



**Politécnico
de Viseu**

Escola Superior
de Tecnologia
e Gestão de Lamego

Frontend de plataforma de gestão e monitorização de rotas de motas elétricas em contexto de serviços urbanos

Ana Paula Pombinho Vigário

Janeiro, 2026



**Politécnico
de Viseu**

Escola Superior
de Tecnologia
e Gestão de Lamego

Frontend de plataforma de gestão e monitorização de rotas de motas elétricas em contexto de serviços urbanos

Ana Paula Pombinho Vigário

Trabalho de Projeto

Mestrado em Tecnologias de Informação e Automação

Trabalho efetuado sob a orientação de

Professor Doutor Tiago Manuel Campelos Ferreira Pinto

Professor Doutor Fernando Miguel Soares Mamede dos Santos

Janeiro, 2026

***Frontend* de plataforma de gestão e monitorização de rotas de motas elétricas em contexto de serviços urbanos**

Ana Paula Pombinho Vigário

Dissertação

Mestrado em Tecnologias de Informação e Automação

Trabalho efetuado sob a orientação de
Professor Doutor Tiago Manuel Campelos Ferreira Pinto
Professor Doutor Fernando Miguel Soares Mamede dos Santos

Lamego, 2026

“Deus manifesta-se num sopro...”
-Autor desconhecido

Este trabalho foi escrito ao abrigo do novo Acordo Ortográfico.

AGRADECIMENTOS

A realização desta dissertação de mestrado não teria sido possível sem o apoio de diversas pessoas, às quais expresso o meu mais sincero reconhecimento.

À minha família, ao meu marido, aos meus filhos, minha irmã e ao meu pai, por todo apoio que me deram na realização desta conquista. Por todo apoio incondicional, paciência e motivação ao longo de todo este percurso académico.

Ao meu orientador, Professor Doutor Tiago Manuel Campelos Ferreira Pinto, pela aposta, pela disponibilidade, paciência e orientação, com seus conselhos e ideias, tal como, a partilha de conhecimento.

À toda equipa A-Mover, por permitirem a minha participação neste projeto, principalmente à “minha” equipa de Logística, por sempre partilharem desafios, conquistas, ensinamentos e suporte ao decorrer deste percurso académico.

Em suma, agradeço a todas as partes envolvidas que tornaram possível a realização deste projeto. O meu muito obrigada, a todos que, de uma forma ou de outra contribuíram para que esta dissertação se tornasse realidade.

RESUMO

O crescimento acelerado da população urbana tem intensificado os desafios associados à mobilidade sustentável, à eficiência logística e à segurança rodoviária. Neste contexto, os motociclos elétricos surgem como uma solução promissora, conjugando baixo impacto ambiental com elevada agilidade operacional.

No entanto, a utilização destes veículos é condicionada pela ausência de sistemas de navegação e apoio à condução verdadeiramente adaptados às características destes veículos e dos condutores. A presente dissertação propõe a implementação de um *frontend* de serviços para apoio à utilização profissional de motociclos. O *frontend* desenvolvido permite aos condutores de motociclos, em utilização profissional, autenticar-se no sistema, consultar e gerir serviços de entrega e recolha associados a uma rota diária, bem como visualizar essa rota otimizada num mapa interativo e acompanhar a navegação em tempo real. A *interface* adapta-se dinamicamente às atualizações, respondendo de maneira eficiente a mudanças imprevistas, como desvios de rota, de forma clara, rápida e intuitiva, contribuindo para a redução dos riscos associados à condução, de acordo com as necessidades e especificações do projeto A-Mover (Agenda Mobilizadora para o Desenvolvimento de Produtos e Sistemas Inteligentes de Mobilidade Verde), que é orientado para o desenvolvimento de soluções tecnológicas inovadoras no domínio da mobilidade urbana sustentável.

A solução proposta encontra-se integrada a um *backend* robusto, responsável por alimentar a *interface* com base de dados completa e atualizada. A metodologia adotada baseia-se numa abordagem interativa e centrada no utilizador, concebida segundo princípios de Interface do utilizador, Experiência do utilizador e Design Centrado no Utilizador, e implementada para plataforma Android, com ligação aos dispositivos de *display* na mota elétrica. Para validar a solução desenvolvida, foram realizados testes funcionais, de desempenho, de integração e unitários em ambiente controlado, incidindo sobre as principais funcionalidades da aplicação, como a autenticação, gestão tarefas, apresentação de rotas e a atualização do estado das entregas. Os resultados obtidos demonstraram a robustez e a viabilidade da solução proposta, confirmando o seu correto funcionamento.

Palavras-chave: A-Mover; Aplicação móvel, *Design* Centrado no utilizador; Experiência do utilizador, Interface do utilizador, Mobilidade urbana sustentável; Motociclos elétricos; Serviço de logística; Sistema de navegação inteligente.

ABSTRACT

The accelerated growth of the urban population has intensified the challenges associated with sustainable mobility, logistical efficiency, and road safety. In this context, electric motorcycles emerge as a promising solution, combining low environmental impact with high operational agility.

However, the use of these vehicles is limited by the lack of navigation and riding assistance systems truly adapted to the characteristics of both the vehicles and their riders. This dissertation proposes the implementation of a service-oriented frontend to support the professional use of motorcycles. The developed frontend allows professional motorcycle riders to authenticate themselves in the system, consult and manage delivery and pickup services associated with a daily route, as well as visualize the optimized route on an interactive map and monitor navigation in real time. The interface dynamically adapts to updates, efficiently responding to unforeseen changes, such as route deviations, in a clear, fast, and intuitive manner, contributing to the reduction of risks associated with riding, in accordance with the needs and specifications of the A-Mover project (Mobilizing Agenda for the Development of Intelligent Green Mobility Products and Systems), which is focused on the development of innovative technological solutions in the field of sustainable urban mobility.

The proposed solution is integrated with a robust backend, responsible for supplying the interface with a complete and up-to-date database. The adopted methodology is based on an iterative and user-centered approach, designed according to User Interface, User Experience, and User-Centered Design principles, and implemented for the Android platform, with connectivity to the display devices installed on the electric motorcycle. To validate the developed solution, functional, performance, and integration tests were carried out in a controlled environment, focusing on the main application functionalities, such as authentication, task management, route presentation, and delivery status updates. The results obtained demonstrated the robustness and feasibility of the proposed solution, confirming its correct operation.

Key Words: A-Mover; Electric motorcycles; Intelligent navigation system; Logistics service; Mobile application; Sustainable urban mobility; User-centered design; User experience; User interface.

