backend-frameworks-list-choosing-the-right-one
13. Space-O Technologies. (2024). *Top Frontend Frameworks Comparison: Which One to Choose in 2024?* Disponível em: https://www.spaceotechnologies.com/blog/front-end-frameworks-comparison/

Sistema de Manutenção Proativa e Resposta a Incidentes de Segurança

Diogo Nunes[1] and F. Jorge Duarte[1]

ISEP - IPP, Porto 1211895,fjd@isep.ipp.pt

Abstract. Este artigo descreve o desenho, a implementação e a avaliação de uma plataforma de monitorização e resposta automática a vulnerabilidades e incidentes operacionais. A solução colmata a ausência de mecanismos automatizados e auditáveis para (i) publicação de páginas de manutenção, (ii) encerramento controlado da aplicação e (iii) notificação estruturada perante quebras de segurança ou picos de utilização. Integra uma *pipeline* CI/CD (Continuous Integration and Continuous Deployment) que, a cada duas horas, analisa dependências NuGet e npm, normaliza resultados e publica-os no Azure Log Analytics; consoante a severidade, uma Azure Logic App notifica a equipa e pode acionar o encerramento do Azure App Service. No perímetro, o Azure Front Door, protegido por WAF (Web Application Firewall), gere o encaminhamento entre a aplicação e páginas de manutenção alojadas em Azure Storage. Em complemento, o Azure Application Insights e o Azure Monitor recolhem telemetria e métricas de CPU/RAM, originando alertas que garantem visibilidade operacional contínua. Na avaliação, verificou-se que os *health probes* a cada 2 s suportaram *failover* observado de ∼15 s, com limite alvo inferior a 30 s, os ataques simulados foram bloqueados (HTTP 403) e os relatórios de vulnerabilidades foram ingeridos e distribuídos com fiabilidade. A abordagem demonstra viabilidade de resposta proativa, escalável e alinhada com práticas DevSecOps (Development, Security and Operations) em ambientes *cloud-native*.

Keywords: Monitorização · Observabilidade · Resposta automática a incidentes · DevSecOps · Azure Front Door · Azure Monitor

1 Introdução

A digitalização acelerada e a centralização de serviços web elevaram os requisitos de disponibilidade, segurança e observabilidade. Qualquer falha ou ataque pode traduzir-se em impactos operacionais, legais e reputacionais. É, por isso, essencial monitorizar, antecipar e reagir a comportamentos inesperados em produção com mecanismos automatizados e auditáveis.

Apresenta-se o desenho e a implementação de um sistema de manutenção proativa e resposta automatizada a incidentes de segurança em aplicações web. A solução assenta em serviços Microsoft Azure e práticas DevSecOps (Development, Security and Operations). A validação foi feita com uma prova de conceito composta por *backend* em ASP.NET Core e *frontend* em Angular, o que permitiu recolher telemetria real e exercitar cenários de falha e de ataque.

1.1 Descrição do Problema

Não existiam, no contexto que motivou este trabalho, mecanismos integrados que assegurassem, de forma autónoma e auditável, a deteção atempada de anomalias e vulnerabilidades, o encerramento controlado da aplicação quando necessário, a publicação de páginas de manutenção (planeada e de emergência) e a notificação estruturada da equipa técnica. Pretendia-se reduzir MTTD (Mean Time to Detection) e MTTR (Mean Time to Resolution), diminuir intervenção manual em momentos críticos e preservar a experiência do utilizador.

1.2 Objetivos

Apresentam-se de seguida os principais objetivos do trabalho:

- Integrar Application Insights e Azure Monitor para recolha de telemetria e análise contínua de fiabilidade e desempenho.
- Configurar alertas acionáveis (recursos e segurança), com notificações automáticas.
- Operar páginas de manutenção (planeada/emergência) com ativação controlada e comunicação clara.
- Implementar respostas automáticas (Logic Apps/Functions), incluindo encerramento seguro da aplicação perante vulnerabilidades críticas.
- Validar a solução com testes funcionais, de carga e cenários de ataque.

2 Estado da Arte

A solução proposta insere-se na confluência de observabilidade em ambientes *cloud-native*, automatização da resposta a incidentes e integração de práticas DevSecOps nas *pipelines* de desenvolvimento e operação. Esta secção sintetiza os principais trabalhos e tecnologias relevantes que enquadram o desenho adotado.

2.1 Trabalhos Relacionados

A investigação recente em operações em *cloud* tem procurado aumentar a visibilidade ponta a ponta, detetar anomalias mais cedo e reduzir os tempos de deteção e resolução de incidentes. A linha comum passa por instrumentação consistente, modelos de deteção cada vez mais robustos e automação criteriosa da resposta [1,2,5].

Na observabilidade, a correlação de *logs*, *traces* e *metrics* tornou-se prática corrente, suportada por instrumentação aberta como OpenTelemetry e por abordagens de análise comportamental UEBA (User and Entity Behavior Analytics). Conjugam-se técnicas estatísticas com modelos de aprendizagem para identificar desvios [3]. Continuam, porém, desafios de calibração e de controlo do ruído de alertas [2].

Em AIOps (Artificial Intelligence for IT Operations), a aplicação de métodos de ML (Machine Learning) e NLP (Natural Language Processing) ajuda a

correlacionar eventos, reduzir redundâncias e priorizar alertas, com ganhos reportados na redução de MTTD/MTTR [1,2]. Ganham relevância modelos preditivos para antecipar incidentes, mas persistem necessidades de explicabilidade e adaptação contínua a contextos dinâmicos [2].

Na resposta automatizada a incidentes SOAR (Security Orchestration, Automation and Response), generalizou-se o uso de *playbooks* para tarefas repetitivas de isolamento, bloqueio e recolha de evidências. A prática dominante é a automação assistida: automatizar o que é rotineiro, mantendo aprovação humana quando as ações são disruptivas, assegurando auditoria e rastreabilidade [5].

Quanto à resiliência e manutenção, destacam-se o recurso a WAF, *health probes* frequentes e *failover* para páginas estáticas fora da aplicação, bem como padrões ao nível da aplicação, como *circuit breakers*, *fallbacks* e *feature flags* [7]. A eficácia depende de limiares e temporizações bem afinados para evitar oscilações e proteger a experiência do utilizador.

Por fim, a integração de práticas DevSecOps nas *pipelines* CI/CD tornou-se transversal, combinando SAST (Static Application Security Testing), DAST (Dynamic Application Security Testing) e SCA (Software Composition Analysis) com análises de IaC (Infrastructure as Code), SBOMs (Software Bill of Materials) e PaC (Policy as Code) para reforçar a governança. Modelos de maturidade apoiam uma adoção faseada, especialmente relevante em organizações com recursos limitados [6].

2.2 Tecnologias Existentes

As três principais plataformas *cloud-native*, Microsoft Azure, Amazon Web Services (AWS) e Google Cloud Platform (GCP), oferecem capacidades convergentes para observabilidade, segurança e automação, com diferenças de integração e governança [8].

Na observabilidade e monitorização, o Azure integra *logs*, *metrics*, alertas e *tracing* distribuído em Monitor e Application Insights. Inclui KQL (Kusto Query Language) e Smart Detection. A AWS combina CloudWatch, X-Ray e CloudTrail numa abordagem modular. O GCP agrega Cloud Monitoring, Cloud Logging e Cloud Trace, com suporte nativo a OpenTelemetry [4,8].

Na gestão de páginas de manutenção, no Azure é possível usar Deployment Slots do App Service e Traffic Manager. Também é comum servir páginas estáticas via Storage integrado com Front Door ou Application Gateway. Na AWS, a combinação típica é S3 com CloudFront, podendo existir resposta direta via ALB. No GCP, usa-se Cloud Storage com Cloud Load Balancer ou uma implementação dedicada no Cloud Run.

Na automação CI/CD e orquestração, o Azure disponibiliza Azure DevOps e GitHub Actions para *pipelines as code* e serviços *serverless* como Logic Apps e Functions. Na AWS, o conjunto inclui CodeCommit, CodeBuild, CodePipeline e CodeDeploy. No GCP, Cloud Build e Cloud Deploy integram-se de forma estreita com GKE e Cloud Run.

2.3 Síntese das Escolhas Tecnológicas

A opção por **Microsoft Azure** reflete o alinhamento com a infraestrutura existente da organização (políticas de segurança e pipelines já definidos) e a integração nativa necessária para observabilidade e resposta automatizada (Monitor, Application Insights, Log Analytics, Front Door/WAF, Logic Apps). Ao nível da *stack*, adotou-se **ASP.NET Core** (*backend*), **Angular** (*frontend*) e **SQL Server** (dados), privilegiando documentação, integração com o ecossistema e produtividade.

3 Análise e Desenho da Solução

Esta secção apresenta o enquadramento do problema, a definição dos requisitos e o desenho arquitetural que suporta monitorização, segurança e continuidade de serviço na aplicação.

3.1 Domínio do Problema

A organização opera uma aplicação web em Microsoft Azure com requisitos elevados de disponibilidade e segurança. É necessário detetar e responder a incidentes com rapidez, bloquear tráfego malicioso no perímetro e mitigar o risco contínuo de vulnerabilidades em dependências NuGet/npm. O problema a resolver consiste em garantir visibilidade ponta a ponta e resposta automática em produção: filtrar e encaminhar pedidos no perímetro, monitorizar aplicação e infraestrutura com alertas de baixo ruído, gerir continuidade através de páginas de manutenção com comutação rápida e automatizar a triagem de vulnerabilidades com ação preventiva quando crítico. O objetivo é reduzir MTTD e MTTR e preservar a experiência do utilizador mesmo perante falhas ou ameaças.

3.2 Engenharia de Requisitos

O levantamento de requisitos foi estruturado de acordo com o modelo FURPS+ [9]. Esta abordagem permite sintetizar de forma clara os objetivos operacionais e não funcionais que a solução deve cumprir, conforme se apresenta de seguida:

– **Funcionalidade (F)/Requisitos Funcionais (RF): RF1** bloquear automaticamente pedidos maliciosos com base em regras WAF; **RF2** alertar a equipa quando um pedido malicioso é bloqueado; **RF3** executar periodicamente a *pipeline* que analisa vulnerabilidades no *backend* e *frontend*; **RF4** registar todas as vulnerabilidades detetadas para auditoria; **RF5** notificar por alerta e email quando forem detetadas vulnerabilidades; **RF6** desligar automaticamente a aplicação quando existirem vulnerabilidades críticas no *backend*; **RF7** notificar com justificação técnica sempre que a aplicação for desativada; **RF8** redirecionar para página de manutenção de emergência quando a aplicação está indisponível; **RF9** Ativação manual de manutenção

planeada com definição de IPs autorizados; **RF10** assegurar acesso normal quando não houver manutenção ativa; **RF11** alertar e registar eventos quando limites de CPU/RAM forem ultrapassados; **RF12** detetar anomalias via Application Insights e alertar a equipa; **RF13** registar pedidos maliciosos bloqueados com detalhes para auditoria.
- **Usabilidade (U): U1** mensagens claras nas páginas de manutenção; **U2** relatórios legíveis por email.
- **Fiabilidade (R): R1** redirecionar automaticamente para página de emergência em falha.
- **Desempenho (P): P1** *failover* entre aplicação e emergência $< 30\,\text{s}$.
- **Suporte/Manutenção (S): S1** passos para ativar manutenção planeada claramente documentados.
- **Segurança (X): X1** apenas IPs autorizados acedem em manutenção; **X2** infraestrutura isolada (VNet + Private Endpoints); **X3** vulnerabilidades registadas centralmente; **X4** pedidos maliciosos bloqueados antes da aplicação.
- **Restrições (C): C1** infraestrutura integralmente em Microsoft Azure.

3.3 Arquitetura da Solução (modelo C4)

Adotou-se o modelo C4 [10] para a documentação arquitetural. Esta secção apresenta a vista lógica do Nível 2 (Contentores), ilustrada na Fig. 1, evidenciando o percurso do tráfego e a separação de responsabilidades. No perímetro, o Azure Front Door com WAF filtra e encaminha pedidos. A aplicação (Web App) encontra-se alojada em App Service e persiste dados em Azure SQL Server Database. As páginas de manutenção *Scheduled Maintenance Page* e *Emergency Maintenance Page*, residem em Storage Accounts e suportam *failover*/*fallback* rápidos. A observabilidade é assegurada por Application Insights, Azure Monitor e Log Analytics. A *pipeline* de CI/CD (Azure Pipelines) integra-se com o App Service e com o Log Analytics, automatizando a análise de vulnerabilidades e a disponibilização de *logs*. As respostas automáticas (notificação e, quando aplicável, encerramento controlado da aplicação) são orquestradas por uma Azure Logic App.

Alternativas consideradas

- **Páginas de manutenção no *frontend***: solução simples, mas acopla o *failover* à própria aplicação e exigiria novos *deploys* para ajustes.
- **Separar *frontend* e *backend* em App Services distintos**: aumentaria a escalabilidade e isolamento, mas também a complexidade operacional.
- **Application Gateway vs Front Door**: optou-se por Front Door pelo encaminhamento global, gestão centralizada de regras e facilidade na aplicação de listas de permissões em manutenção.

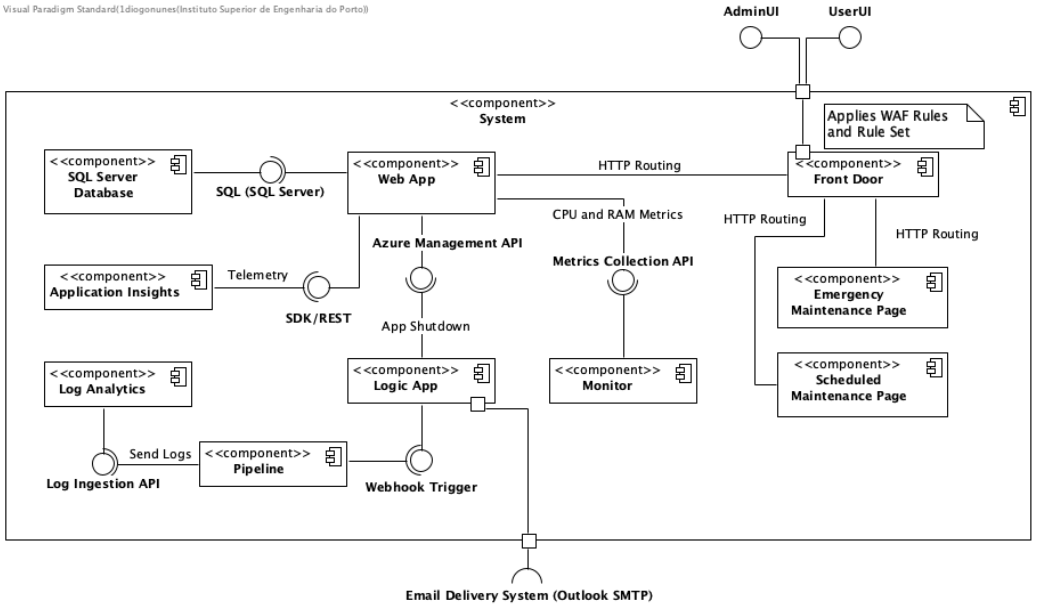


Fig. 1. Diagrama de vista lógica — Nível 2.

4 Implementação da Solução

Com base no desenho arquitetural, esta secção detalha a concretização da solução na plataforma Microsoft Azure, incluindo topologia arquitetural, *pipeline* de vulnerabilidades e mecanismos de monitorização e manutenção.

4.1 Descrição da Implementação

Topologia e Recursos da Infraestrutura A solução foi implementada na Microsoft Azure com isolamento de rede em Virtual Network e comunicação privada entre serviços por Private Endpoints. O ponto de entrada é o Azure Front Door Premium com WAF, responsável por filtragem e encaminhamento global. A aplicação (Angular no *frontend* e ASP.NET Core no *backend* é alojada em App Service e persiste dados em Azure SQL Database com acesso privado. As páginas de manutenção (planeada e de emergência) são servidas a partir de Azure Storage Accounts. A observabilidade integra Application Insights, Azure Monitor e Log Analytics, garantindo telemetria e alertas centralizados.