ÍNDICE GERAL

RESUMO	iv
ABSTRACT	vi
ÍNDICE DE TABELAS	xi
ÍNDICE DE FIGURAS	xii
LISTA DE ABREVIATURAS, SIGLAS E ACRÓNIMOS	xv
I. INTRODUÇÃO	17
I.1 Contexto e motivação	19
I.2 Estado atual da investigação	20
I.3 Objetivos gerais e específicos	20
I.4 Estrutura da dissertação	21
II.ENQUADRAMENTO TEÓRICO.....	23
II.1 Utilização de motas elétricas	23
II.2 Desafios da apresentação na condução.....	25
II.3 Experiência do Utilizador (UX) /Interface do Utilizador (UI)	26
II.4 Aplicações de logística	28
II.5 Aplicações de navegações	29
II.6 Dados para aplicações de navegação.....	31
II.7 Algoritmos de otimização.....	34
II.8 Resumo do capítulo	35
III. ARQUITETURA DA SOLUÇÃO	37
III.1 Descrição do problema.....	37
III.2 Requisitos funcionais e não funcionais	38
III.3 Casos de uso	40
III.4 Modelo da base de dados	48
III.5 Arquitetura	49
III.6 <i>Mockups</i>	53

III.7 Problemas de roteamento de veículos	57
III.8 Resumo do capítulo.....	59
IV. IMPLEMENTAÇÃO DA APLICAÇÃO.....	61
IV. 1 Metodologia e processo de desenvolvimento	61
IV. 2 Tecnologia utilizadas	63
IV. 4 Base de dados.....	67
IV. 5 API	67
IV. 6 Desenvolvimento da aplicação.....	69
IV. 7 Resumo do capítulo.....	80
V. TESTE E VALIDAÇÃO	81
V.1 Testes funcionais - Caso de estudo.....	81
V.2 Teste de integração	82
V.3 Testes de performance/desempenho.....	83
V.4 Testes unitários.....	83
V.5 Resultados dos testes - aplicação.....	84
V.6 Algoritmo de otimização	85
V.7 Resumo do capítulo	86
VI. CONCLUSÕES E TRABALHOS FUTUROS	88
VI.1 Conclusões	89
VI.2 Limitações e trabalhos futuros	90
VII. REFERÊNCIAS.....	91
APÊNDICES	98
Apêndice A – Publicações	98
Apêndice B - Tabela síntese com as principais características das aplicações de navegação analisadas	101
Apêndice C – Diagrama entidade-relacionamento base de dados	109
Apêndice D – Protótipo funcional e navegável da aplicação. Layouts criados:	111

Apêndice E – Testes funcionais - Caso de estudo	113
Apêndice F – Testes de integração – APIs	132
Apêndice G – Testes de performance/desempenho	136
Apêndice H – Testes unitários	144
Apêndice I – Testes algoritmos	147

ÍNDICE DE TABELAS

Tabela 1 - Características UI/UX	26
Tabela 2 - Comparação entre UI,UX,DCU	27
Tabela 3 - Quadro comparativo: aplicações de navegação e suporte a APIs	30
Tabela 4 - Caso de uso - AUTENTICAÇÃO DO UTILIZADOR	42
Tabela 5 - Caso de uso - Rotas e tarefas	42
Tabela 6 - Caso de uso - iniciar rota/navegação	43
Tabela 7 - Caso de uso - AtualizaR estado da tarefa	44
Tabela 8 - Caso de uso - Apresenta rota atualizada.....	45
Tabela 9 - Caso de uso - detalhe da tarefa	46
Tabela 10 - Caso de uso - consultar histórico de tarefas	47
Tabela 11 - Caso de uso - detalhe tarefa histórico.....	48
Tabela 12 - Comparação das Frameworks estudadas	66
Tabela 13 - Características das aplicações de navegação	101
Tabela 14 - Custo arranque aplicação.....	137
Tabela 15 - Custo carregamento da lista de tarefas	139
Tabela 16 - Custo carregamento da lista de tarefas	142

ÍNDICE DE FIGURAS

Figura 1 - Como funcionam as APIs de tráfego – criado pela autora	32
Figura 2 - Diagrama de caso de uso	41
Figura 3 - Diagrama entidade-relacionamento base de dados.....	49
Figura 4 - Arquitetura do sistema.....	52
Figura 5 - <i>Mockups (a) ecrã de autenticação e (b) Home</i>	54
Figura 6 - <i>Mockups das (a) tarefas e do (b) histórico</i>	55
Figura 7 - <i>Mockup de detalhe da tarefa</i>	56
Figura 8 - <i>Mockup navegação</i>	56
Figura 9 - <i>Mockup navegação display do motociclo</i>	57
Figura 10 - Metodologia de desenvolvimento.....	62
Figura 11 - Arquitetura: tecnologias utilizadas	63
Figura 12 - Tabelas <i>routes</i> e <i>routes_files</i>	67
Figura 13 - Parte código-fonte da API	68
Figura 14 - Parte ficheiro JSON	69
Figura 15 - Diagrama de sequência autenticar utilizador	70
Figura 16 - <i>Interface de autenticação da aplicação</i>	71
Figura 17 - Diagrama de sequência consultar rota e tarefas.....	72
Figura 18 - <i>Interface rota do dia</i>	73
Figura 19 - Diagrama de sequência iniciar navegação/rota.....	74
Figura 20 - <i>Interface de navegação</i>	75
Figura 21 - Diagrama sequência alterar estado tarefa	76
Figura 22 - <i>Interfaces (a) detalhe da tarefa (b) finalizar tarefa</i>	77
Figura 23 - <i>Interface rota atualizada</i>	78
Figura 24 - <i>Interfaces (a) detalhe da tarefa (b) chamada ao cliente</i>	79
Figura 25 - <i>Interface Histórico</i>	80
Figura 26 - Exemplo de teste efetuado - validação de credenciais.....	82
Figura 27 - Confirmação de <i>logs</i> ok do Supabase.....	83
Figura 28 - Android Profiler.....	83
Figura 29 - Teste Sumary LoginViewModel	84
Figura 30 - Diagrama entidade-relacionamento	110
Figura 31 - Ecrã inicial	111
Figura 32 - Ecrã login.....	111

Figura 33 - Ecrã home	111
Figura 34 - Ecrã tarefas	112
Figura 35 - Ecrã histórico	112
Figura 36 - Ecrã detalhe tarefa	112
Figura 37 - Ecrã navegação	112
Figura 38 - Ecrã comprovativo.....	112
Figura 39 - (a)Layout login -> (b) Inexistência de credenciais inseridos	113
Figura 40 - (a)Layout login -> (b) Página inicial	114
Figura 41 - Utilizadores-condutores.....	115
Figura 42 - (a) <i>Interface</i> rota diária (b) <i>Interface</i> sem rota	115
Figura 43 - (a) <i>Interface card</i> expandido (b) <i>Interface</i> Detalhe da tarefa.....	116
Figura 44 - (a) <i>Interface</i> câmara fotográfica (b) <i>Interface</i> assinatura	117
Figura 45 - <i>Interface pop-up</i> mudança de estado tarefa	118
Figura 46 - (a) <i>Interface</i> rota inicial (b) <i>Interface</i> rota “otimizada”.....	119
Figura 47 - (a) <i>Interface</i> visualização prévia (b) <i>Interface</i> navegação.....	120
Figura 48 - (a) <i>Interface</i> histórico (b) <i>Interface</i> detalhe tarefa - histórico	121
Figura 49 - <i>Interfaces</i> (a) histórico (escolha data) (b) histórico filtrado por data	122
Figura 50 - <i>Interfaces</i> (a) histórico (escolha data) (b) histórico filtrado por data	123
Figura 51 - <i>Interfaces</i> (a) detalhe da tarefa (b) Chamada ao cliente	124
Figura 52 - <i>Interface</i> registar ocorrências	125
Figura 53 - <i>Interfaces</i> (a) notificação fora da aplicação (b) mensagem fora da aplicação	126
Figura 54 - Resultados teste instrumentados de funcionalidades Home	127
Figura 55 - <i>Interface</i> login em ecrã <i>phone</i>	128
Figura 56 - <i>Interface</i> login em ecrã dobrável	129
Figura 57 - <i>Interface</i> login em ecrã <i>tablet</i>	130
Figura 58 - <i>Interface</i> home em ecrã <i>tablet</i>	130
Figura 59 - API Gateway Supabase	132
Figura 60 - Registo das chamadas API Supabase durante processo de atualização do histórico	133
Figura 61 - Resposta do endpoint da API FastAPI - rotas	134
Figura 62 - Resposta do endpoint da API FastAPI - rotas	134
Figura 62 - Tabela tasks confirmação de alteração de estado	135
Figura 63 - Dashboard Firebase	136

Figura 64 -Análise Capture System Activities - arranque aplicação.....	137
Figura 65 - Análise View Live Telemetry - arranque aplicação	138
Figura 66 - Análise Capture System Activities - carregamento lista/rota.....	139
Figura 67 - Análise View Live Telemetry - carregamento lista/rota.....	139
Figura 68 - Análise Capture System Activities - rotas	140
Figura 69 - Análise View Live Telemetry - rotas.....	140
Figura 70 - Análise Capture System Activities - Atualização rotas.....	141
Figura 71 - Análise View Live Telemetry - atualização rotas.....	142
Figura 72 - Resultados teste desempenho arranque da aplicação.....	143
Figura 73 - Resultados teste unitário LoginViewModel	144
Figura 74 - Resultados teste unitário TaskRemoteViewModel.....	145

LISTA DE ABREVIATURAS, SIGLAS E ACRÓNIMOS

API – Application Programming Interface (Interface de Programação de Aplicações)

DCU – Design Centrado no Utilizador

DVRP – Dynamic Vehicle Routing Problem (Problema de Roteamento Dinâmico de Veículos)

GPS – Global Positioning System (Sistema de Posicionamento Global)

GRASP – Greedy Randomized Adaptive Search Procedure

HCI – Human-Computer Interaction (Interação Humano-Computador)

JSON – JavaScript Object Notation

MDVRP – Multi Depot Vehicle Routing Problem (Problema de Roteamento de Veículos com Múltiplos Depósitos)

ODS – Objetivos de Desenvolvimento Sustentável

ONU – Organização das Nações Unidas

TSP – Travelling Salesman Problem (Problema do Caixeiro-Viajante)

UI – User Interface (Interface do Utilizador)

UX – User Experience (Experiência do Utilizador)

VRP – Vehicle Routing Problem (Problema de Roteamento de Veículos)

CVRP – Capacitated Vehicle Routing Problem (Problema de Roteamento de Veículos com Capacidade)

VRPTW – Vehicle Routing Problem with Time Windows (Problema de Roteamento de Veículos com Janelas Temporais)

I. INTRODUÇÃO

O crescimento acelerado da população do meio urbano tem vindo a intensificar, de forma significativa. Segundo a Organização das Nações Unidas (United Nations, 2025), cerca de 45% da população mundial vive atualmente em meios urbanos, muitos dos quais se encontram em crescimento acelerado, prevendo-se que dois terços do crescimento populacional mundial, entre 2025 e 2050, ocorrerão em cidades. Este cenário coloca desafios cada vez mais complexos ao cumprimento dos Objetivos de Desenvolvimento Sustentável (ODS), particularmente no que respeita à sustentabilidade das cidades.

A mobilidade urbana sustentável assume-se, assim, como um desafio emergente, diretamente associado ao Objetivo 11 — Cidades e Comunidades Sustentáveis da ODS. No âmbito deste objetivo, a meta 11.6 estabelece que, até 2030, deve-se reduzir o impacto ambiental negativo *per capita* nas cidades, com especial atenção à qualidade do ar, à gestão de resíduos urbanos e a outros fatores ambientais (Objetivos de Desenvolvimento Sustentável, 2025). O cumprimento desta meta está fortemente associado à promoção de soluções de mobilidade mais limpas, destacando-se a mobilidade elétrica como uma alternativa para a diminuição das emissões de dióxido de carbono e dos níveis de ruído urbano.

De acordo com a proposta de resolução do Parlamento Europeu sobre o novo quadro da União Europeia para a mobilidade urbana, as novas medidas a implementar devem fundamentar-se em avaliações de impacto abrangentes, que considerem simultaneamente a dimensão económica, social e ambiental, bem como a diversidade das necessidades de mobilidade da população. Adicionalmente, a avaliação do desempenho ambiental das diferentes soluções de transporte urbano deve basear-se nas emissões ao longo do ciclo de vida dos veículos e em indicadores ambientais relevantes. A resolução salienta ainda os benefícios adicionais da redução do tráfego, em particular permitindo o acesso dos veículos de emergência (Parlamento Europeu, 2023).

O tráfego caótico constitui um dos principais problemas das cidades modernas. Falar de sustentabilidade urbana não implica apenas refletir sobre os aspetos ambientais e económicos de uma sociedade, mas também considerar as dimensões sociais, nomeadamente a equidade no acesso aos serviços, a segurança e a qualidade de vida. A facilidade de deslocação, seja ao nível da logística de última milha, seja na resposta a serviços urgentes, deve ser assegurada de forma rápida, eficiente e sustentável. Para Black et al. (2002), a

mobilidade urbana sustentável deve garantir o acesso a bens e serviços de maneira eficiente para todos os habitantes urbanos, protegendo simultaneamente o meio ambiente, o património cultural e os ecossistemas para a geração presente, sem, no entanto, limitar as oportunidades das gerações futuras. Essa perspetiva demonstra que a problemática da mobilidade sustentável não é recente, constituindo um desafio discutido há várias décadas e que, ainda hoje, se encontra em contínuo processo de desenvolvimento e aperfeiçoamento (Vigário et al., 2024).