Pipeline de Análise e Resposta a Vulnerabilidades A *pipeline* CI/CD (Azure Pipelines) corre de 2 em 2 horas, auditando dependências do *backend* (NuGet) e do *frontend* (npm), consolidando resultados e enviando-os para Log Analytics. Consoante a gravidade das vulnerabilidades, desencadeia notificações e, quando aplicável, encerramento controlado do serviço.

Stage 1 — Análise de vulnerabilidades no *backend*. Nesta fase é feita uma auditoria periódica a pacotes .NET, produzindo um relatório JSON com a lista de vulnerabilidades e respetivas severidades. A Fig. 2 ilustra, de forma sintetizada, as etapas principais desta fase da *pipeline*.

Fig. 2. Etapas principais da Stage 1 da *pipeline* de vulnerabilidades (*backend*).

Stage 2 — Análise de vulnerabilidades no *frontend*. Nesta fase é realizada a auditoria a pacotes npm, gerando um relatório JSON que inclui apenas vulnerabilidades com severidade *High* ou superior. A Fig. 3 apresenta as etapas principais desta fase da *pipeline*.

Fig. 3. Etapas principais da Stage 2 da *pipeline* de vulnerabilidades (*frontend*).

Stage 3 — Execução do `log-vulnerabilities.js`. Nesta fase é executado o *script* de consolidação e encaminhamento: normaliza campos, agrega resultados e publica no Log Analytics, acionando também a Logic App com o relatório consolidado. A *pipeline* termina em `failed` sempre que existam vulnerabilidades detetadas (todas no *backend*; $\geq$ High no *frontend*), forçando triagem. A Fig. 4 ilustra as etapas principais desta fase da *pipeline*.

Fig. 4. Etapas principais da Stage 3 da *pipeline* de vulnerabilidades (consolidação e resposta).

Gestão de segredos na *pipeline*. As credenciais e parâmetros sensíveis são mantidos como *secrets* e injetados em tempo de execução como variáveis de ambiente, com permissões mínimas via RBAC (Role-Based Access Control). Os *secrets* são `AZURE_CLIENT_ID`, `AZURE_CLIENT_SECRET`, `AZURE_TENANT_ID`, `LOGS_INGESTION_ENDPOINT`, `DATA_COLLECTION_RULE_ID`, `STREAM_NAME` e `LOGICAPP_CALLBACK_URL`.

Funcionamento interno do `log-vulnerabilities.js`. O *script* lê os relatórios NuGet e npm, uniformiza campos (pacote, versão, severidade, origem, *timestamp*), aplica um mapeamento comum de severidades, remove duplicados e enriquece registos com metadados de execução (*pipeline*, *stage*, *commit*). Em seguida, produz *payloads* JSON para ingestão analítica e *payload* de notificação para a Logic App.

Ingestão de logs no Log Analytics. A ingestão é feita via Logs Ingestion API, através de um Data Collection Endpoint (DCE) e de uma Data Collection Rule (DCR) que mapeiam os registos para a tabela personalizada `Vulnerabilities_CL`. Falhas temporárias originam novas tentativas. Os dados ficam imediatamente disponíveis para KQL e para as regras de alerta.

Alertas baseados em *logs* de vulnerabilidades. A regra "PESTI Pipeline Vulnerability Scan Detected" avalia a tabela `Vulnerabilities_CL` numa janela

móvel de 15 min e dispara quando existe pelo menos um registo novo, independentemente da severidade ou da origem (NuGet/npm). Emite notificação no portal e por email através do grupo de ação.

Alerta de vulnerabilidades críticas no *backend*. A regra "PESTI Critical NuGet Vulnerabilities Detected" filtra a `Vulnerabilities_CL` por origem NuGet e severidade `CRITICAL` na mesma janela de 15 min. Para além da notificação no portal e por email, invoca a Logic App para resposta automatizada.

Resposta automatizada via Logic App. A Logic App recebe o relatório consolidado, transforma-o em tabela HTML e envia *email* detalhado. Se forem detetadas vulnerabilidades *Critical* no *backend*, executa o encerramento controlado da aplicação, enviando um segundo *email* com justificação e mantendo registos para auditoria.

Gestão de Pedidos e Mecanismos de Manutenção O Front Door aplica primeiro as regras do WAF (bloqueio a 403 de tráfego malicioso) e, consoante o estado das origens e a manutenção ativa, encaminha para a aplicação ou para as páginas estáticas no Storage, preservando a experiência do utilizador.

Alerta de blocos críticos no WAF. Existe uma regra que avalia minuto a minuto os registos do WAF com Inbound Anomaly Score crítico; agrega a referência, IP e regras acionadas, enviando notificação imediata para triagem.

Gestão de páginas de manutenção. Em manutenção planeada, uma Rule Set aplica *whitelisting* por IP e redireciona o restante tráfego para o *static website* em Storage. Antes de ativar, edita-se o HTML com janela temporal e contactos. Em falha inesperada, o *failover* para a página de emergência ocorre automaticamente.

Monitorização e Alertas com Azure Monitor e Application Insights Foram configurados alertas de CPU e RAM com limiar de 80%, janela de 5 min e reavaliação a cada minuto. No Application Insights está ativa a Smart Detection (anomalias de exceções, falhas, latência, dependências, memória e *traces*) para deteção precoce e diagnóstico.

4.2 Testes

A validação combinou testes manuais e automatizados de forma a confirmar o cumprimento dos requisitos da solução. Nos manuais, confirmou-se a manutenção planeada com IP *whitelisting*, o *failover* para a página de emergência com comutação observada próxima de 15 s e o *fallback* automático após recuperação. O intervalo observado é consistente com os *health probes*, sustentando *failover* $<$ 30 s. Verificou-se ainda o ciclo completo de vulnerabilidades (detetar $\rightarrow$ ingerir $\rightarrow$ notificar $\rightarrow$ encerrar quando crítico).

Nos automatizados, uma *collection* do Postman validou os bloqueios 403 para *SQL Injection*, *Cross-Site Scripting* e *Command Injection* e corroboração em Log Analytics. Um teste de carga (Azure Load Testing) provocou a ativação dos alertas de CPU/RAM, com limiares de 80% (média 5 min; reavaliação a cada 1 min).

4.3 Avaliação da Solução

Os resultados cumprem os critérios definidos. Face às métricas estabelecidas (CPU/RAM a 80% e *failover* $< 30\,\text{s}$), o *failover* ficou em torno de 15 s e o retorno automático foi estável. Na vertente de segurança no perímetro, foram simulados ataques de *SQL Injection*, *Cross-Site Scripting* (XSS) e *Command Injection*, com múltiplos *payloads* por tipo de ataque. Em todos os casos, os pedidos maliciosos foram bloqueados com `HTTP 403` pelo WAF e ficaram registados para análise em Log Analytics; numa execução adicional apenas com tráfego legítimo, não se observaram blocos indevidos, evidenciando ausência de falsos positivos no conjunto de testes efetuado. Sob carga, os limiares de CPU/RAM foram atingidos como previsto e os alertas emitidos apresentaram baixo ruído.

A *pipeline* com cadência de 2 h detetou e ingeriu vulnerabilidades na tabela `Vulnerabilities_CL`. As regras acionaram as notificações e, quando a severidade foi *Critical* no *backend*, ocorreu encerramento controlado com justificação, incluindo alertas baseados em *logs* (p. ex., WAF *Critical Blocks* e *Vulnerabilities Detected*). Estes resultados confirmam que os mecanismos de monitorização, alerta e resposta automática se comportam de forma consistente com os requisitos definidos.

5 Conclusões

Nesta secção apresentam-se os principais resultados obtidos, discutindo em que medida os objetivos foram alcançados e apontando limitações e possíveis linhas de evolução futura.

5.1 Objetivos Concretizados

A implementação cumpriu os objetivos definidos para o projeto. Segue uma lista dos resultados alcançados:

- **Observabilidade**: integração de Azure Monitor e Application Insights para telemetria em tempo real e deteção inteligente de anomalias.
- **Alertas e notificações**: configuração de alertas para recursos e vulnerabilidades (CPU/RAM a 80% com janela de 5 minutos e avaliação a cada 1 minuto e regras KQL sobre `Vulnerabilities_CL`), com notificações acionáveis.
- **Continuidade de serviço**: disponibilização de páginas de manutenção planeada e de emergência, com *failover* validado em cerca de 15 segundos e *fallback* automático.
- **Resposta automática a incidentes**: *pipeline* periódica (cadência de 2 horas), ingestão no Log Analytics e orquestração via Logic App para notificação e encerramento controlado quando crítico no *backend*.
- **Documentação e validação**: processo documentado e testes manuais e automatizados a corroborar os requisitos.

5.2 Limitações e Trabalho Futuro

Foram observadas limitações operacionais. A propagação de alterações no Azure Front Door é lenta (2–10 minutos) e afeta iterações rápidas. A página de manutenção planeada exige edição manual e fica sujeita a erro humano. Para enriquecer as notificações é preciso recorrer à Logic App, porque o formato nativo é rígido. A *pipeline* ainda não integra testes unitários. Estas questões não invalidam os resultados, mas aumentam a complexidade operacional e justificam mitigação.

Como linhas de evolução, propõe-se automatizar a manutenção planeada, incluindo a edição da página e a gestão da Rule Set. Recomenda-se considerar um *token* de acesso temporário além do IP *whitelisting*. Deve avaliar-se uma abordagem híbrida de perímetro (Front Door + Application Gateway) para reduzir ainda mais o tempo de transição. Importa integrar testes automatizados na *pipeline*. Por fim, considera-se relevante analisar a adoção de ferramentas complementares de SCA (Software Composition Analysis) e QA (Quality Assurance), como Dependabot, SonarQube e Snyk.
Em síntese, a solução cumpriu os objetivos de desempenho, segurança e disponibilidade, demonstrando a eficácia de uma abordagem DevSecOps proativa em ambiente Azure.

References

1. Notaro, P., Cardoso, J., Gerndt, M.: A Systematic Mapping Study in AIOps. In: ICSOC 2020 Workshops, LNCS 12632, pp. 110–123 (2021). https://api.semanticscholar.org/CorpusID:229210898
2. Remil, Y., Bendimerad, A., Mathonat, R., Kaytoue, M.: AIOps Solutions for Incident Management: Technical Guidelines and A Comprehensive Literature Review. arXiv:2404.01363 (2024). https://arxiv.org/abs/2404.01363
3. Wang, X. et al.: Graph-based Anomaly Detection and Root Cause Analysis for Microservices in Cloud-Native Platform. In: 2024 IEEE SustainCom (2024). https://api.semanticscholar.org/CorpusID:276935952
4. Kratzke, N.: Cloud-Native Observability: The Many-Faceted Benefits of Structured and Unified Logging—A Multi-Case Study. Future Internet 14(10), 274 (2022). https://doi.org/10.3390/fi14100274
5. Karlzén, H., Sommestad, T.: Automatic Incident Response Solutions: A Review of Proposed Solutions' Input and Output. In: ARES 2023, pp. 37:1–37:9 (2023). https://doi.org/10.1145/3600160.3605066
6. Cheenepalli, J., Hastings, J.D., Ahmed, K.M., Fenner, C.: Advancing DevSecOps in SMEs: Challenges and Best Practices for Secure CI/CD Pipelines. arXiv:2503.22612 (2025). https://arxiv.org/abs/2503.22612
7. Microsoft Azure: Circuit Breaker pattern. https://learn.microsoft.com/en-us/azure/architecture/patterns/circuit-breaker
8. Google Cloud: Google Cloud Platform Comparison with AWS and Azure. https://cloud.google.com/docs/get-started/aws-azure-gcp-service-comparison
9. Grady, R.B.: Practical Software Metrics for Project Management and Process Improvement. Prentice Hall (1992).
10. Brown, S.: C4 Model for Visualising Software Architecture. https://c4model.com

Power BI Conversacional com RAG

João Pedro Sá Fernandes[1], F. Jorge Duarte[1], and David Mota[2]

ISEP/IPP - DevScope, Porto
1211681[1],fjd[1]@isep.ipp.pt david.mota@devscope.net[2]

Resumo O crescimento do uso do Microsoft Power BI multiplicou o número de relatórios estratégicos, mas o acesso tradicional via *dashboards* mantém uma barreira de literacia em BI. Este artigo sintetiza o projeto de um protótipo de assistente conversacional que permite interrogar relatórios Power BI em linguagem natural com base numa arquitetura RAG, em contexto empresarial. A solução automatiza a exportação para PDF, armazena os ficheiros no Azure Blob Storage, extrai informações dos mesmos, gera *embeddings* (ChromaDB) e fundamenta respostas com LLM do Azure OpenAI, orquestrada por FastAPI e LangChain, com autenticação corporativa via Azure AD/MSAL. Ensaios conceptuais realizados com dados reais indicam respostas em $< 4\,\mathrm{s}$, redução substancial do tempo de procura de métricas e elevada satisfação dos utilizadores, com citação explícita dos trechos de origem, caracterizando o trabalho como prova de conceito com potencial de validação futura.

Palavras-chave Business Intelligence Conversacional · Power BI · Chatbots · Retrieval-Augmented Generation (RAG) · Azure OpenAI · Inteligência Artificial

1 Introdução

As organizações recorrem a Power BI para análise e decisão, mas o acesso por *dashboards* continua pouco interativo para não especialistas. Pretende-se um fluxo *end-to-end* que exporte, indexe e recupere conteúdo de relatórios, devolvendo respostas auditáveis em linguagem natural. A abordagem adotada conjuga recuperação vetorial e geração (RAG), com armazenamento em Azure Blob, núcleo em FastAPI/LangChain, *vector store* ChromaDB e autenticação MSAL.

1.1 Descrição do problema

Sem integração direta e automatizada entre Power BI e um sistema conversacional, o acesso à informação depende de navegação manual e de equipas de BI, reduzindo a eficiência no dia-a-dia e dificultando respostas rápidas e fundamentadas.

1.2 Objetivos

O objetivo principal deste trabalho é automatizar a extração de relatórios Power BI para a *cloud* e disponibilizá-los a um LLM via RAG, permitindo perguntas em linguagem natural com respostas precisas e citadas. Os objetivos específicos foram:

- Exportação automática de relatórios para PDF;
- Ingestão, *chunking* e *embeddings* persistidos em ChromaDB;
- Recuperação semântica e geração com *prompt* conservador;
- UI *web* com SSO corporativo;
- Avaliação funcional e de desempenho.

2 Estado da Arte

A literatura recente em BI conversacional destaca o valor de interfaces naturais para democratizar o acesso a informação analítica, reduzindo a dependência de literacia em ferramentas de BI e de equipas especializadas [2]. Em particular, assistentes conversacionais ligados a *dashboards* permitem consultar relatórios e indicadores em linguagem natural, aproximando as perguntas dos utilizadores da forma como pensam o negócio [1,2]. No entanto, vários estudos apontam limitações persistentes em atualização contínua de dados, precisão das respostas e mecanismos de segurança e governança [2,5].

O paradigma Retrieval-Augmented Generation (RAG) surgiu como resposta a algumas destas lacunas: combina modelos de linguagem com mecanismos de recuperação de documentos, ancorando as respostas em evidências concretas e mitigando alucinações típicas de LLMs [3]. Em cenários empresariais, em que a veracidade e a rastreabilidade são essenciais, esta abordagem tem ganho tração, embora ainda com pouca documentação detalhada em contextos concretos de Power BI.