O caminho passa pela micromobilidade, veículos pequenos e leves, e que são concebidos maioritariamente para viagens curtas e urbanas. Esta tendência visa preencher lacunas no sistema de transporte público e oferecer alternativas mais práticas e sustentáveis para as deslocações urbanas, especialmente em áreas congestionadas, neste cenário, as motas elétricas surgem como uma alternativa promissora (Grupo Ageas Portugal, 2023). Elas oferecem vantagens em termos de sustentabilidade e da eficiência operacional.

No entanto, a condução de veículos de duas rodas em áreas urbanas torna-se uma tarefa ainda mais complexa que exige uma elevada capacidade de concentração, habilidades de antecipação e atenção redobrada dos condutores. Ledesma et al. (2023) destacam que, apesar da extensa investigação sobre distração em condutores automóveis, a literatura científica sobre motociclistas ainda é limitada. No seu estudo, realizado com 418 motociclistas espanhóis, os autores analisaram erros de condução associados à distração, com os erros de condução relacionados com falta de atenção (ARDES_M), envolvimento em atividades distrativas (IDA), como a utilização de sistemas de navegação e a audição de música, bem como lapsos de atenção no quotidiano (ARCES) e histórico de acidentes e quase-acidentes. Os resultados mostram uma correlação significativa entre o envolvimento em atividades distrativas e o aumento da frequência de erros de condução, sendo a utilização de sistemas de navegação/mapas e a audição de rádio ou música as práticas mais comuns. Embora menos frequentes, a utilização do telemóvel revelou igualmente associação aos erros de condução.

Os autores concluem que as plataformas tecnológicas e dispositivos utilizados durante a condução representam fontes relevantes de distração também para os motociclistas. A natureza predominantemente visual e interativa destas *interfaces* obriga ao condutor a desviar o olhar da estrada, comprometendo o desempenho, a perceção do ambiente envolvente e, consequentemente, a segurança da condução. No estudo efetuado por Useche et al. (2025), concluíram que a crescente integração de tecnologias digitais está a transformar o perfil do

motociclista, aumentando a sua exposição a riscos de natureza cognitiva e comportamental. Adicionalmente, o *stress* laboral revelou-se um preditor significativo da ocorrência de distrações digitais. Perante os resultados, os autores defendem a necessidade de implementar estratégias de prevenção diferenciada por género, bem como políticas de segurança rodoviária que considerem não apenas fatores individuais, mas também as condições laborais associados à mobilidade diária.

I.1 Contexto e motivação

Os sistemas de navegação têm evoluído significativamente, passando de ferramentas básicas de cartografia para plataformas inteligentes integradas com dados em tempo real. Ainda que se verifique uma crescente inclusão de funcionalidade específicas para veículos elétricos, como a definição de rotas com pontos de carregamento, a adaptação destas soluções ao contexto particular dos motociclos elétricos permanece incipiente. Os navegadores generalistas oferecem atualmente uma experiência de condução mais eficiente, personalizada e potencialmente mais segura, na sua maioria, ainda concebidos com foco prioritário na condução de veículos ligeiros (Vigário A. e al., 2025a).

A utilização de motociclos elétricos deve ser acompanhada pela implementação de medidas que garantam a segurança dos motociclistas. Para que as *interfaces* de assistência ao condutor cumpram efetivamente o seu papel de apoio à condução, sem comprometer as limitações e desafios enfrentados na condução urbana, estas devem ser projetados de maneira simples, intuitiva e com elevada legibilidade visual (Chilro et al., 2024).

A complexidade da utilização de dispositivos de informação, como pequenos ecrãs, *smartphones*, acoplados ao guiador, dificulta a consulta de informações durante a condução e constituem uma fonte de distração. A necessidade de desviar o olhar da via para interpretar informação apresentada em interfaces pouco otimizadas compromete a perceção do ambiente envolvente e aumenta o risco de ocorrência de acidentes. As exigências específicas associadas a condução desse tipo de veículo são consideradas elevadas, tornando o desenvolvimento de soluções dedicadas particularmente desafiante (Chadalavada et al., 2025).

Sendo assim, torna-se fundamental o desenvolvimento de soluções tecnológicas que conciliem eficiência operacional, sustentabilidade ambiental e segurança rodoviária, minimizando a carga cognitiva do condutor e promovendo uma interação intuitiva e segura. É neste enquadramento que o trabalho se insere, procurando responder aos desafios atuais da

mobilidade urbana sustentável através de integração de soluções inteligentes aplicadas à logística urbana com recurso a motas elétricas.

I.2 Estado atual da investigação

A literatura científica recente evidencia um interesse crescente nas áreas da mobilidade sustentável (Chen et al., 2025), dos veículos elétricos (International Energy Agency [IEA], 2025) e do *Design* Centrado no Utilizador (DCU) (Emma, 2024). Estudos demonstram que a apresentação inadequada de informação durante a condução aumenta significativamente o tempo de reação e a probabilidade de erro (Ito et al., 2019).

Investigação realizada no domínio das *interfaces* para motociclos destaca a importância da redução da complexidade visual, da utilização criteriosa de alertas sonoros e da antecipação adequada das instruções (Ito et al., 2019). O *design* centrado no utilizador reforça que os sistemas concebidos com base nas necessidades reais dos utilizadores apresentam melhores níveis de aceitação, usabilidade e segurança percebida (Emma, 2024).

No entanto, apesar dos avanços verificados, existe uma escassez de soluções que articulem as funcionalidades de navegação, e de otimização de rotas adaptadas ao contexto específico dos motociclos elétricos, especialmente num domínio de serviços urbanos.

I.3 Objetivos gerais e específicos

A presente dissertação tem como objetivo geral o desenvolvimento de um sistema de logística/navegação inteligente e centrado no utilizador, especificamente concebido para motociclos elétricos utilizados em ambiente urbano. Deste modo, o trabalho propõe a criação de um *frontend* de serviços para apoio à utilização profissional de motociclo, integrado com um *backend* robusto que alimentará a *interface* com uma base de dados completa e atualizada. Desenhar uma *interface* adaptável ao utilizador, nos dispositivos acoplados à mota elétrica, com alguma aprendizagem de acordo com a utilização/interação. Os objetivos específicos são:

- Criar uma *interface* de utilizador baseada em princípios de *Interface* do Utilizador (UI), Experiência do Utilizador (UX) e DCU adequada às restrições visuais, temporais e cognitivas dos condutores de motociclos em meio urbano;
- Criar um *frontend* com informação útil na condução de motas elétricas;

- Integrar funcionalidades de navegação adaptadas a veículos elétricos, incluindo planeamento de rotas com pontos de carregamento e otimização de percursos urbanos.
- Suportar operações de logística urbana, nomeadamente a gestão dinâmica de rotas, tarefas e entregas em tempo real.
- Avaliar o impacto da solução proposta ao nível da usabilidade, segurança percebida e eficiência operacional.