2.1 Integração de um *chatbot* com Power BI

A Microsoft tem investido na integração entre soluções conversacionais e Power BI, nomeadamente através de Power Virtual Agents/Copilot Studio e Microsoft Bot Framework [1,4]. Estes esforços procuram tornar os dados empresariais acessíveis de forma intuitiva, permitindo que utilizadores coloquem perguntas sobre *dashboards* e relatórios usando linguagem natural [2,5].

Apesar disso, muitas soluções descritas na literatura e em documentação técnica dependem ainda de processos *batch* ou semiautomáticos para exportar e sincronizar dados, recorrendo a scripts ou integrações pontuais [1,4]. Esta abordagem dificulta a escalabilidade, compromete a atualidade das respostas e raramente explicita como é feita a ligação entre o conteúdo dos relatórios e o texto gerado pelo LLM. Em particular, poucos trabalhos detalham estratégias de *grounding* semântico com citações das fontes, ou a utilização sistemática de arquiteturas RAG integradas com Power BI [3,5].

2.2 Lacunas identificadas

Com base nos trabalhos identificados com a análise do estado da arte, podem sintetizar-se quatro grupos principais de lacunas em BI conversacional com Power BI:

- **Atualização em tempo real**: grande parte das soluções depende de fluxos *batch* ou exportações periódicas, o que reduz a capacidade de responder com dados efetivamente atuais em cenários onde métricas e KPIs mudam rapidamente [2,5].
- **Precisão e contextualização**: muitos *chatbots* baseiam-se em FAQs ou regras fixas, com pouco entendimento contextual e sem ligação clara às fontes de dados; a adoção de RAG é frequentemente mencionada, mas pouco detalhada ao nível de implementação [2,3].
- **Automatização e CI/CD**: integrações mais antigas com Power BI recorrem a scripts ou processos manuais para exportação, o que dificulta a manutenção e a evolução para *pipelines* automatizadas e reprodutíveis [1,4,5].
- **Governança e segurança**: a integração de dados sensíveis de BI exige autenticação corporativa, controlo de acessos, encriptação e auditoria, mas muitos exemplos práticos abordam estes temas de forma superficial [7,4].

Estas lacunas motivam a necessidade de soluções que combinem chatbots, Power BI e RAG com maior preocupação por atualização contínua, transparência das respostas e operacionalização em ambiente empresarial.

2.3 Tecnologias existentes e opções adotadas

Diversas tecnologias suportam a construção de arquiteturas RAG no ecossistema Microsoft. Foram analisadas alternativas para cada componente e justificada a escolha adotada no protótipo:

- **Extração automatizada e *backend***: consideraram-se Azure Functions, GitHub Actions/Azure DevOps e serviços FastAPI expostos via HTTP. Optou-se por FastAPI por concentrar a lógica de exportação e ingestão num único serviço leve, fácil de testar localmente e acionável a partir de outras ferramentas de automatização [6].
- **Armazenamento de documentos**: consideram-se Azure Blob Storage, Azure Data Lake Gen2 e Azure Files. Escolheu-se Azure Blob Storage por oferecer um modelo simples de ficheiros, encriptação nativa e custos reduzidos, adequado ao volume de relatórios previsto [7].
- **Indexação vetorial**: foram avaliadas ChromaDB, FAISS e Weaviate. A opção recaiu em ChromaDB pela integração direta com LangChain, facilidade de utilização e adequação a um protótipo com volume moderado de dados, mantendo a possibilidade de evoluir para soluções mais distribuídas no futuro [8,10].
- **LLM e *embeddings***: seguindo as recomendações da própria Microsoft, utilizaram-se modelos do Azure OpenAI com RAG, combinando *embeddings* para recuperação semântica com modelos generativos para compor respostas em linguagem natural [3].

– ***Frontend* e autenticação**: a camada de interface foi implementada em React com MSAL/Azure AD, garantindo *Single Sign-On* corporativo e integração com o restante ecossistema Microsoft, com possibilidade de publicação em Teams/Outlook via Bot Framework [4,5].

3 Análise e Desenho da Solução

Esta secção faz a ponte entre o problema identificado e a arquitetura proposta, detalhando o modelo de domínio, o fluxo *end-to-end*da solução e as principais componentes que suportam a abordagem RAG com Power BI.

3.1 Descrição geral da solução

A solução organiza-se num fluxo automatizado que exporta relatórios do Power BI para PDF através de scripts desenvolvidos em Python, guarda os ficheiros de forma centralizada no Azure Blob Storage, processa os documentos para extração de texto, criação de *embeddings* e indexação numa base vetorial (ChromaDB), aplica a arquitetura *Retrieval-Augmented Generation* (RAG), com fases de recuperação de contexto e geração de respostas, e disponibiliza as respostas aos utilizadores através de uma aplicação *web* em React.

Esta cadeia garante que as respostas do assistente conversacional não são apenas geradas pelo LLM, mas ancoradas em evidência extraída de relatórios reais, reduzindo *alucinações* e permitindo auditoria posterior às fontes consultadas.

3.2 Modelo de domínio

O modelo de domínio inclui as entidades Utilizador, *Query*, ExportJob, Blob-Ficheiro, Documento, *Chunk*, *Embedding* e Resposta. Um ExportJob invoca a Power BI REST API; o PDF no Azure Blob origina um Documento do qual se extraem *chunks* com *embeddings*. A *Query* recupera *chunks* relevantes e compõe a Resposta citável, preservando a rastreabilidade *Query*→*Chunks*→Resposta.

A Fig. 1 sintetiza estas relações, evidenciando o fluxo lógico desde o pedido do utilizador até à geração de uma resposta fundamentada.

3.3 Componentes da arquitetura

A arquitetura materializa o fluxo descrito na Secção 3.2 num conjunto de componentes que suportam a extração automática de relatórios Power BI, a indexação semântica e a interação conversacional em linguagem natural. A exportação de relatórios Power BI é realizada por scripts em Python que invocam a Power BI REST API para gerar PDFs e acompanham o estado por *polling*, preparando a posterior ingestão. Os ficheiros gerados são guardados num repositório central em Azure Blob Storage, com encriptação em repouso e em trânsito e organização por *containers*, o que facilita a integração com outros serviços Azure. Sobre estes documentos atua o módulo de indexação vetorial com ChromaDB, responsável

pela extração de texto, *chunking*, geração de *embeddings* via Azure OpenAI e persistência na ChromaDB (através de LangChain) para suportar pesquisa semântica eficiente.

O núcleo RAG, suportado por Azure OpenAI, recupera os *chunks* mais relevantes na base vetorial e gera respostas em linguagem natural, privilegiando respostas conservadoras (não sei) quando não existem evidências suficientes. O *backend* em FastAPI expõe *endpoints* para ingestão e consulta, integra Azure Blob, ChromaDB e Azure OpenAI, aplica autenticação com JWT e regista *logs* estruturados para diagnóstico e auditoria. A camada de apresentação é assegurada por um *frontend* em React com MSAL, que disponibiliza uma interface de *chat web* responsiva, com *Single Sign-On* corporativo e comunicação com o *backend* via Axios, permitindo ao utilizador colocar perguntas em linguagem natural e visualizar respostas com citações.

Em conjunto, estes componentes respondem tanto aos requisitos funcionais de exportação, ingestão e consulta como aos requisitos não funcionais de desempenho, segurança e usabilidade identificados no modelo FURPS.

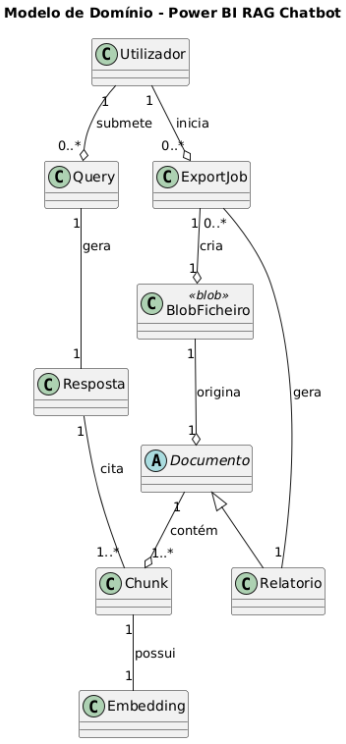


Fig. 1: Modelo de domínio do sistema RAG para Power BI.

3.4 Requisitos funcionais

Requisitos Funcionais (RF):

– RF1: Exportar automaticamente relatórios Power BI para PDF (REST API, *polling* de estado).

– RF2: Armazenar PDFs no Azure Blob com metadados (relatório, página, data).
– RF3: Ingestão: extração de texto, *chunking* e geração de *embeddings*; persistência em ChromaDB.
– RF4: RAG: recuperar trechos relevantes e gerar respostas citadas (não sei quando não tem evidências).
– RF5: Expor rotas *web* em FastAPI para ingestão/consulta com segurança corporativa.
– RF6: UI *web* com MSAL/*SSO*.

3.5 Visão arquitetural (C4)

A arquitetura é descrita combinando C4 e 4+1, para comunicar a solução em níveis progressivos (Contexto → Lógica/Implementação/Processos/Física) e garantir que as decisões técnicas respondem aos requisitos funcionais e não funcionais identificados. Esta articulação permite mostrar, de forma coerente, quem interage com o sistema, como os contentores principais se relacionam e como os fluxos críticos são executados e implantados.

Nível 2 Vista lógica. O *chatbot web* comunica com o *backend* FastAPI, que orquestra a cadeia RAG com LangChain e Azure OpenAI. Os relatórios são exportados do Power BI e armazenados no Azure Blob; durante a ingestão, o *backend* extrai texto dos PDFs, segmenta em *chunks*, gera *embeddings* e persiste-os na base vetorial ChromaDB para pesquisa semântica. Na consulta, realiza-se a procura por similaridade na ChromaDB e o contexto recuperado fundamenta a resposta gerada pelo LLM, com prioridade a respostas ancoradas em evidência, conforme ilustrado na Fig. 2. Esta vista evidencia os contentores e as suas responsabilidades: interface (React+MSAL), lógica e APIs (FastAPI+LangChain), armazenamento de documentos (Azure Blob), indexação e recuperação (ChromaDB) e serviços externos de IA (Azure OpenAI).

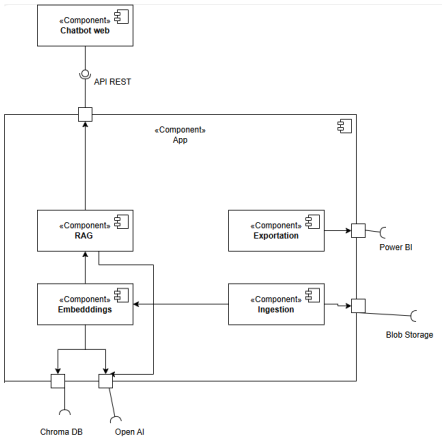


Fig. 2: C4 Nível 2 Diagrama de vista lógica.

3.6 FURPS requisitos não funcionais

O modelo FURPS orientou os requisitos não funcionais da solução, com foco em: usabilidade (UI responsiva em React com pesquisa em linguagem natural e *SSO* via MSAL); confiabilidade (autenticação segura com AAD/JWT, *logging* estruturado e rastreabilidade *Query*→*Chunks*→Resposta); desempenho (latência média $< 4\,\text{s}$ e recuperação vetorial $< 40\,\text{ms}$); e suportabilidade (código modular em FastAPI/LangChain com testes unitários e de integração).

3.7 Contributo face ao estado da arte

Face a este panorama, a abordagem proposta diferencia-se ao combinar, num único fluxo *end-to-end*, a exportação automatizada de relatórios Power BI via REST API, o armazenamento seguro em Azure Blob Storage, a indexação vetorial com ChromaDB e a geração de respostas com Azure OpenAI num cenário RAG, com autenticação corporativa e citação explícita dos trechos utilizados. Em contraste com soluções focadas em FAQs ou em integrações pouco detalhadas, o protótipo enfatiza *grounding* nas fontes e rastreabilidade *Query*→*Chunks*→Resposta, respondendo diretamente às lacunas identificadas na literatura em termos de atualização, precisão e governança.

4 Implementação da Solução

Esta secção descreve a construção prática do sistema, mostrando como as decisões de análise e desenho se materializaram em código e serviços *cloud*. São abordadas a *pipeline* RAG (ingestão, indexação semântica e consulta), os serviços de *backend* em FastAPI, a interface *frontend* em React com MSAL, a exportação via Power BI REST API com armazenamento em Azure Blob e os aspetos de segurança e operação.

4.1 Visão geral da *pipeline* RAG

A Fig. 3 apresenta uma visão de alto nível da *pipeline* RAG adotada. A solução materializa uma *pipeline* em duas fases bem definidas:

- **Ingestão e indexação**: PDFs são exportados do Power BI e armazenados no Azure Blob; o texto é extraído, segmentado em *chunks* e transformado em vetores densos (`text-embedding-ada-002` via Azure OpenAI); os vetores e metadados ficam persistidos na ChromaDB para recuperação posterior.
- **Consulta**: a pergunta do utilizador é convertida em *embedding*, usada para procurar *top-k* por similaridade na ChromaDB e os trechos recuperados fundamentam a resposta gerada pelo LLM, com prioridade a respostas ancoradas em evidência.

Esta implementação reflete as recomendações para reduzir alucinações e tornar as respostas auditáveis, permitindo ao utilizador inspecionar os trechos que suportam cada resposta.

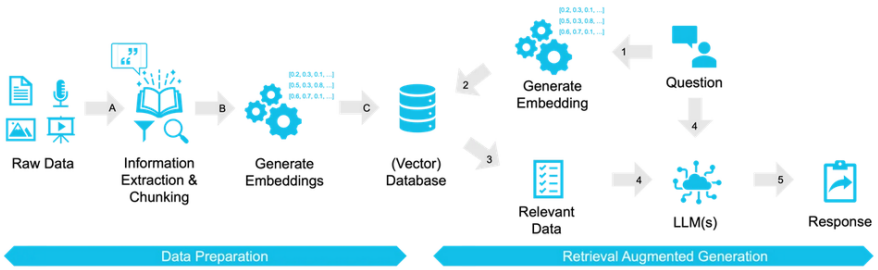


Fig. 3: *Pipeline* RAG utilizada no protótipo. [9]

4.2 Serviços e integração

O *backend* em FastAPI integra os serviços *cloud* e a lógica RAG, expondo *endpoints* para ingestão de PDFs (extração de texto, *chunking*, geração de *embeddings* e indexação em ChromaDB) e para consulta (receção da pergunta, recuperação semântica e chamada ao LLM). A utilização de FastAPI facilita a definição de rotas, a documentação automática da API e a criação de testes com `TestClient`, deixando preparada a evolução para *pipelines* automatizadas de *deploy* e CI/CD.

A interface *frontend* foi desenvolvida em React e comunica com o *backend* via Axios, disponibilizando uma interface *web* responsiva onde o utilizador pode colocar perguntas e visualizar respostas com as respetivas citações. A autenticação é gerida com MSAL/Azure AD, garantindo *Single Sign-On* corporativo e reutilização de identidades existentes, enquanto o *frontend* apresenta *feedback* visual sobre o estado das operações (carregamento, sucesso, erro) e organiza o histórico de interações de forma simples.

A exportação de relatórios Power BI para PDF é automática e transparente para o utilizador: o *backend* invoca a Power BI REST API, realiza *polling* até o ficheiro ficar pronto e guarda o PDF no Azure Blob Storage com metadados adequados, ficando este disponível para ingestão. Os documentos são armazenados com encriptação em repouso e em trânsito, e os *embeddings* são persistidos na ChromaDB com metadados que permitem ligar cada resposta aos trechos de origem, suportando consultas por proximidade vetorial e rastreabilidade *Query*→*Chunks*→Resposta.

Ao nível da segurança e operação, rotas sensíveis são protegidas com JWT emitidos após autenticação em Azure AD, e o sistema regista *logs* estruturados por pedido, incluindo informação sobre o utilizador autenticado, tempos de resposta, erros e identificadores de correlação. Estes registos suportam monitorização contínua, diagnóstico de problemas e auditoria em contexto empresarial, completando a implementação prática da arquitetura definida na Secção 3.

5 Avaliação e Testes

5.1 Metodologia

Esta secção apresenta a metodologia de avaliação adotada, estruturada em três camadas: testes unitários e de integração com `pytest`/`TestClient` no nú-

cleo RAG e serviços, testes de desempenho focados na segmentação de documentos, recuperação vetorial e latência de resposta, e um cenário *end-to-end* ingestão→consulta com dados reais. Em conjunto, estes testes visam comprovar a robustez técnica da solução e o cumprimento dos requisitos definidos.

5.2 Resultados

Alguns dos principais resultados obtidos através dos testes implementados foram: a *pipeline* RAG interpreta perguntas em linguagem natural, localiza conteúdo relevante e gera respostas coerentes fundamentadas em trechos reais dos documentos, reconhecendo corretamente quando não existe informação disponível. O tempo médio de resposta ficou abaixo de 4 s. Relativamente ao desempenho, a segmentação de documentos grandes (100 000 caracteres) ocorre em menos de 1 s e o *retrieval* semântico apresenta tempos < 40 ms (top-k). Relativamente à qualidade, as respostas mantêm *grounding* com citações claras e satisfação percebida através de testes internos. A cobertura de testes é $\sim$63% (núcleo RAG $> 80\%$), recomendando-se ampliar E2E/CI.

6 Conclusão

O projeto demonstrou a viabilidade de integrar relatórios Power BI numa cadeia RAG apoiada por serviços Azure, cobrindo exportação automática, armazenamento em Azure Blob Storage, indexação com *embeddings* e geração com Azure OpenAI, orquestrados por FastAPI/LangChain e expostos numa interface React com autenticação corporativa. Ensaios com dados reais indicam tempos médios de resposta inferiores a 4 s e processamento de cerca de 20 relatórios PDF, com redução substancial do esforço de procura de métricas e fundamentação explícita nas fontes. Neste sentido, o sistema materializa uma prova de conceito tecnicamente sólida de BI conversacional com Power BI e RAG, ainda em fase de validação organizacional.

6.1 Objetivos concretizados

Globalmente, os objetivos foram maioritariamente cumpridos: a exportação automática para Azure Blob foi implementada mas não totalmente validada por dependência de licença Power BI Pro; a *pipeline* RAG (ingestão, *chunking*, *embeddings*, recuperação) encontra-se funcional; o *bot* React com MSAL atinge latências médias de cerca de 3,7 s; e foram aplicados mecanismos de segurança com AAD, JWT e RBAC, ainda sem auditoria externa. Ensaios internos indicam reduções de cerca de 80 % no tempo de pesquisa de métricas.

6.2 Limitações

Apesar da cobertura funcional alcançada, subsistem limitações relevantes para uso em produção. A exportação via Power BI REST API está implementada e

integrada com Azure Blob Storage, mas não foi validada *end-to-end* por falta de licença Power BI Pro, requisito para geração de PDFs. O canal conversacional permanece limitado ao *frontend web* em React, sem publicação em Microsoft Teams/Outlook via Bot Framework. A *vector store* em ChromaDB com persistência local é adequada ao protótipo, mas não garante alta disponibilidade nem escalabilidade horizontal. A cobertura de testes ronda 63% (núcleo RAG $> 80\%$), ainda sem *pipeline end-to-end* nem CI/CD automatizada, e não existem métricas de utilização nem *feedback loop* dos utilizadores, o que dificulta a afinação progressiva do sistema.

6.3 Trabalho futuro e apreciação final

A solução cumpriu a finalidade proposta de permitir consultas em linguagem natural com respostas fundamentadas, sustentando ganhos práticos no acesso a informação de BI e preparando a evolução para contextos empresariais. Como trabalho futuro, recomenda-se: validar o fluxo de exportação com licenciamento adequado, publicar o agente em canais corporativos (Teams/Outlook), migrar a base vetorial para serviços geridos e integrar CI/CD com testes *end-to-end*, e realizar ensaios com utilizadores finais em ambientes realistas, incorporando *feedback loop* e métricas de utilização. Estes passos são essenciais para consolidar a transição do protótipo para operação e medir, de forma sistemática, o impacto da abordagem RAG na acessibilidade e precisão da análise em Power BI.

References

1. V. V. Gurbade, P. Verma, S. Gundewar, V. S. Bramhe: Building a Data Lake for Power BI in the Cloud: A Review on Utilizing Cloud Storage Services for Large Datasets. *Proc. 2024 2nd DMIHER Int. Conf. on Recent Trends in Engineering, Science & Management (RTESM)* (2024).
2. M. Arslan, S. Munawar, C. Cruz: Business insights using RAG–LLMs: a review and case study. *Journal of Decision Systems* (2024).
3. L. Benaddi, A. Souha, C. Ouaddi, A. Jakimi, B. Ouchao: Towards a unified metamodel for developing the conversational agents for smart tourism. *Procedia Computer Science* 236 (2024), 241–247.
4. S. M. A. Khan: Microsoft Azure Bot Service. *Technical article*, ResearchGate (2023).
5. H. Andersson: Retrieval-Augmented Generation with Azure OpenAI. Bachelor Thesis, Mälardalen University, Sweden (2024).
6. FastAPI Documentation (2024). `https://fastapi.tiangolo.com/`
7. Microsoft Learn Azure Blob Storage (2024). `https://learn.microsoft.com/azure/storage/`
8. LangChain Docs Chroma Integration (2024). `https://python.langchain.com/docs/integrations/vectorstores/chroma`
9. CrateDB: RAG Pipelines for Chatbots (2024). `https://cratedb.com/use-cases/chatbots/rag-pipelines`
10. J. Johnson, M. Douze, H. Jégou: Billion-Scale Similarity Search with GPUs (FAISS White Paper) (2019). `https://faiss.ai`

Session 6B

"Transformação Digital: Soluções para Gestão Empresarial, Biomédica e Académica"

Chair: Paulo Maio **Room:** H207 **Time:** 16:45 - 17:30

Paper	Presenter
Microsoft Power Platform - Management Solution for the Production Department	Beatriz Azevedo
Simulação de Eletrocardiograma com o modelo de McSharry	Sofia Nogueira
Design and Implementation of a Conference Management System for Use in Academic Institutions	Piedade Carvalho

Microsoft Power Platform - Management Solution for the Production Department

Beatriz Azevedo[1] and Emanuel Silva[1[0000-0002-5046-89424]]

[1,2] Instituto Superior de Engenharia do Porto
{1221672,ecs}@isep.ipp.pt

Abstract. In the current context of digital transition, companies face the constant challenge of modernising their internal processes, increasing their operational efficiency and responding quickly to market demands. Digital transformation has become an essential factor for the competitiveness and sustainability of organisations, requiring the adoption of new tools and methodologies.
It is against this backdrop that VoltEdge emerges, an organisation in the electric mobility sector seeking to evolve its working methods, promoting more effective management of its operations.
This paper details the entire process of developing a solution to the problem presented by the company. This process involves identifying and analysing the requirements, designing the solution, implementing it and testing it.
Finally, the results obtained are presented and a reflection is made on the work carried out, in which the limitations to its realisation are presented and possible improvements are suggested.

Keywords: Operations Management, Power Platform, Low-code.

1 Introduction

In the current era of digital transformation, organizations increasingly rely on innovative technological solutions to optimize processes, enhance productivity, and support data-driven decision-making. Within this context, Microsoft's Power Platform has emerged as a key ecosystem, enabling the rapid development of business applications, process automation, and data integration through low-code tools.

The project presented in this paper was developed under the Learning and Development (L&D) programme of Hitachi Solutions, a global consultancy specializing in digital transformation across industries such as manufacturing, retail, financial services, and healthcare. The initiative aimed to provide trainees and recent graduates with practical experience in applying both technical and business knowledge to a real-world-inspired scenario.

The proposed solution was based on a fictitious company, VoltEdge, operating in the electric mobility sector and focused on the design and implementation of a digital system to improve operational management within its production department. The project enabled the exploration of Power Platform technologies, particularly Power Apps,

Power Automate, and Dataverse, while reinforcing concepts of software engineering, system integration, and user experience design.

1.1 The Problem

VoltEdge is a fictional electric mobility company focused on charging solutions for electric vehicles. The company aims to transition from Excel-based processes to more integrated digital solutions. The main challenge was to develop an application to support critical activities—customer management, projects, and sales, warehousing, and production operations—automating processes, improving interdepartmental communication, enhancing efficiency, and supporting decision-making.

1.2 Objectives

The main objectives are:

- Creating and structuring Dataverse entities (Case Reports, Tests, Devices, Assembly Lines, Packages, etc.);
- Implementing automated logic using Power Automate Flows and C# plug-ins (e.g., automatic code generation, status updates, performance indicators);
- Developing customized interfaces for operational data recording and visualization;
- Creating Power Pages and a Canvas App for web and mobile access;
- Applying security via personalized profiles and column-level restrictions;
- Simulating maintenance, testing, and equipment replacement scenarios;
- Digitally representing the full production cycle with quality and traceability validations.

1.3 Approach

The project followed an agile-inspired methodology adapted for the educational context, allowing progressive learning through practical application. Weekly thematic modules covered specific competences, starting with theory presented by company professionals and followed by hands-on. Continuous documentation through reports facilitated reflection and assessment. The 17-week internship (February–June 2025) was structured to progress from fundamental to advanced topics. Key activities included introductory sessions, Dataverse learning and development, automation and plug-ins, security, web resources, Power Pages, Canvas App development, presentations, and final report preparation.

2 State of the art

This section reviews work related to the project, documenting current practices, similar systems, and the technologies used, justifying contributions to the field.

2.1 Related Works

ERP (Enterprise Resource Planning) in Rolls-Royce: The company integrated over 1,500 previously independent systems using an ERP solution to improve efficiency, reduce costs, and provide faster access to information, despite challenges such as cultural resistance and technical migration issues [1].

Mesfine Industrial Engineering (MIE): MIE replaced multiple outdated systems with Microsoft Dynamics SL, implementing ERP to unify processes across departments. Benefits included improved data flow, reduced production delays, and better deadline compliance, achieved through phased planning and specialized teams [2..4].

Alternatives to ERP:

- CRM (Customer Relationship Management): Focused on sales, marketing, and customer service, improving communication and loyalty [5];
- FSM (Field Service Management): Optimizes field operations such as maintenance and installations, managing work orders, teams, and inventory [6].

Comparison: ERP integrates all company areas, while CRM and FSM are specialized tools. Choice depends on integration needs, cost, implementation complexity, and organizational priorities.

2.2 Existing Technologies

Low-Code Platforms: These platforms simplify application development using visual interfaces, reducing programming complexity and enabling rapid deployment. This makes work faster and applications can be created in less time than with the traditional method, which also helps to cut costs [7]. They also support advanced features like artificial intelligence (AI), business intelligence (BI), and robotic process automation (RPA). The most relevant and well-known low-code platforms are the following:

- Microsoft Power Platform: Includes Power Apps (custom apps), Power Automate (workflow automation), Power Pages (websites), Canvas Apps (drag-and-drop app design), Dataverse (data storage), Dynamics 365 (integrated business management), and Copilot Studio (intelligent virtual assistants) [8].
- OutSystems: A Lisbon-based low-code platform for rapid development, using AI for optimisation and supporting diverse sectors[9][10].
- Mendix: Offers no-code (Mendix Studio) and low-code (Mendix Studio Pro) environments, integrating AI assistance and promoting collaboration between technical and non-technical teams [11].

Platform Comparison: According to Forrester Wave 2025, Microsoft leads in both capabilities and strategy, OutSystems follows closely with strong features, and Mendix balances functionality and collaboration, i.e., Microsoft Power Platform leads with its deep ecosystem and advanced AI/RPA features, while OutSystems focuses on high-performance, scalable enterprise solutions. Mendix stands out for its collaborative no-code/low-code environment suited to diverse development teams. The choice depends on project requirements and organizational context.

3 Analysis

This section details the analysis, requirements of a Microsoft Power Platform-based solution for a company providing sustainable electric vehicle charging solutions, focusing on the Production Department.

3.1 Problem Domain

The company relied heavily on Excel spreadsheets, causing communication issues between Production, Storage, and Sales Departments. The solution digitizes Production processes, responsible for manufacturing, maintaining, and delivering chargers and batteries. Main concepts include:

- **Assembly Line / Process / Supervisor** – manages production stages and staff.
- **Device / Item / Package** – tracks machinery, products, and packaging.
- **Case Report / Revision / Test / Testing Goal Indicator** – monitors faults, checks, and performance indicators.
- **Service / Replacement Requests** – handles maintenance needs.

The Production Department has two key roles:

- **Assembly Supervisor** – oversees production and maintenance.
- **Quality Standards Supervisor** – ensures compliance with quality standards and manages incidents.

Functional Requirements

The system's functional requirements define the essential capabilities to support production management, including entity creation, modification, automation, and real-time monitoring. These are organized into several Use Cases (UCs) that describe the main interactions between users and system components.

- **UC01 – Create Entities:** The Assembly Supervisor manages entities such as Assembly Line, Assembly Process, Device, and Package, while the Quality Standards Supervisor oversees Case Reports, Replacement Requests, Revisions, Tests, and Testing Goal Indicators.
- **UC02 – Edit Entities:** Enables update and maintenance of records across all entities.
- **UC03 – Complete Assembly Process:** Covers all production stages, from identification to storage, with automatic item status updates.
- **UC04 – Generate Low Productivity Alerts:** A Power Automate flow detects devices with productivity below 85% and sends email alerts to the supervisor.
- **UC05 – Trigger Replacement Request:** A custom button allows the Quality Standards Supervisor to create a replacement request and update device status automatically.
- **UC06 – Generate Unique Code:** A C# plugin generates unique alphanumeric identifiers for Case Reports, ensuring traceability.

- **UC07 – Scan QR Code:** The mobile Canvas App enables the Assembly Supervisor to access device information on-site via QR code scanning.

Non-Functional Requirements, Based on FURPS+ model

- **Functionality:** Access control and permission mechanisms were implemented through role-based profiles, ensuring data confidentiality and compliance. Evaluation considered correct role assignment and successful enforcement of user restrictions.
- **Usability:** The solution achieved multi-device compatibility (mobile, tablet, desktop). Usability was measured by interface responsiveness and error-free task execution across devices.
- **Reliability:** Stability was assessed through continuous operation and consistent response times under normal workloads.
- **Performance:** Automated workflows and plugins were optimized for execution time and system load.
- **Supportability:** All functionalities were fully tested and documented, ensuring maintainability and adaptability to future updates.
- **Other:** The system was entirely developed using Microsoft Power Platform and Dynamics 365, supporting scalability, integration, and low-code extensibility.

4 Solution Design

The design illustrates how the system components interact using a low-code approach. **Fig. 1** ilustrates the system architecture which includes the following components:

- Model-Driven App – main interface for supervisors;
- Canvas App – mobile access for field operations;
- Power Pages – external portal with Copilot chatbot;
- Power Automate – automates workflows like alerts and record creation;
- Dataverse – unified database for all data;
- Copilot Studio – chatbot support;
- External Service API – integration with Teams and Outlook.

Activity Flows for Key Use Cases:

- Register/Update Entity – users fill in forms in Power Apps; data validated and stored in Dataverse;
- Complete Assembly Process – five phases from Identification → Production → Final Item → Case Report → End Item, guided by Business Process Flow;
- Generate Productivity Alerts – automated flow checks device productivity and sends email alerts;
- Create Replacement Request – button triggers JavaScript flow to create request and update status;

- Generate Unique Code – plugin automatically generates unique alphanumeric code for Case Reports;
- Scan QR Code – mobile app validates QR code and retrieves device info from Dataverse.