Em síntese, este trabalho tem como objetivo o desenvolvimento de uma *interface* intuitiva, de fácil utilização e, acima de tudo, segura, destinada a apoiar a condução de motociclos no contexto dos serviços em meio urbano. A solução proposta procura promover simultaneamente a segurança do condutor e a proteção do meio envolvente, minimizando a distração, a carga cognitiva e os riscos associados à interação com sistemas digitais durante a condução.

I.4 Estrutura da dissertação

Esta dissertação encontra-se dividida em sete capítulos. No presente capítulo, a Introdução, introduz-se o tema principal deste trabalho, bem como o contexto e motivação para o desenvolvimento da solução proposta.

O Capítulo II apresenta o enquadramento teórico das principais áreas associadas ao tema central deste trabalho, estabelecendo base conceptual necessária para a compreensão das soluções técnicas e metodológicas adotadas ao longo do desenvolvimento do sistema proposto, abordam os conceitos centrais, princípios e desafios inerentes ao desenvolvimento de *interface* para dispositivos móveis utilizados em ambiente da logística.

No Capítulo III é apresentado a arquitetura, a descrição do problema, o levantamento de requisitos, a descrição dos casos de uso, a descrição do modelo da base de dados. Apresentado a arquitetura do sistema e mockups realizados.

No Capítulo IV é apresentado a implementação da aplicação, expondo as metodologias e processo de desenvolvimento, as tecnologias utilizadas, os estudos dos algoritmos de otimização, base de dados, APIs , seguido do desenvolvimento da aplicação.

No Capítulo V é apresentado os testes e validação funcionais, de integração e de performance efetuados à aplicação, apresentando, por fim, os resultados conseguidos.

No Capítulo VI é apresentado a conclusão desta dissertação, demonstrando os objetivos conseguidos, bem com as limitações expostas durante o desenvolvimento do *frontend* e trabalhos futuros.

II.ENQUADRAMENTO TEÓRICO

Este capítulo tem como objetivo apresentar os principais conceitos na área das aplicações de logísticas, enquadrando características fundamentais e relevância dos sistemas, bem como o papel dos motociclos elétricos e das tecnologias digitais no contexto da mobilidade urbana. Para esse efeito, é realizada uma revisão da literatura científica e técnica, com vista à identificação das informações conceptuais mais relevantes para a conceção da aplicação proposta, bem como uma análise aprofundada do estado da arte, abrangendo não apenas exemplos de aplicações existentes no domínio da logística, mas também os componentes essenciais envolvidos na sua conceção, desenvolvimento e implementação.

A secção II.1 analisa os benefícios dos veículos de duas rodas, a secção II.2 apresenta os desafios inerentes à apresentação de informação durante a condução. Na secção II.3 explora os princípios de DCU aplicados às interfaces digitais. Nas secções II.4 e II.5 procedem-se à caracterização das principais plataforma, as de logística e navegação atualmente disponíveis, identificando as suas funcionalidades, limitações e adequação ao contexto específico dos motociclos elétricos. A secção III.6 apresenta as arquiteturas subjacentes aos sistemas de aquisição e distribuição de dados de tráfego em tempo real. Complementarmente, na secção II.7 é apresentado os principais algoritmos de otimização de rotas. Este enquadramento permite estabelecer uma base conceptual sólida para a definição dos requisitos funcionais e tecnológicos da solução proposta nesta dissertação. Por fim, na secção II.8 é apresentado um resumo do capítulo.

II.1 Utilização de motas elétricas

Em comparação com os autocarros e outros meios transportes públicos, estes veículos destacam-se como um modo de transporte particularmente adequado às exigências de alta flexibilidade da mobilidade urbana. A utilização de motociclos permite colmatar lacunas existentes nos sistemas de transporte público e estabelecer ligações eficientes. Na Bélgica, um estudo de 2012 descobriu que se apenas 10% dos condutores trocassem seus automóveis por veículos de duas ou três rodas, o tempo gasto no trânsito diminuiria em 40% e, quando 25% dos condutores efetuassem a troca, o congestionamento seria totalmente eliminado. Os

benefícios em termos de tempo na rede rodoviária belga foram estimados em cinquenta milhões de euros por ano (International Motorcycle Manufacturers Association, 2025).

Segundo o Relatório de Estado, da Organização Mundial da Saúde, de 2023, o número de veículos motorizados de duas e três rodas quase triplicaram, com um aumento de 175% desde 2011. O aumento global é impulsionado pela região do Sudeste Asiático, com um aumento de 273% e pela região das Américas, com um aumento de 217%, na Europa, o aumento foi de 142% (World Health Organization, 2025). Apesar deste crescimento, a taxa de utilização de motociclos nas cidades europeias ainda é baixa quando comparada com a observada em diversas cidades asiáticas, onde estes veículos desfrutam de maior aceitação social e estatuto funcional.

Para além do seu contributo para a mobilidade urbana, os veículos de duas e três rodas desempenham um papel relevante no impulso do crescimento económico e social, sendo frequentemente utilizados na prestação de serviços essenciais. A sua utilização como veículos comerciais, aumentou significativamente na última década em várias grandes cidades, como Barcelona e Paris, onde são comuns operações de pequenas entregas logísticas e frotas de “*mototaxi*”. A facilidade de manobra destes veículos permite a circulação em condições de estradas difíceis e espaços reduzidos, sendo igualmente fundamentais em contextos de emergência, nomeadamente no transporte de pessoas feridas, na entrega de suprimentos básicos e no apoio imediato após catástrofes naturais (Safer Motorcycling, 2025).

A literatura emergente evidencia ainda que os motociclos oferecem benefícios de acessibilidade quando comparados com o automóvel individual, e em certos contextos, com o próprio transporte público. Em cidades densamente povoadas da Ásia e da Europa, estes veículos revelam-se mais acessíveis, flexíveis, fiáveis e muitas vezes mais rápidos do que os veículos ligeiros. Estudo realizado por Kasalawat e Shahi (2024), utilizando o conceito de *Motorcycle Equivalent Unit (MEU)*, demonstrou que, em média, um motociclo ocupa 3,32 vezes menos espaço viário do que os automóveis, ocupando igualmente menos área de estacionamento. A médio e longo prazo, a mobilidade baseada em veículos de duas rodas poderá influenciar positivamente a utilização do solo e o desenvolvimento urbano, contribuindo para cidades mais densas, funcionais e habitáveis.