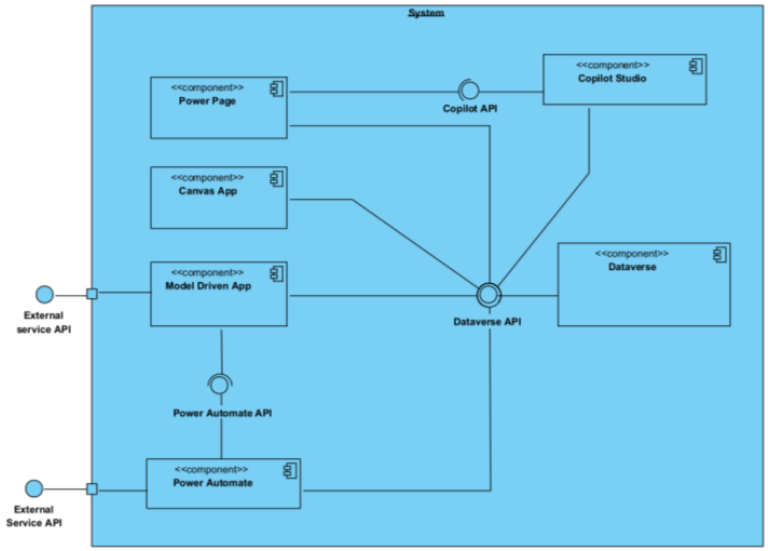


Fig. 1. – Component Diagram

5 Implementing the solution

The implementation phase followed a use case–driven approach, ensuring each functionality was developed, validated, and integrated within the Microsoft Power Platform. Using Dataverse, Power Apps, Power Automate, and C# plugins, the system ensured cohesive management of business logic and user interactions. UC01 and UC02 served as foundational models for subsequent implementations.

5.1 Implementation

The development process was organized around specific Use Cases (UCs), each corresponding to a key operational requirement within the production management workflow.

- **UC01-Create Entities**

The first stage focused on structuring the database in Microsoft Dataverse. Custom entities were created to represent key business objects—such as devices, industrial plants, and case reports—accessible through the Power Apps Main Page. Columns were defined with mandatory and optional attributes, while Choice and Lookup fields ensured data integrity and logical relationships. Custom forms were developed as user interfaces, allowing intuitive and controlled record creation and editing

- **UC02 – Edit Entities**

This stage enabled users to manage and modify existing records directly through the configured forms. Customized views were developed to display only the information relevant to each user's role, filtering records according to business unit and operational context. The system's security configuration relied on role-based access control, defining permissions at a granular level—Create, Read, Write, Delete, Append, Assign, and Share—thus ensuring both usability and compliance with access policies.

- **UC03 – Complete an Assembly Process**

The assembly process was modeled as a Business Process Flow with stages: Identification → Production → Final Item → Case Report → Closure. Workflows and business rules automated status updates, tracking items from In Use to Produced and In Testing, concluding with the assembly automatically set to Inactive.

- **UC04 – Generate Productivity Alerts**

Operational efficiency was further enhanced through a Power Automate flow designed to monitor device productivity. Executed every two days, the flow queried Dataverse to identify devices with performance levels below 85%. When such cases were detected, automated email notifications were sent to the Quality Standards Supervisor, enabling proactive intervention and maintenance scheduling.

- **UC05 – Create New Replacement Request**

To streamline maintenance operations, a custom button was integrated into the Device form using the XRM Toolbox and Ribbon Workbench.
When triggered, a JavaScript Web Resource initiated a Power Automate flow that automatically generated a new Service Request and updated the device's status to Awaits Replacement Request Approval. The system provided immediate feedback through an on-screen pop-up, confirming successful submission and improving the user experience.

- **UC06 – Generate Unique Code for Case Report**

Data uniqueness and traceability were ensured through a C# plugin registered via XrmToolBox. Executed in the pre-operation stage of Case Report creation, the plugin—implemented in the class CaseReportHelper—generated a unique alphanumeric identifier following the format [IndustrialPlant]-[ItemCode]-[RandomCode] or [ItemCode]-[RandomCode].
This process guaranteed the generation of non-redundant codes, combining structured and random elements (four to five alphanumeric characters) before record insertion into Dataverse.

- **UC07 – Visualize Device Information On-Site**

Finally, a mobile Canvas App was developed to facilitate on-site access to device information. Using QR code scanning, supervisors could instantly retrieve key data fields,

such as device code, operational status, and timestamp of the scan. This feature significantly improved real-time monitoring and decision-making efficiency, providing a fast and intuitive interface for field operations.

5.2 Power Pages Interactive Website

A customer-oriented website was developed, featuring key sections such as Home, FAQ, Privacy Policy, and Contact pages to provide users with clear and accessible information. The solution also includes a Production Department Dashboard, which enables supervisors to easily create, edit, and manage records directly through the web interface.

Accessibility was a core consideration during development, with the ARC Toolkit employed to ensure compliance with inclusive browsing standards. In terms of security, the platform incorporates robust authentication mechanisms and access control to protect user data and maintain system integrity.

Additionally, a Copilot-powered chatbot was integrated into the website to enhance user experience. The chatbot assists visitors by answering frequently asked questions, guiding them through the site, and offering personalized support based on available data.

5.3 Tests

Several tests were conducted to validate the correct functioning of the implemented requirements.

Power Automate Flows:

Both manual and automated tests were performed using Power Automate's built-in tools. Manual testing allowed real-time observation of the logic and data flow, while automated testing reused previous run data to validate fixes and improvements. During development, various flows—such as productivity alerts and automatic creation of Replacement Requests—were tested to confirm that they behaved correctly in different scenarios.

Plugins:

Plugin tests were implemented using the FakeXrmEasy framework, which simulates the Dataverse environment and allows isolated unit testing of plugin logic. Following the AAA pattern (Arrange, Act, Assert), simulated entities were created and initialized, the plugin execution context was configured, and the results were validated—ensuring, for example, that unique codes were generated correctly for case reports.

Acceptance Tests:

Acceptance testing was carried out on the model-driven application, the canvas app, and the company website. Supervisors and mentors participated through weekly meetings, reviewing implemented features, providing feedback, and suggesting improvements. This continuous validation ensured that the solution met the defined requirements and

5.4 Solution Evaluation

The evaluation process focused on verifying that all functional and non-functional requirements identified earlier were fulfilled. Tests on Power Automate flows, plugins, and applications confirmed that the implemented logic correctly addressed all use cases, enabling VoltEdge's production staff to perform their tasks efficiently.

The solution was continuously assessed in weekly review meetings, where progress was demonstrated and feedback was collected. The final presentation to company staff and management received highly positive feedback, emphasizing the system's robustness, usability, and alignment with business objectives.

In summary, all requirements were successfully achieved, and the overall evaluation of the solution was very positive, both technically and functionally.

6 Conclusions

All project objectives were achieved. Dataverse entities were structured with proper relationships, and automation was implemented via Power Automate flows and C# plugins for code generation, device status updates, and performance calculations. Customized interfaces, a Power Pages website, and a mobile Canvas App facilitated user interaction and field management. Security was ensured through role-based access and column-level restrictions. Operational scenarios validated system robustness, while the solution provided a full digital representation of the production cycle with quality assurance and traceability, delivering a comprehensive, business-aligned system for VoltEdge.

6.1 Limitations and future work

The main limitation encountered was the time required to acquire proficiency in Microsoft Power Platform tools. As a result, part of the internship period was dedicated to learning Dataverse, Power Automate, Power Apps, and Power Pages before implementation could advance.

Future developments could extend the Canvas App to additional operational areas, such as test management, revisions, and performance monitoring. Increased automation, particularly in reporting, quality validation, and proactive notifications would further reduce manual effort. Moreover, implementing analytical dashboards with real-time metrics could enhance monitoring and decision-making at the managerial level. These improvements highlight the system's scalability and flexibility, confirming its potential for continuous evolution within the organization.

6.2 Final assessment

The overall outcome of the project is highly positive. Despite initial challenges, all goals were accomplished, demonstrating consistent progress and practical learning. The

internship provided valuable experience in applying low-code technologies and developing integrated business solutions using Microsoft Power Platform and C#.

Beyond technical growth, the experience at Hitachi Solutions Portugal fostered collaboration, autonomy, and professional maturity, contributing significantly to both technical competence and soft skill development. This project therefore represents not only a successful technological implementation but also a meaningful step in academic and professional advancement.

References

1. Yusuf, Y., Gunasekaran, A., & Abthorpe, M. S. (2004). Enterprise information systems project implementation:: A case study of ERP in Rolls-Royce. *International Journal of Production Economics*, *87*(3), 251–266. https://doi.org/10.1016/J.IJPE.2003.10.004
2. Boltena, A. S., & Gomez, J. M. (2012). A Successful ERP Implementation in an Ethiopian Company: A case Study of ERP Implementation in Mesfine Industrial Engineering Pvt. Ltd. *Procedia Technology*, *5*, 40–49. https://doi.org/10.1016/J.PROTCY.2012.09.005
3. Chopra, R., Sawant, L., Kodi, D., & Terkar, R. (2022). Utilization of ERP systems in manufacturing industry for productivity improvement. *Materials Today: Proceedings*, *62*, 1238–1245. https://doi.org/10.1016/J.MATPR.2022.04.529
4. AlBar, A. M., Hddas, M. A., & Hoque, Md. R. (2014). Enterprise Resource Planning (ERP) Systems: Emergence, Importance and Challenges. *The International Technology Management Review 2014 4:4*, *4*(4), 170–175. https://doi.org/10.2991/ITMR.2014.4.4.1
5. Guerola-Navarro, V., Gil-Gomez, H., Oltra-Badenes, R., & Soto-Acosta, P. (2024). Customer relationship management and its impact on entrepreneurial marketing: a literature review. *International Entrepreneurship and Management Journal*, *20*(2), 507–547. https://doi.org/10.1007/S11365-022-00800-X/TABLES/6
6. Stumpp, N., Aschenbrenner, D., Stahl, M., & Aßmuth, A. (2024). *PLASMA -- Platform for Service Management in Digital Remote Maintenance Applications*. https://arxiv.org/pdf/2405.11836
7. Bock, A. C., & Frank, U. (2021). Low-Code Platform. Business & Information Systems Engineering, 63(6), 733-740.
8. Microsoft Corporation: New ways of development with copilots and Microsoft Power Platform. Microsoft Power Platform Blog (2024). https://www.microsoft.com/en-us/power-platform/blog/2024/05/21/new-ways-of-development-with-copilots-and-microsoft-power-platform/. (acesso em: 23 out 2025)
9. *Low Code Development Meets AI Innovation | OutSystems*. (n.d.). Retrieved June 17, 2025, from https://www.outsystems.com/
10. Martins, R., Caldeira, F., Sa, F., Abbasi, M., & Martins, P. (2020). An overview on how to develop a low-code application using OutSystems. *Proceedings of the International Conference on Smart Technologies in Computing, Electrical and Electronics, ICSTCEE 2020*, 395–401. https://doi.org/10.1109/ICSTCEE49637.2020.9277404
11. *AI Low-Code Application Development Platform | Mendix. (n.d.). Retrieved June 18, 2025, from https://www.mendix.com/platform/ai/*

Simulação de Eletrocardiograma com o modelo de *McSharry*

Camila Hayashida[1], Lícia Almeida[1], Sofia Nogueira[1,2], C.A. Ramos [1,2], G. Vilão [1,2]
[1]Departamento de Física, ISEP, Politécnico do Porto, Rua Dr. António Bernardino de Almeida 431, 4249-015 Porto, Portugal
[2]CIETI, ISEP, Politécnico do Porto, Rua Dr. António Bernardino de Almeida 431, 4249-015 Porto, Portugal
{1211843, 1211459, 1210684, CAR, GMR}
@isep.ipp.pt

Resumo. O presente estudo teve como objetivo desenvolver um sistema de simulação de sinais ECG baseados no modelo matemático de *McSharry*, recorrendo a dois métodos numérico distintos: *Euler Maruyama*, na sua formulação estocástica, e *Runge Kutta*, enquanto referência determinística. O estudo focou-se em avaliar a capacidade destes métodos para gerar sinais ECG realistas e fisiologicamente plausíveis, tanto em condições normais como em cenários patológicos, nomeadamente taquicardia, através da modulação apropriada dos parâmetros intrínsecos do modelo.

Para analisar a comparações entre os métodos foram implementadas métricas abrangentes no domínio temporal e espectral, incluindo erro quadrático médio, erro absoluto médio, coeficientes de correlação, análise de intervalos RR, amplitudes dos picos R. Os resultados demonstraram que ambos os métodos produzem sinais com morfologia consistente com registos reais, evidenciando uma reprodução fidedigna das ondas P, QRS e T. O método *Runge Kutta* forneceu uma trajetória determinística suave e estável, enquanto o método Euler *Maruyama* introduziu variabilidade ciclo a ciclo semelhante à observada em sinais biológicos, reforçando a relevância da modelação estocástica para simulação de fenómenos cardíacos. A análise evidenciou, no entanto, que alguns coeficientes de amplitude associados ao modelo necessitam de refinamento para assegurar plena coerência fisiológica ao nível das ondas PQRST. Apesar destas limitações, concluiu-se que a modelação matemática constitui uma abordagem promissora para a simulação de sinais ECG, com potencial para aplicações futuras em diagnóstico e monitorização cardíaca, apesar das limitações associadas aos métodos numéricos e ao próprio modelo.

Palavras-chave: Modelo matemático, *McSharry, Euler-Maruyama, Runge-Kutta,* Simulação de ECG

1 Introdução

A modelização de sistemas fisiológicos tem ganho relevância em contextos clínicos devido ao seu potencial para apoiar o diagnóstico, a monitorização e a personalização terapêutica. Um modelo matemático permite representar, de forma aproximada, o comportamento de um sistema real, embora sujeito a limitações inerentes à aquisição de sinais biológicos, frequentemente afetados por ruído e variações na amplitude [1].

Entre estes sinais, o eletrocardiograma (ECG) assume particular importância na avaliação da função cardíaca, sendo o método padrão para detetar anomalias no ritmo e na condução elétrica. A análise do complexo PQRST possibilita identificar arritmias e outras patologias cardiovasculares, responsáveis por aproximadamente 17,9 milhões de mortes anuais segundo a Organização Mundial da Saúde [2]. As principais componentes do ECG e respetivas características fisiológicas encontram se resumidas na Tabela 1.

Tab. 1 - Componentes do complexo PQRST de um individuo saudável

Componente	Descrição fisiológica	Amplitude	Duração
Onda P	Despolarização auricular	0,2 – 0,3 mV	< 0,12 s
Intervalo PR	Condução entre aurículas e ventrículos, incluindo despolarização auricular e atraso no nó AV	--	0,12 – 0,20 s
Complexo QRS	Despolarização ventricular	0,5 – 2,0 mV	0,06 – 0,10 s
Segmento ST	Período isoelétrico após a despolarização ventricular	--	0,08 – 0,12 s
Onda T	Repolarização ventricular	0,1 – 0,5 mV	0,10 – 0,25 s
Onda U	Repolarização tardia das fibras de Purkinje	< 0,1 mV	Variável

A atividade cardíaca resulta da despolarização e repolarização das células miocárdicas, gerando correntes elétricas que se propagam pelos tecidos e podem ser registadas à superfície por elétrodos, originando o traçado do ECG (Fig. 1) [1], [3]. Para além do diagnóstico, o ECG permite avaliar a variabilidade da frequência cardíaca através da análise dos intervalos R R, constituindo um importante indicador da modulação autonómica do coração [4].

O ECG é igualmente essencial na identificação de alterações como a taquicardia, caracterizada por frequência superior a 100 bpm, frequentemente associada a risco aumentado de eventos adversos [5]. Avanços em tecnologias digitais, sistemas IoT e inteligência artificial têm ampliado a capacidade de monitorização contínua do ECG, permitindo a análise em tempo real de grandes volumes de dados fisiológicos e promovendo deteção precoce de anomalias [2], [3], [5].

A escolha do modelo de McSharry justifica se pela sua capacidade de representar simultaneamente a morfologia completa do complexo PQRST e a variabilidade cardíaca, oferecendo maior fidelidade fisiológica do que modelos alternativos, como o de Clifford, que se centram sobretudo nos intervalos R R e em aproximações vetoriais do

dipolo cardíaco [6]. A integração numérica do modelo foi realizada pelos métodos Runge Kutta e Euler Maruyama: o primeiro fornece uma solução determinística precisa, enquanto o segundo introduz variabilidade estocástica, aproximando o comportamento irregular dos sinais reais [7].

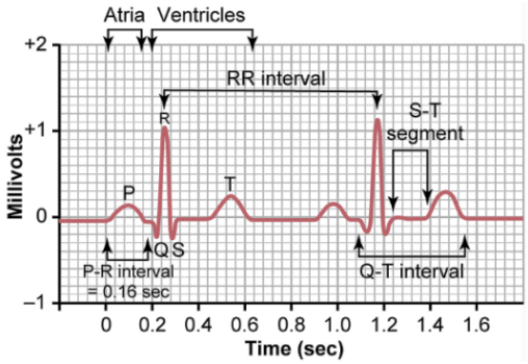


Fig. 1 – Representação típica da forma de onda e intervalos de um sinal ECG [3].

O objetivo deste estudo consistiu no desenvolvimento de um sistema de simulação de ECG baseado no modelo de *McSharry*, recorrendo aos métodos de *Euler Maruyama* e *Runge Kutta* para gerar sinais sintéticos capazes de reproduzir com fidelidade a atividade elétrica cardíaca em condições normais e patológicas.

2 Revisão Bibliográfica

A modelização de sinais fisiológicos, em específico o ECG, tem sido um campo de vasta investigação, procurando a reprodução precisa dos processos dinâmicos do sistema cardíaco. Nesse sentido, o modelo proposto por *McSharry et al.* representa uma contribuição importante, oferecendo a abordagem dinâmica pretendida para a geração de sinais sintéticos que preservam as características morfológicas e temporais observadas em registos reais de ECG. No entanto, este modelo apresenta algumas limitações que restringem a sua capacidade de manter efeitos a longo prazo e o *fine-tune* de comportamentos dinâmicos, o que implica dificuldades na simulação de sinais de ECG mais complexos que possuam ruído e artefactos, onde prevalecem as transições abruptas entre ritmos cardíacos [8]. *McSharry et al.* verificaram que o modelo apresenta resultados muito favoráveis para a representação de casos sem ruído, sendo a utilização de métodos e algoritmos simples suficiente. Por outro lado, para sinais de ECG com mais ruído e artefactos, concluíram que é necessário o uso de métodos mais complicados, de modo a representar e simular sinais mais realistas [7]. Não obstante, o modelo fornece uma base robusta para a simulação de sinais fisiológicos, possuindo a capacidade de reproduzir padrões e variabilidades inerentes ao comportamento cardíaco [7].

Na implementação do modelo de *McSharry*, recorreu se ao método de *Euler Maruyama*, uma adaptação estocástica do esquema de Euler tradicional amplamente utilizada na resolução de equações diferenciais estocásticas. Esta abordagem, conforme discutido por Ferreira e Lima [9], permite incorporar de forma eficaz a variabilidade e o

ruído inerentes aos sistemas biológicos, possibilitando a geração de sinais cardíacos que refletem de modo mais realista a natureza dinâmica e incerta da atividade elétrica do coração.

Outra abordagem relevante é a utilização do método de *Runge-Kutta* para a integração numérica do modelo. Este, conforme demonstrado por estudos na área, apresenta elevada precisão na resolução de equações diferenciais ordinárias, capturando com exatidão as rápidas variações e as nuances não-lineares do sinal de ECG. Para este método, já é empregue uma função em MATLAB/Octave com uma variação adaptativa baseada em *Runge-Kutta* de ordem 4(5), sendo uma solução prática e eficiente para a integração numérica do modelo [10], [11], [12].

3 Metodologia

O estudo foi desenvolvido em GNU Octave. Para a simulação dos sinais fisiológicos, foi implementado o modelo de *McSharry* descrito em [7] e revisto por dos Reis et al. [9]. Para validar a morfologia dos sinais gerados, procedeu se a uma comparação visual com registos reais de ECG, obtidos a partir de uma base de dados contendo traçados de indivíduos saudáveis e pacientes com alterações cardíacas, disponibilizada pelo *Ch. Pervaiz Elahi Institute of Cardiology Multan, Pakistan* [13]. A Figura 2 apresenta um diagrama de blocos que resume o procedimento metodológico seguido.

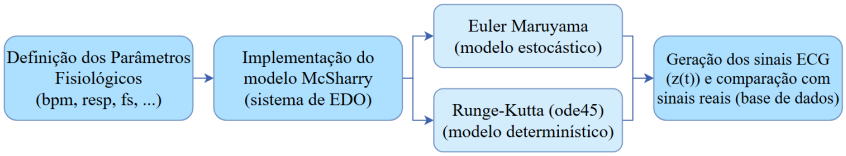


Fig. 2 - Diagrama de blocos do procedimento realizado.

3.1 Planificação do Desenvolvimento

A planificação do estudo envolveu a definição das variáveis de entrada e saída, da metodologia de modelação adotada e dos procedimentos de validação, assegurando uma implementação estruturada e consistente. Entre as variáveis de entrada comuns aos dois métodos numéricos incluem se a frequência cardíaca e respiratória, a amplitude da componente respiratória, os coeficientes de amplitude e largura angular das ondas PQRST, bem como os ângulos que determinam a posição destas no ciclo cardíaco. No método de *Euler Maruyama* são ainda introduzidos parâmetros adicionais (A1, A2, A3 e B), responsáveis pela incorporação de ruído estocástico e variabilidade fisiológica. Em ambos os métodos, a variável de saída principal é $z(t)$, correspondente à forma de onda do ECG.

A morfologia do sinal é obtida através de uma combinação de cinco gaussianas centradas em ângulos fixos, associadas às ondas P, Q, R, S e T, cujas amplitudes são ajustadas de acordo com a frequência cardíaca. Esta estrutura assegura que ambos os métodos gerem sinais compatíveis, permitindo uma comparação direta dos seus resultados.

3.2 Resolução numérica

O modelo foi resolvido recorrendo a duas abordagens distintas:

- *Euler Maruyama*, utilizando um passo temporal fixo (*dt*) e incluindo ruído gaussiano para modelar a variabilidade fisiológica. O método requer a definição explícita das condições iniciais das variáveis internas (*x, y, z e u*) e a parametrização dos coeficientes estocásticos A1, A2, A3 e do termo de acoplamento B. Esta abordagem é sensível à escolha do passo temporal e à magnitude do ruído, podendo alterar a morfologia do sinal.
- *Runge Kutta* (ode45), um método determinístico de passos adaptativos, empregado sem termos estocásticos. As tolerâncias numéricas (RelTol = 10^{-6} e AbsTol = 10^{-9}) garantem a precisão e, resultando num sinal mais regular e previsível, embora sem variabilidade fisiológica.

A Tabela 2 apresenta os parâmetros comuns utilizados nas simulações, tanto para condições fisiológicas como patológicas.

Tab. 2 - Parâmetros comuns utilizados no modelo de geração de ECG (saudável e patológico).

Parâmetros	Valores
Frequência Cardíaca (bpm) - Saudável; Patológico	77; 213
Tempo de Simulação (s)	10
Amplitude respiratória	$1{,}5\times10^{-4}$
Frequência Respiratória (Hz)	0,25
Ângulos centrais P, Q, R, S, T (°)	[−60,−15,0,15,90]
Amplitudes base P, Q, R, S, T	[24, -100, 750, -150, 15]
Larguras angulares	[0,25; 0,10; 0,10; 0,10; 0,40]
Condições iniciais	(1, 0, 0)

3.3 Implementação Computacional

A implementação consistiu na tradução computacional das equações diferenciais do modelo, originando dois scripts distintos correspondentes aos métodos de Euler Maruyama e Runge Kutta. O excerto de código na Cod. 1, exemplifica a atualização iterativa das variáveis no método estocástico, enquanto a Cod. 2 apresenta a integração do sistema via ode45.

```
for i = 2:N
  xi = randn; r = sqrt(x(i-1)^2 + y(i-1)^2); gamma = 1 - r;

  x(i) = x(i-1) + dt*(gamma*x(i-1) - w*y(i-1) + A1*xi + B*sin(u(i-1)));
  y(i) = y(i-1) + dt*(gamma*y(i-1) + w*x(i-1) + A2*xi);

  th  = atan2(y(i), x(i)); dth = mod(th - theta_i, 2*pi) - pi;
  sumGauss = sum( a_i .* dth .* exp( -dth.^2 ./ (2*(b_i.^2)) ) );

  z0   = A_resp * sin(2*pi*fr*t(i));
  z(i) = z(i-1) + dt*( z0 - z(i-1) - sumGauss + A3*xi );
  u(i) = u(i-1) + dt*w;
end
```

Cod. 1 – Excerto de código implementado para o método de *Euler-Maruyama*.

Em ambos os casos, o núcleo matemático permanece idêntico: um oscilador em coordenadas polares cuja dinâmica angular serve de base para gerar o sinal $z(t)$, representando o ECG. A base comum dos métodos assegura a produção de sinais com morfologia coerente, permitindo comparação direta.

```
[tR, YR] = ode45(@(t, y) system_eq(t, y, A_resp, fr, a_i, b_i, theta_i,
                        f_ecg), tspan, y0, opts);

function dy = system_eq(t, y, A_resp, fr, a_i, b_i, theta_i, f_ecg)
  x = y(1);yv = y(2);z = y(3);

  r2 = x^2 + yv^2; gamma = 1 - sqrt(r2); theta = atan2(yv, x);
  z0 = A_resp * sin(2 * pi * fr * t);

  sum_term = 0;
  for i = 1:length(a_i)
      delta_theta = mod((theta - theta_i(i)), 2*pi) - pi;
      sum_term = sum_term + a_i(i) * delta_theta * exp(-delta_theta^2 /
(2 * b_i(i)^2));
  end

  dx = gamma * x - 2 * pi * f_ecg * yv;
  dyv = gamma * yv + 2 * pi * f_ecg * x;
  dz = z0 - z - sum_term;
  dy = [dx; dyv; dz];
end
```

Cod. 2 –Implementação completa do método de Runge-Kutta de ordem 4(5), utilizando o *solver ode*45 para a integração numérica do modelo sintético de ECG proposto por McSharry.

4 Resultados e Discussão

Nesta secção apresentam se os resultados obtidos na implementação do modelo de *McSharry* e a análise comparativa entre os sinais de ECG simulados pelas abordagens determinística (*Runge Kutta*) e estocástica (*Euler Maruyama*). O modelo foi testado para diferentes valores de frequência cardíaca, abrangendo tanto condições fisiológicas normais (60 a 100 bpm) como um cenário patológico correspondente a taquicardia sinusal (>150 bpm). Em cada simulação avaliou-se a morfologia das ondas P, QRS e T, a regularidade do ritmo e a correspondência com traçados reais de ECG.

4.1 Validação do Modelo

A validação consistiu em analisar a capacidade do modelo de *McSharry* em gerar sinais fisiologicamente plausíveis e coerentes com as frequências cardíacas especificadas. Para tal, os métodos de *Runge Kutta* (ode45) e *Euler Maruyama* foram aplicados ao mesmo conjunto de parâmetros, permitindo verificar se diferentes esquemas de integração conduzem a resultados consistentes.

Análise do ECG Simulado em Indivíduos Saudáveis

Para 77 bpm, ambos os métodos produziram sinais praticamente sobreponíveis, com morfologia típica do ECG normal: onda P de baixa amplitude, complexo QRS estreito

e onda T bem definida. A sobreposição apresentada na Fig. 3 ilustra a equivalência ponto a ponto entre os dois métodos, demonstrando que a inclusão de ruído no *Euler Maruyama* não distorce a estrutura do complexo PQRST.

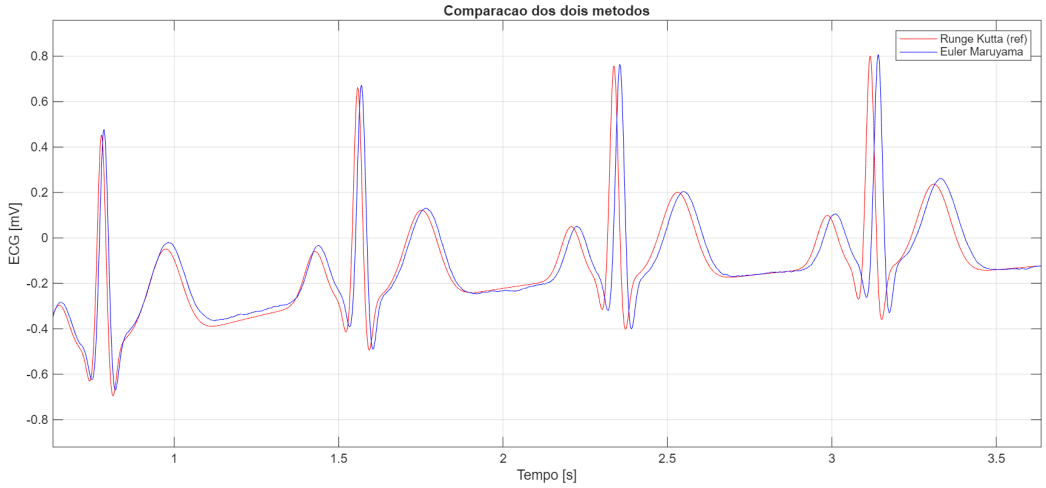


Fig. 3 – Sinal de ECG simulado com comparação dos métodos de *Euler-Maruyama* (–) e *Runge-Kutta* (–), sobrepostos na janela 0.5-3.5 s - representação de 4 ciclos.

Além da análise visual, realizou se uma comparação quantitativa entre os sinais produzidos pelos dois métodos:

- RMSE global (Erro Quadrático Médio)**:** valores reduzidos, evidenciando erro ponto a ponto mínimo.
- MAE (Erro absoluto médio): igualmente baixo, confirmando proximidade entre as formas de onda.
- Correlação: superior a 0.99, indicando elevada semelhança morfológica.
- Erro médio e variância do erro: próximos de zero, revelando ausência de viés sistemático.
- SNR (relação Sinal-Ruído): elevado, demonstrando que o ruído introduzido é pequeno face ao conteúdo fisiológico.
- Comparação RR: intervalos RR médios equivalentes, com ligeiramente maior variabilidade no método estocástico.
- Amplitude do pico R: valores próximos entre métodos, com pequenas variações atribuídas ao ruído.

A análise espectral revelou um espectro típico de sinais ECG reais, sem pico dominante na frequência cardíaca, mas com distribuição energética coerente nas bandas fisiologicamente relevantes.

Análise do ECG Simulado em Condição Patológica (Taquicardia)

Para 213 bpm, ambos os métodos mantiveram a consistência morfológica, reproduzindo a característica redução dos intervalos RR e o aumento do número de batimentos por unidade de tempo. A sobreposição dos traçados (Fig. 4) confirma que o comportamento dinâmico do modelo se mantém estável mesmo em frequências elevadas.

Os parâmetros calculados mostraram:

- Redução acentuada dos intervalos RR, evidenciando o ritmo taquicardíaco.
- Manutenção da estrutura PQRST, apesar da maior densidade de ciclos.
- Pequenas diferenças entre métodos, mais visíveis na variabilidade introduzida pelo *Euler Maruyama*.