Apesar destas vantagens, a inexistência de sistemas de navegação verdadeiramente eficazes para motociclos constitui uma lacuna amplamente reconhecida na literatura. As

soluções atualmente disponíveis apresentam limitações significativas na forma como transmitem informações ao condutor, não garantindo uma comunicação clara, rápida e segura dos dados essenciais à navegação. Para ultrapassar este problema, é fundamental que o *design* da apresentação da informação considere as especificidades da condução de motociclos, nomeadamente o movimento contínuo do campo visual entre a estrada e o painel de instrumentos, bem como as restrições temporais impostas pela dinâmica do tráfego urbano (Dias et al, 2025).

II.2 Desafios da apresentação na condução

O comportamento visual do condutor assume particular relevância, Ito et al. (2019) analisaram o movimento dos olhos e o tempo de observação necessários para a correta interpretação de informação de navegação, concluindo que esta apresenta a uma distância compreendida entre 25 metros e 85 metros antes de um ponto crítico, como uma interseção, sendo 40 metros considerado o valor padrão mais adequado para uma velocidade de 30 km/h. Este intervalo permite ao condutor identificar, interpretar e reagir à informação apresentada com um nível aceitável de segurança.

Contudo, quando a informação é apresentada apenas a 10 metros, à mesma velocidade, o tempo disponível para processamento é inferior a um segundo. Este período revela-se insuficiente para que o condutor interprete adequadamente, tome a decisão, desacelere e execute a manobra de forma segura. Tal situação aumenta significativamente a probabilidade de comportamentos de risco, como mudanças abruptas de direção ou travagens súbitas.

Adicionalmente, a utilização de dispositivos de apresentação de informação durante a condução implica um aumento da carga cognitiva, uma vez que o condutor necessita de dividir os seus recursos mentais entre a tarefa primária de controlo do veículo e a tarefa secundária de interpretação das informações apresentadas. Considerando que a capacidade cognitiva humana é limitada, aumentos substanciais desta carga estão associados à diminuição da atenção dedicada ao ambiente envolvente, à redução da perceção de aumento do tempo de reação, fatores diretamente relacionados com a ocorrência de situações potencialmente perigosas (Gebretsadik & Suzuki, 2024).

Deste modo, a literatura converge na necessidade de que os sistemas de navegação para motociclos devem priorizar a minimização da complexidade visual e cognitiva das *interfaces*, promovendo soluções que sejam não apenas informativas, mas também intuitivas, rapidamente interpretáveis e compatíveis com as limitações humanas em contextos dinâmicos de condução (Connected Motorcycle Consortium, 2023). Essas aplicações devem ser centradas no utilizador, priorizando a sua utilidade e funcionalidades específicas a cada tipo de utilização, de forma a garantir elevados níveis de satisfação e eficácia na interação com o sistema.

II.3 Experiência do Utilizador (UX) /Interface do Utilizador (UI)

No domínio da interação humano-computador (*Human-Computer Interaction* -HCI), a integração coerente entre a UX, a UI e o DCU é determinante para o desenvolvimento de sistemas eficazes. Como apresentado na **Tabela 1**, características de UI e de UX, a última refere-se ao conjunto de perceções, emoções e respostas resultantes da interação do utilizador com um produto ou serviço, abrangendo todo o seu percurso, desde a intenção inicial de utilização até a experiência pós-utilização (Vigário et al., 2026). Esse conceito inclui aspetos como usabilidade, acessibilidade, utilidade e desejabilidade. Por sua vez, a UI corresponde aos elementos visuais e interativos com os quais o utilizador interage diretamente, tais como botões, ícones, tipografia, cores e disposição gráfica (UserTesting, 2023).

Tabela 1 - Características UI/UX

UI	UX
Design visual e interativo	Experiência geral do utilizador
Botões, cores, ícones, fontes	Facilidade de utilização e navegação
Layouts e resposta	Pesquisa, testes e <i>feedback</i>
Estética e identidade visual	Eficiência na jornada do utilizador

O DCU assume particular relevância enquanto metodologia de conceção que prioriza as necessidades reais, preferências e limitações do utilizador final ao longo de todo o processo de desenvolvimento de sistemas digitais, visando compreender o contexto da utilização e produzir sistemas inteligentes capazes de resolver problemas efetivamente relevantes (Falcone

& Miola, 2024). Na **Tabela 2**, pode-se ver a comparação entre esses três conceitos, dando ênfase ao tipo, quando ocorre, objetivo e parâmetros.

Na era digital, o sucesso de muitos produtos depende fortemente da qualidade da experiência do utilizador. Assim, UI,UX e DCU não devem ser encarados como conceitos isolados, mas como componentes complementares de um mesmo processo de desenvolvimento orientado à criação de soluções tecnológicas funcionais e centradas nos utilizadores. Contudo, os sistemas de visualização atualmente disponibilizados por diversos fabricantes oferecem, na sua maioria, apenas funcionalidades básicas, como a apresentação do histórico de manutenção, avisos de serviço futuro e atualizações pontuais do sistema, revelando um reduzido foco na otimização da experiência de navegação e apoio à condução (Chilro et al., 2024; Vigário et al, 2026).

Tabela 2 - Comparação entre UI,UX,DCU

Aspeto	UI	UX	DCU
Tipo	Componente visual/interativa	Resultado	Processo
Quando ocorre	Durante a utilização do sistema	Durante e após a utilização	Durante o desenvolvimento
Objetivo	Permitir interação clara e intuitiva	Boa Experiência	Criar um sistema adequado ao utilizador
Parâmetros	Avalia-se	Mede-se	Aplica-se

Num cenário urbano dinâmico, caracterizado por rápidas mudanças nas condições de trânsito e ambientais, é essencial que o motociclista disponha de *interfaces* que permitam uma interação simples, acesso imediato à informação crítica e controlo efetivo sobre a personalização da sua experiência. Estas características são determinantes para assegurar níveis adequados de segurança e conforto durante a condução de motociclos elétricos (Dias et al., 2025). No contexto da logística, estas soluções criam um fator diferencial para os utilizadores, criando *interfaces* específicas, adaptadas às exigências da atividade.

II.4 Aplicações de logística

Existem várias aplicações móveis focadas em otimizar a logística, gestão de entregas e recolhas. Estas plataformas oferecem funcionalidades avançadas de atribuição de tarefas, monitorização em tempo real e registo do estado das encomendas, contribuindo significativamente para o aumento da eficiência operacional. No entanto, apesar da vasta oferta disponível no mercado, a maioria destas soluções não foi concebida especificamente para a utilização por condutores de motociclos elétricos, nem considera de forma aprofundada as particularidades deste tipo de veículo, como as limitações de autonomia, a necessidade de planeamento de carregamentos, as exigências de segurança acrescidas e as restrições associadas à interação com interfaces durante a condução (Vigário et al., 2025a).

Aplicações de grande escala, como a Glovo e a Uber Eats, apresentam fluxos operacionais mais complexos e fortemente estruturados (Glovo, 2025; Uber Eats, 2025). No caso da Glovo, o estafeta dispõe de informação detalhada sobre horários de trabalho, lista de pedidos, estado das recolhas e entregas, bem como mecanismos de reporte de problemas, como a ausência do cliente ou dificuldades no local de recolha. A aplicação apenas revela a morada de entrega após a recolha estar concluída e utiliza sistemas de navegação externos, como o Waze, para orientar o condutor. Inclui ainda códigos de confirmação de recolha, confirmação de entrega, envio de recibos e relatórios de atividade.

De forma semelhante, a plataforma Uber Eats exige validações adicionais de segurança, como a confirmação da utilização de equipamento de proteção e a verificação do condutor através da fotografia (Uber Eats, 2025). O sistema apresenta a área de trabalho, mensagens internas, detalhes completos do pedido, códigos de recolha e múltiplos métodos de confirmação de entrega, incluindo um PIN de quatro dígitos ou fotografia do local quando a entrega é realizada sem contacto direto com o destinatário. A navegação é automaticamente iniciada após a recolha, recorrendo a serviços de mapas integrados no dispositivo.

No domínio das soluções empresariais de gestão de frotas e planeamento de rotas, destacam-se plataformas como Routific, Onfleet e OptimoRoute, que oferecem maior controlo operacional e funcionalidades avançadas de planeamento (Routific, 2025; Onfleet, 2024; OptimoRoute, 2025). A Routific disponibiliza uma aplicação móvel onde o condutor pode visualizar as rotas atribuídas, o histórico de entregas e o seu perfil, navegar entre paragens, consultar instruções específicas, recolher prova de entrega através de assinatura ou

fotografia e adicionar notas para o remetente. O sistema suporta ainda o acompanhamento em tempo real por parte dos gestores logísticos.

A aplicação Onfleet segue uma abordagem semelhante, apresentando listas de tarefas, visualização em mapa, início e conclusão de serviços através de interações simples (como “deslizar para iniciar”) e recolha estruturada de comprovativos de entrega, incluindo assinaturas, fotografias e seleção de motivos de falha quando a entrega não é possível. Oferece igualmente funcionalidades de filtragem de tarefas, *chat* interno e progressão automática para a tarefa seguinte (Onfleet, 2024; Onfleet, 2025).

Por sua vez, o OptimoRoute integra planeamento avançado de rotas, com possibilidade de alteração da hora de início pelo condutor, navegação através de múltiplos serviços (Google Maps, Waze, HERE, entre outros), funcionamento em modo *offline*, sincronização automática com o sistema central e recolha detalhada de prova de entrega, incluindo digitalização de códigos de barras, assinaturas digitais, fotografias e notas operacionais. Esta plataforma encontra-se particularmente orientada para contextos profissionais de distribuição e serviços (OptimoRoute, 2025).

Em síntese, embora as plataformas de logística e gestão de entregas estudadas disponibilizem um conjunto abrangente de funcionalidade operacionais, estas tendem a valorizar a coordenação de tarefas e o controlo empresarial, deixando para segundo plano a adaptação da interface e da experiência de utilização aos motociclos elétricos. Assim, as aplicações de navegação assumem um papel complementar e determinante, ao fornecerem suporte à condução.

II.5 Aplicações de navegações

Com a adoção dos *smartphones*, as aplicações de navegação tornaram-se essenciais para a mobilidade. O portal *web* AutoExpress, na sua secção Product Grup Tests, Bernard (2025) procedeu à seleção das aplicações de navegação mais populares disponíveis na Apple App Store e avaliou o desempenho de três modelos distintos de iPhone (do iPhone 12 ao 14). Os resultados indicaram que a maioria dos utilizadores preferiu o Google Maps, cuja posição de liderança é atribuída, sobretudo, à elevada precisão na pesquisa de destinos e à eficácia da navegação curva-a-curva (*turn-by-turn*). Em segundo lugar surgiu o Waze, reconhecido como uma solução particularmente eficaz para utilizadores que pretendem evitar

congestionamentos, beneficiando da sua abordagem em *crowdsourcing* e na atualização contínua de dados em tempo real (Bernard, 2025).

Uma das conclusões mais relevantes do estudo prendeu-se com o desempenho do TomTom AmiGo, que evidenciou uma otimização significativa para cenários de mobilidade elétrica, apresentando elevada compatibilidade com os requisitos específicos desse tipo de veículo. O conjunto de aplicações analisadas incluiu ainda o TomTom GO, Apple Maps, Sygic Car Navigation, MyRoute-App e Yandex Navi.

Paralelamente, o sítio especializado Holiday Extras identificou as dez melhores aplicações gratuitas de navegação para Android e iOS em 2025, nomeadamente: Google Maps, Waze, Apple Maps, Karta GPS – Maps & CarPlay, TomTom GO, ViaMichelin GPS Route Planner, HERE WeGo: Maps & Navigation, Maps.me, MyRoute-App Navigation e Roadtrippers – Trip Planner (Holiday Extras, 2025). No Apêndice desta dissertação apresenta-se uma tabela síntese com as principais características das aplicações de navegação analisadas, permitindo uma comparação estruturada das respetivas funcionalidades, desempenho e abordagens tecnológicas, como apresentado na **Tabela 3**

Tabela 3 - Quadro comparativo: aplicações de navegação e suporte a APIs

Aplicação	Suporte	Funcionalidades Principais	Integração com Veículos	Dados em Tempo Real
Google Maps	Google Maps Data API (directions, geolocation, places, etc.)	Rotas otimizadas, trânsito em tempo real, pontos de interesse, StreetView	Android Auto, Apple CarPlay	☐
Waze	Waze Transport SDK e Waze API	Colaboração em tempo real, alertas de trânsito, acidente, radares	Android Auto, Apple CarPlay	☐
TomTom GO	TomTom Maps APIs (routing, traffic, search)	Navegação <i>offline</i> , alertas de trânsito, zonas de velocidade	Android Auto, Apple CarPlay	☐
Apple Maps	MapKit e MapKit JS	Navegação integrada, sugestões inteligentes, transporte público	Apple CarPlay	☐ (limitado a dispositivos Apple)

As aplicações de navegação são componentes essenciais das soluções de mobilidade urbana, contribuindo para a segurança rodoviária, para a redução dos tempos de deslocação, para a diminuição dos níveis de ruído e das emissões de gases poluentes, bem como para a mitigação do desgaste das infraestruturas viárias (Bagabaldo et al., 2024). No caso de veículos

elétricos, ajudam no planeamento de rotas com menor consumo energético e menor exposição a tráfegos, melhorando, assim, a autonomia das baterias.