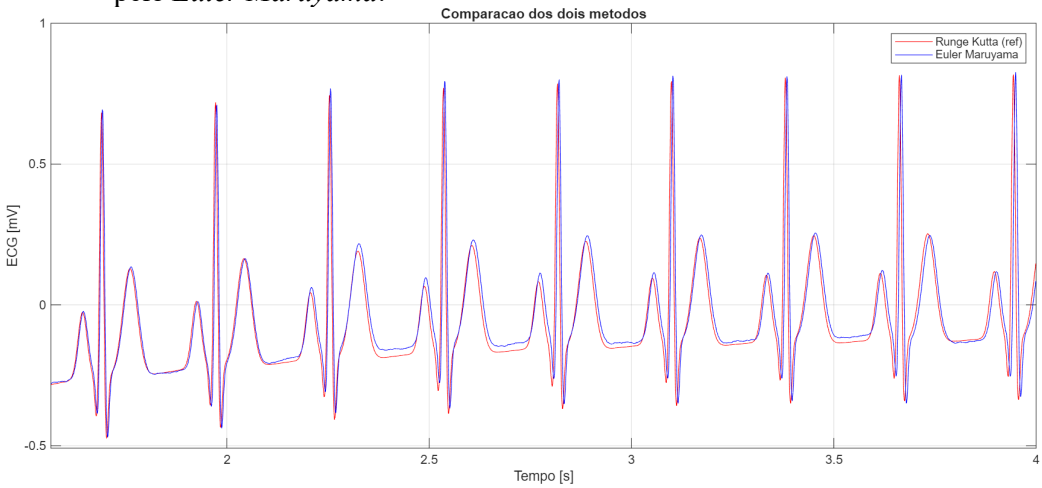


Fig. 4 – Sinal de ECG simulado com comparação dos métodos de *Euler-Maruyama* (–) e *Runge-Kutta* (–), sobrepostos na janela 1.5-4 s - representação de 9 ciclos.

Comparação dos Traçados: ECG Real *vs* ECG Simulado

Para verificar a fidelidade fisiológica das simulações, os sinais gerados foram comparados com traçados reais provenientes da base de dados [13]. A Tabela 3 apresenta os parâmetros morfológicos principais. Os valores simulados mostram correspondência elevada com os dados reais, tanto nos cenários saudável como patológico.

As diferenças observadas na amplitude das ondas, em particular nas ondas P e QRS, são atribuídas à natureza genérica dos parâmetros do modelo, que não é calibrado individualmente para cada paciente. Ainda assim, a morfologia P-QRS-T e a regularidade do ritmo foram reproduzidas com elevada fidelidade.

Tab. 3 - Dados morfológicos de sinal simulado e real para os casos saudável e patológico.

Característica	ECG simulado	ECG real [13]	ECG simulado	ECG real [13]
	Saudável		Taquicardia	
Ritmo (RR)	≈ 0,80	≈ 0,72	≈ 0,28	≈ 0,32
Pico R amplitude QRS	≈ 1,1 mV ≈ 1,4 mV	≈ 1,2 mV ≈ 1,8–1,9 mV	≈ 1,1 mV ≈ 1,4 mV	≈ 0,8 mV ≈ 1,3–1,6 mV
Morfologia P-QRS-T	Onda P presente; QRS estreito;	Onda P pequena; QRS estreito e alto;	Onda P presente; QRS estreito; morfologia P-QRS-T mantida	Onda P visível, mas pequena; QRS estreito; morfologia P-QRS-T mantida

Síntese da Comparação Entre Métodos

Os resultados obtidos permitem sintetizar as diferenças entre os dois métodos numéricos:

Runge Kutta (ode45)

- Produz sinais mais regulares e previsíveis.
- Erro numérico mínimo devido ao passo adaptativo.
- Ausência de ruído implica menor variabilidade fisiológica.

Euler Maruyama

- Introduz variabilidade estocástica semelhante à observada em ECG reais.
- Leve aumento no erro e na variância dos intervalos RR.
- Mantém elevada fidelidade morfológica apesar da discretização e do ruído.

Em média, o RMSE entre os dois métodos manteve se reduzido, com correlação muito elevada, demonstrando que ambos convergem para soluções praticamente idênticas do modelo de McSharry. A análise de múltiplas realizações do método estocástico confirmou que a média das trajetórias se aproxima da solução determinística, reforçando a consistência do modelo.

5 Conclusão

O estudo demonstrou que a modelação matemática, suportada pelo modelo de *McSharry*, constitui uma ferramenta eficaz para a simulação de sinais ECG com morfologia fisiologicamente plausível. A implementação dos métodos numéricos de *Runge Kutta* e *Euler Maruyama* permitiu gerar sinais consistentes tanto em condições normais como patológicas, reproduzindo adequadamente as ondas P, QRS e T. A comparação quantitativa entre os métodos revelou elevada concordância, refletida em valores reduzidos de RMSE, correlação superior a 0.99, erro médio próximo de zero e SNR elevado. O método de *Euler Maruyama* introduziu variabilidade estocástica semelhante à observada em sinais reais, enquanto o método de *Runge Kutta* produziu sinais mais regulares e determinísticos.

A validação com registos reais evidenciou boa correspondência da morfologia e dos intervalos RR, embora fossem necessários ajustes nas amplitudes das ondas para garantir maior coerência fisiológica. Persistem limitações relacionadas com a generalização do modelo e ausência de calibração individual, sugerindo a necessidade de validação clínica mais abrangente.

Em resume, o trabalho confirma o potencial da simulação matemática de ECG para apoio ao diagnóstico, desenvolvimento de sistemas de monitorização e teste de algoritmos de processamento, constituindo uma base promissora para investigações futuras em otimização de parâmetros e integração de métodos de inteligência artificial.

Referências

1. A. Monteiro, D. Guimarães, P. Carvalho, J. Alves, G. Vilão, "Interface para a análise de um sinal ECG." Simpósio de Engenharia Informática 2022, 2022

2. M. A. Serhani, H. T. El Kassabi, H. Ismail, and A. N. Navaz, "ECG monitoring systems: Review, architecture, processes, and key challenges," Mar. 02, 2020, MDPI AG. doi: 10.3390/s20061796.
3. A. José and M. Meireles, "ECG Denoising Based on Adaptive Signal Processing Technique," Nov. 2011.
4. P. Madona, R. I. Basti, and M. M. Zain, "PQRST wave detection on ECG signals," Gac Sanit, vol. 35, pp. S364–S369, Jan. 2021, doi: 10.1016/j.gaceta.2021.10.052.
5. J. Whitaker, M. J. Wright, and U. Tedrow, "Diagnosis and management of ventricular tachycardia," Clinical Medicine, Journal of the Royal College of Physicians of London, vol. 23, no. 5, pp. 442–448, Sep. 2023, doi: 10.7861/clinmed.2023-23.5.Cardio3.
6. Clifford, G. D., Nemati, S., & Sameni, R. (2010). An artificial vector model for generating abnormal electrocardiographic rhythms. Physiological measurement, 31(5), 595–609. https://doi.org/10.1088/0967-3334/31/5/001
7. P. E. McSharry, G. D. Clifford, L. Tarassenko, and L. A. Smith, "A dynamical model for generating synthetic electrocardiogram signals," IEEE Trans Biomed Eng, vol. 50, no. 3, pp. 289–294, Mar. 2003, doi: 10.1109/TBME.2003.808805.
8. A. Takha, M. L. Talbi, and P. Ravier, "Fractional calculus integration for improved ECG modeling: A McSharry model expansion," Med Eng Phys, vol. 132, p. 104237, 2024, doi: https://doi.org/10.1016/j.medengphy.2024.104237.
9. P. D. C. dos Reis et al., "Mathematical model for reproducing ECG signals from real clinical conditions," Revista Brasileira de Ensino de Física, vol. 46, 2024, doi: 10.1590/1806-9126-RBEF-20230367.
10. O. D. Nekui, W. Wang, C. Liu, Z. Wang, and B. Ding, "IoT-Based Heartbeat Rate-Monitoring Device Powered by Harvested Kinetic Energy," Sensors, vol. 24, no. 13, Jul. 2024, doi: 10.3390/s24134249.
11. M. Sinnoor and S. K. Janardhan, "Arrhythmia Identification and Classification using Runge Kutta Optimizer-Based Hyperparameter Optimization for Long Short Term Memory," Journal of The Institution of Engineers (India): Series B, Oct. 2024, doi: 10.1007/s40031-024-01038-7.
12. D. Yamuna, A. Ramaswamy, and G. Gulothungan, "Computational model based approach to analyze pacemaking activity of sinoatrial node due to calcium channels using Runge Kutta method," in AIP Conference Proceedings, American Institute of Physics Inc., Nov. 2023. doi: 10.1063/5.0174336.
13. A. H. Khan and M. Hussain, "ECG Images dataset of Cardiac Patients," Mar. 19, 2021.

Design and Implementation of a Conference Management System for Use in Academic Institutions

Soares, J.[1] and Carvalho, P.[1,2]
[1] ISEP-PPorto, Porto, Portugal
[2] GILT, ISEP-PPorto, Porto, Portugal

Abstract. This paper presents the development of a web-based conference management system designed to serve the research units of universities. The system supports the creation and management of multiple academic conferences, providing functionalities for paper submission, peer review, and event scheduling while ensuring role-based access control and customization.
The implementation used SvelteKit, TailwindCSS, and MongoDB, leveraging modern web frameworks to enhance user experience and scalability. Although not all features were completed, notably the customizable forms and automatic program generation, the project achieved the main functional objectives and provided valuable insights into contemporary full-stack web development.

Keywords: Conference Management Systems, Peer Review, Web Application, SvelteKit, Svelte, MongoDB

1 Introduction

The increasing scale and complexity of academic and scientific conferences require efficient digital tools for management and collaboration. As higher education institutions (such as universities) and their research centers organize multiple events annually, there is a pressing need for an internal, cost-effective system capable of handling submission, review, and scheduling processes.

Currently, research units within academic institutions, depend in general on third-party platforms such as EasyChair, which, while feature-rich, impose financial and functional limitations due to paid licensing tiers. This work proposes a proprietary, web-based conference management platform developed using modern JavaScript technologies (SvelteKit, TailwindCSS, and MongoDB) to replace such solutions and provide extensibility for future use within other organizational units.

The project's objectives were to: implement secure, authenticated access with permission levels; allow conference creation, paper submission, and review workflows; enable discussion between authors and reviewers; and support email templates and submission customization.

The research units previously relied on EasyChair for managing conference workflows. However, recurring costs and limited customization motivated the development of an in-house alternative.

The project was carried out within an internship in the third year of the BSc in Informatics Engineering. It provided a learning opportunity in modern full-stack web development, emphasizing usability, security, and maintainability.

The system was designed for use in organizational units of academic institutions, such as research and development groups, departments, and laboratories. The system should be reusable across multiple conferences, allow for the configuration of roles, permissions, and forms, and ensure scalability within the internal infrastructure of academic institutions.

2 State of the Art

The study of conference management systems reveals a mature yet fragmented ecosystem of software solutions developed over the past two decades. These systems typically address one or more of the following domains:

(1) submission and peer review management;
(2) event scheduling and program generation;
(3) participant communication and registration.

However, most tools either specialize in a single domain or present usability and cost limitations that restrict their broader adoption within academic institutions. This section reviews the main existing solutions and the relevant technologies used in modern web development that informed the design of the proposed system.

2.1 Existing Conference Management Systems

EasyChair is the most widely adopted online conference management platform in academia, supporting functionalities such as paper submission, blind and double-blind review, reviewer assignment, and decision management [2]. The system also offers event scheduling, statistics, and email communication. However, EasyChair's major drawbacks are its proprietary nature, high learning curve, and licensing costs for premium features such as advanced analytics and extended storage. Its interface, while functional, remains dated and unintuitive for first-time users. Additionally, conferences must be created by EasyChair administrators, limiting the autonomy of smaller organizations.

OpenConf represents one of the earliest open-source alternatives, providing extensive configurability and allowing full self-hosting [8]. It supports abstract and paper submission, customizable review forms, and schedule generation. Its *Community Edition* is free but requires manual installation and server maintenance, while *Professional Editions* introduce additional features and technical support. Despite its flexibility, OpenConf's user interface feels outdated and lacks modern UX standards, which limits adoption among non-technical organizers.

Oxford Abstracts focuses primarily on abstract submission and review workflows, simplifying the early stages of conference organization [9]. It features a highly intuitive interface, powerful filters, and reporting tools. However, the system does not support full paper submission or multi-track event scheduling, which reduces its suitability for large-scale academic events. Its reliance on paid licensing tiers also introduces similar cost barriers to EasyChair.

As shown in Table 1, a comparative analysis of these systems highlights their main features and limitations.

Table 1. Applications Feature Comparison

System / Feature	EasyChair	OpenConf	Oxford Abstracts	Proposed System
Paper submission	Yes	Yes	Abstract only	Yes
Peer review management	Yes	Yes	Yes	Yes
Schedule generation	Yes	Yes	Yes	Planned
Customization (forms, roles)	Limited	High (manual)	Moderate	High
Hosting	Proprietary	Self-hosted	Proprietary	Self-hosted
Cost	Paid tiers	Free / Paid	Paid	Free (internal)
Ease of use	Medium–Low	Low	High	High (target)

From the analysis, it becomes evident that no single platform provides an open, modern, and cost-free solution that balances usability, configurability, and extensibility for institutional needs. This gap motivated the creation of a customizable, open internal platform leveraging modern frameworks to combine the flexibility of OpenConf with the usability of Oxford Abstracts.

2.2 Web Technologies in Modern Conference Systems

Beyond functional comparison, the technological foundations of existing tools vary significantly. Traditional platforms were typically implemented using PHP and relational databases such as MySQL. Although reliable, these architectures lack the scalability and responsiveness demanded by today's web applications.

Modern systems, by contrast, are increasingly adopting JavaScript-based full-stack frameworks, leveraging the benefits of reactive interfaces, real-time updates, and serverless deployment [10, 12]. Three key technologies stand out in this context.

The evolution of web frameworks has been marked by the dominance of React, Angular, and Vue [10]. However, Svelte introduces a paradigm shift: instead of running the framework in the browser, Svelte compiles components at build time, generating minimal JavaScript bundles and delivering exceptional runtime performance [5].

SvelteKit, its companion meta-framework, integrates server-side rendering (SSR), routing, and API endpoints in a single unified environment [6]. This drastically simplifies full-stack development and avoids the complexity of maintaining separate frontend and backend applications.

For this project, SvelteKit's tight coupling between UI, logic, and routing proved ideal for prototyping, even though it imposed architectural constraints that limited horizontal scalability.

The choice of Prisma ORM with MongoDB Atlas provided a flexible and scalable backend. Prisma abstracts database interactions using a strongly typed schema and simplifies migration management.

Unlike SQL-based systems, MongoDB's document model enables flexible data storage - particularly suitable for heterogeneous entities such as conferences, papers, and reviews. Studies such as Chauhan [1] highlight MongoDB's efficiency in handling semi-structured research data, a core requirement for this system.

Tailwind CSS and DaisyUI were selected for the front-end styling, instead of traditional frameworks such as Bootstrap [13], to allow a utility-first approach, allowing developers to construct complex responsive layouts without maintaining large CSS files [3]. DaisyUI builds upon Tailwind to provide accessible, themeable UI components, accelerating development and ensuring consistent visual identity.

2.3 Architectural Trends

Modern conference management systems are increasingly adopting modular and layered architectures, such as the Onion or Hexagonal patterns [7, 11]. These models emphasize separation of concerns, testability, and independence from frameworks.

While SvelteKit's integrated architecture limits strict adherence to these patterns, this project applies the same principles conceptually: data flow moves from domain entities to application services, then to adapters (UI and persistence). This conceptual alignment ensures that, despite technical limitations, the system remains maintainable and logically organized.

From the literature and technological review, the following gaps were identified:

- Cost and licensing – Existing systems often require subscriptions for essential features.
- Limited customization – Most platforms offer fixed workflows not adaptable to specific conference types.
- Usability vs. flexibility trade-off – Tools with richer features (e.g., EasyChair) tend to be harder to use.
- Lack of open modern alternatives – Few systems exploit current-generation web frameworks such as SvelteKit or offer seamless integration of frontend and backend logic.

This context justified the design of a new, open, modular system that combines usability, configurability, and institutional autonomy, developed as a case study for academic institutions's research environment.

3 Analysis & Design

This chapter presents the main functional and non-functional requirements of the system.

Functional requirements included conference creation, role management, paper and review submission, email notifications, and discussion threads. Non-functional requirements emphasized usability, system security, performance — with page load times maintained under 2 seconds — and operational reliability.

The system follows a layered architecture inspired by the Onion Model, using SvelteKit to integrate front-end and back-end logic within a unified framework. From

a technological perspective, the frontend was built with SvelteKit, supported by TailwindCSS and DaisyUI for component styling. The backend logic was implemented inside SvelteKit using REST-like endpoints, with Prisma ORM managing persistence on a MongoDB Atlas cloud database. Authentication was ensured through Lucia-Auth, using hashed credentials and role-based permissions. Email notifications were implemented using NodeMailer, supported by dynamic template variables.

The main system actors are the Conference Manager, Author, Reviewer, and Administrator. The core use cases include conference creation and configuration, paper submission and peer review, discussion between authors and reviewers, and email invitation and template customization. Fig. 1 depicts the overall use case diagram.

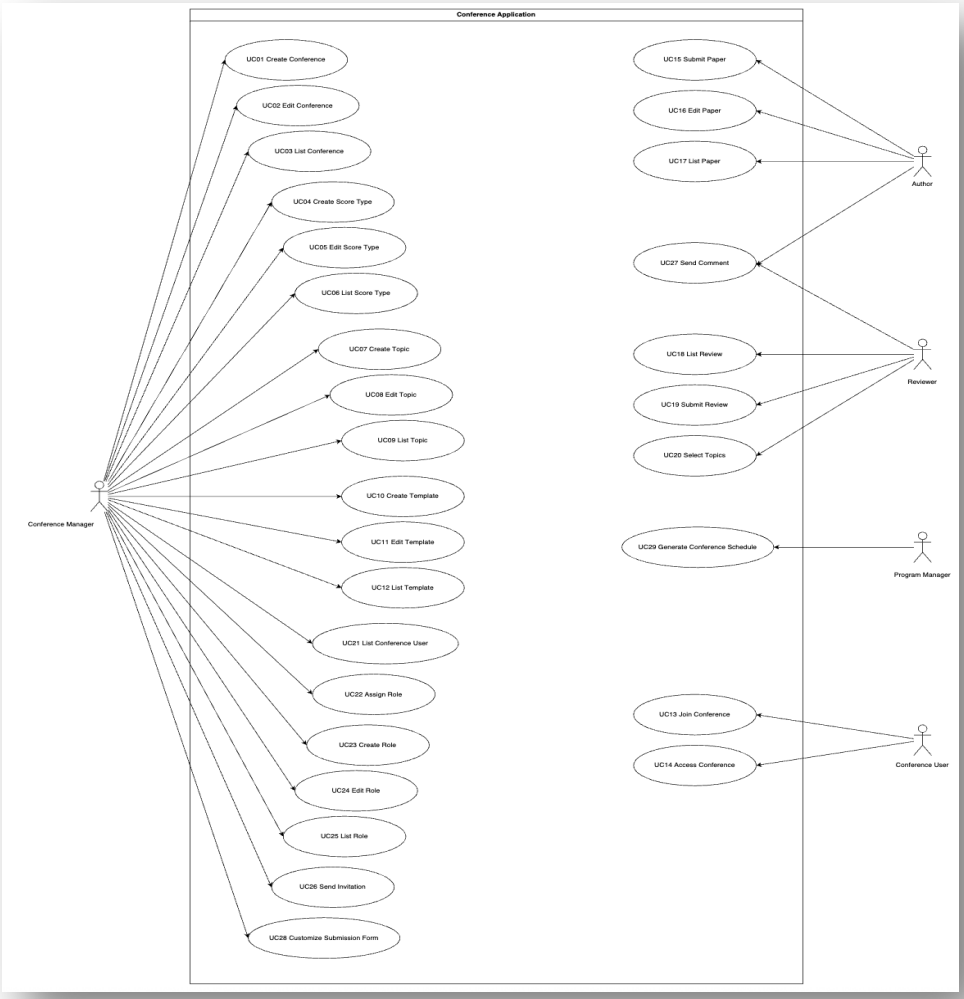


Fig. 1. Use Case Overview Diagram illustrating system actors and interactions.

The developed system is a web-based platform designed to manage the full lifecycle of academic conferences - from creation and submission to peer review and communication among participants.

It was built with a role-based access control model, ensuring that each user interacts only with the functionalities relevant to their role.

Access to the platform is handled through a secure login page, implemented using *Lucia-Auth* for credential verification.

Once authenticated, users are redirected to their personalized dashboard, which adapts according to their assigned role.

This approach enforces both security and functional separation between user types.

The Administrator oversees the entire system and manages global settings. Key functionalities include:

- Creating and managing user accounts and permissions;
- Monitoring all active conferences;
- Managing institutional parameters such as storage limits and access policies.

The administrator dashboard presents a dynamic table with filtering and pagination to simplify multi-conference management.

The Conference Manager is responsible for creating and managing a specific conference.

Their main features include:

- Conference creation (name, deadlines, topics, and configuration) (Fig. 2);
- Participant management (authors and reviewers);
- Reviewer assignment for submitted papers;
- Progress monitoring of the review process;
- Email communication through customizable templates (Fig. 3);
- Form customization for submissions.

The interface is divided into tabbed sections (General, Submissions, Reviews, Communications), allowing quick navigation across management areas.

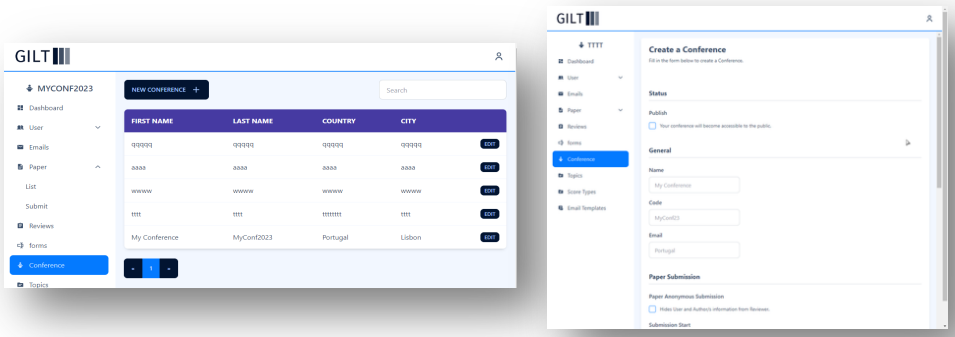


Fig. 2. Conference creation interfaces

Authors use the system to submit abstracts or full papers.

The submission process includes real-time validation using *Zod* and *Superforms*, ensuring correct and complete input.

Each submission is assigned a unique identifier and can be tracked through status indicators such as *Submitted*, *Under Review*, *Accepted*, or *Rejected*.

Authors can also:

- Edit or replace their submission before the deadline;

- View reviewer comments and feedback;
- Engage in direct discussion threads with reviewers.

The Reviewer interface provides access to assigned papers. Each paper entry includes:

- The uploaded file;
- A structured review form with scoring criteria (e.g., relevance, originality, clarity, recommendation);
- Comment fields for both the author and the conference manager.

After completing a review, the submission's status is automatically updated, and managers can view aggregated review data.

The system integrates an automated email notification module implemented with *NodeMailer*. Emails are triggered by key events such as:

- Submission confirmation;
- Reviewer assignment;
- Review completion and decision updates.

Conference managers can customize templates with dynamic placeholders (e.g., {author_name}, {paper_title}), allowing personalized automated communication.

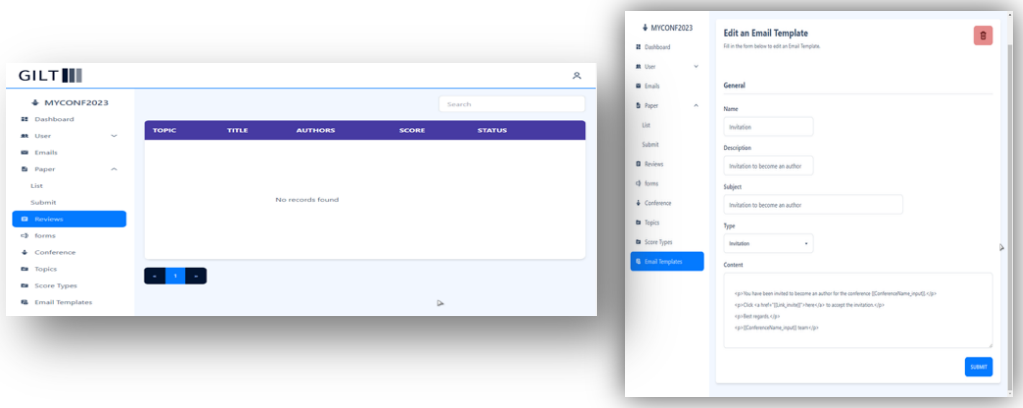


Fig. 3. Reviewer dashboard and email template interfaces

Thus, the application covers all essential stages of a scientific conference (creation, submission, review, and decision) within a unified, modern, and scalable platform. It provides academic institutions with a cost-free, customizable solution that balances usability, flexibility, and institutional autonomy.

4 Implementation

The project was developed using Visual Studio Code and Bitbucket for version control. The JavaScript runtime environment was Node.js [4], with pnpm managing dependencies. Testing was conducted via Playwright and MailCrab(for simulated email testing).

Originally, the system was designed as two separate applications (frontend and backend). Due to complexity in secure API integration, it was refactored into a single

SvelteKit monolithic application, which enabled faster development and easier integration at the cost of horizontal scalability.

This architectural choice is appropriate for prototyping and small-scale institutional use; however, in large-scale or production environments, it introduces limitations in terms of independent scaling and service decoupling. Alternative approaches could include a microservices-based architecture or a dedicated REST API backend separated from the frontend layer, as suggested in the future work section.

The frontend skeleton included: route-based navigation structure; layout inheritance (root and child layouts); TailwindCSS + DaisyUI integration; dynamic table components for pagination, filtering, and sorting.

Using Prisma ORM, models for users, conferences, papers, reviews, and templates were defined. Each entity had CRUD (Create, Read, Update, Delete) endpoints defined in +server.ts files.

Lucia-auth handled authentication, while roles were defined using binary strings for efficient permission storage.

The system implements several key functionalities to support conference management. Secure authentication and role-based access control have been fully implemented, ensuring that users can only access features appropriate to their assigned roles. Conference creation and editing is complete, allowing managers to define conference details, deadlines, and participant roles. The paper submission and editing module is fully operational, enabling authors to submit and modify their work before the deadlines. Similarly, the review submission and scoring functionality has been implemented, providing reviewers with structured forms to evaluate submissions and assign scores. An author–reviewer discussion feature is also available, facilitating direct communication and feedback exchange. In addition, the system supports email templates and invitations, allowing automated, customizable notifications to participants.

However, certain features remain unimplemented, including submission form customization, which would allow conference managers to fully tailor submission fields, and automatic schedule generation, which would automatically create event schedules based on submissions and reviews. These gaps represent potential areas for future development and enhancement.

The interface was built with SvelteKit, TailwindCSS, and DaisyUI, ensuring a responsive, consistent, and accessible design. The layout features a sidebar navigation menu and a main content area that adapts to different screen sizes. Reusable components (tables, forms, and modals) promote visual consistency and simplify maintenance.

5 Evaluation

Reusable form validation using Zod and Superforms streamlined user input validation. Server-side pagination was implemented manually, as no suitable Svelte-native library supported it.

Unit tests verified isolated functions, such as form validation and database queries. Although integration testing was challenging due to SvelteKit's tightly-coupled

architecture, End-to-End (E2E) testing using Playwright successfully validated core workflows including login, conference creation, and paper submission.

Due to the absence of an external test group, usability validation was performed internally. As a consequence, the results reflect a controlled and informal evaluation environment. Future iterations should involve real users — authors, reviewers, and conference managers — to enhance empirical validity.

Regarding performance, testing confirmed that page load times remained consistently below 2 seconds, and that database communication remained stable under normal operation conditions. However, no stress or large-scale concurrency testing was conducted, which restricts conclusions about scalability.

The project successfully demonstrated the feasibility of building a full-featured conference management system using SvelteKit. While the integrated architecture simplified deployment and accelerated development, it also introduced constraints in terms of modular scalability and distribution.

Lessons learned include:

- The importance of early architectural planning for scalability;
- The trade-off between framework simplicity and flexibility;
- The critical role of UI/UX considerations in user-centric systems.

The use of modern technologies - SvelteKit, TailwindCSS, Prisma, and MongoDBpr - oved efficient and educational, promoting deeper understanding of reactive programming and full-stack development.

6 Conclusion

The work presented in this paper achieved most of its original goals, producing a functional prototype of a Conference Management System tailored for academic institutions. It provides a viable internal alternative to EasyChair, offering cost savings and control over data.

In relation to the initial objectives, the system successfully implemented secure authentication, role-based access, conference creation, paper submission, review workflows, author–reviewer discussion, and email templates. However, two structurally significant modules — submission form customization and automated conference program generation — remain unimplemented. These features are considered essential for achieving full competitive parity with established platforms such as EasyChair and OpenConf and therefore represent high-priority areas for future development.

Future work should focus on: separating frontend and backend for horizontal scalability; implementing advanced authorization (JWT); completing missing modules and adding analytics dashboards; Enhancing accessibility and usability with user testing.

Despite limitations, the project delivered a meaningful, functional result and served as a strong foundation for continued development and institutional use.

References

1. A. Chauhan, "A review on various aspects of MongoDB," *International Journal of Engineering Research & Technology (IJERT)*, 2019.
2. EasyChair, "About EasyChair." Accessed: Dec. 3, 2025. [Online]. Available: https://easychair.org/
3. A. Fitzgerald, "Tailwind CSS: What it is, why use it," *HubSpot*, 2022.
4. F. G. Ghansham Jadhav, "Role of Node.js in modern web application development," *International Research Journal of Engineering and Technology (IRJET)*, 2020.
5. C. Humble, "All about Svelte, the much-loved, state-driven web framework," *The New Stack*, 2021.
6. L. Lawson, "Rich Harris talks SvelteKit and what's next for Svelte," *The New Stack*, 2023.
7. R. Nunkesser, "Using Hexagonal Architecture for Mobile Applications," pp. 113–120, 2022, doi: 10.5220/0011075100003266.
8. OpenConf, "OpenConf conference management system." Accessed: Dec. 3, 2025. [Online]. Available: https://www.openconf.com/
9. Oxford Abstracts, "Abstract management and peer review software." Accessed: Dec. 3, 2025. [Online]. Available: https://www.oxfordabstracts.com/
10. E. Saks, "JavaScript frameworks: Angular vs React vs Vue," *Medium*, 2019.
11. J. Sidler, E. Braun, C. Schmitt, T. Schlachter, and V. Hagenmeyer, "Microservice-based Architecture for the Integration of Data Backends and Dashboard Applications," *Karlsruhe Institute of Technology (KIT)*, 2024. [Online]. Available: https://publikationen.bibliothek.kit.edu/1000141315/136598801
12. S. Wildermuth, "Introducing Vite: A better Vue CLI?," *CODE Magazine*, 2022.
13. P. P. Suraj Shahu Gaikwad, "A review paper on Bootstrap framework," *IRE Journals*, 2019.

Authors Index