Por conseguinte, a gestão inteligente do acesso à informação podem tornar-se um fator determinante para a eficiência operacional e a sustentabilidade dos sistemas (National Traffic Designs, n.d.). É crucial entender como essas *interfaces* adquirem dados, como os processam e de que formas são otimizadas e entregues ao utilizador final.

II.6 Dados para aplicações de navegação

As aplicações de navegação ajudam os seus utilizadores a encontrar o percurso mais eficiente para os seus destinos, planeando o itinerário, minimizando o tempo de viagem. No entanto, a sua funcionalidade vai além da simples identificação do caminho mais curto, ao incorporar uma série de características avançadas. Essas aplicações estão integradas com dados em tempo real, como condições de trânsito, atualizações meteorológicas, acidentes, encerramento de estradas e pontos de interesse. Utilizam tecnologias de geolocalização, algoritmos de otimização de rotas e análise preditiva para fornecer rotas adaptadas às condições reais de tráfego (Bagabaldo et al, 2024).

Estas tecnologias utilizam *Interfaces* de Programação de Aplicações (APIs) que permitem a exposição e o intercâmbio controlado de dados, serviços e funcionalidade, permitindo que estas aplicações interajam programaticamente de forma consistente e estruturada, com um conjunto normalizado de *interfaces*, protocolos e convenções, facilitando a integração entre diferentes componentes ou sistemas de *software* (Goodwin, 2024). Adicionalmente, asseguram que apenas a informação necessária é partilhada, mantendo ocultos os detalhes internos dos sistemas, o que contribui para a modularidade, segurança e escalabilidade das soluções tecnológicas (Autobot, 2024).

As APIs de navegação podem aceder a informações em tempo real, processá-las e apresentar ao utilizador os dados relevantes (SoftCircles, 2024). Assim, a interoperabilidade entre aplicações e APIs é um dos pilares fundamentais da mobilidade inteligente e sustentável. O funcionamento dos sistemas de dados de tráfego em tempo real pode ser dividido em três fases principais: aquisição de dados brutos, processamento e análise e, distribuição de dados via APIs (Gong & Ding 2025;Teledjieu & Shafique, 2025; Renninger et al., 2024) (**Figura 1**).

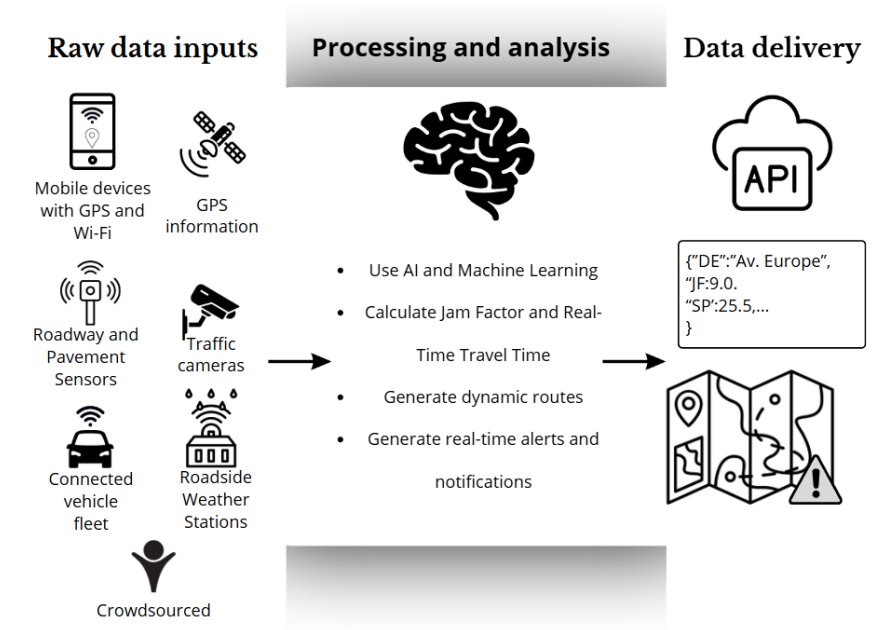


Figura 1 - Como funcionam as APIs de tráfego – criado pela autora

As APIs obtêm dados a partir de múltiplas fontes heterogêneas, entre as principais encontram-se os sistemas de posicionamento global (*Global Positioning System* - GPS), entidades públicas, sensores instalados em infraestruturas urbanas, câmaras de vigilância de tráfego, estações meteorológicas e bases de dados históricas (National Traffic Designs, n.d.; Sunday, 2024a).

Os dados de GPS são obtidos a partir de dispositivos instalados em automóveis, camiões, bem como em dispositivos móveis e sistemas de navegação pessoais. Por exemplo, na aplicação TomTom GO, são recolhidos diariamente 61 mil milhões de pontos de dados GPS provenientes de *smartphones*, sistemas integrados em veículos, dispositivos de navegação pessoal (PNDs) e frotas, cobrindo mais de 3,5 mil milhões de quilómetros de vias em todo o mundo. Uma fonte adicional importante é o *Spatial Data Object* (SDO), utilizado, por exemplo, para o reconhecimento de sinais de trânsito, contribuindo para a deteção mais eficiente de incidentes rodoviários, como obras na via pública. Estes dados permitem traçar uma imagem em tempo real do fluxo e da densidade do tráfego, prevendo não só congestionamentos, mas também cortes de estrada, intervenções em curso e outros incidentes relevantes (TomTom, 2025).

Paralelamente, muitas plataformas recorrem intensivamente ao *crowdsourcing*, permitindo que os utilizadores reportem acidentes, obras e outros incidentes da via.

Aplicações como o Waze, por exemplo, recebem continuamente dados anónimos sobre posição, velocidades e eventos rodoviários pelos próprios utilizadores. Estas contribuições humanas complementam os dados automáticos provenientes de sensores e dispositivos (Suday, 2024b).

Além disso, dados de entidade governamentais, como manutenção programada de estradas, restrições temporárias, como por exemplo, devido a eventos desportivos, são complementados através de contadores e sensores de tráfego instalados em pontos estratégicos, como pontes, cruzamentos e vias rápidas, que registam variáveis como o número e tipo de veículos, velocidade média e padrões de tráfego. Os sensores podem incluir *loops* indutivos embutidos no pavimento, sensores infravermelhos, câmaras com visão computacional, sensores pneumáticos ou de radar, bem como dados provenientes de veículos conectados (FCD) (National Traffic Designs, n.d.).

São utilizados dados provenientes de câmara de vigilância, que captam imagens e vídeos em tempo real sobre ocupação das vias, fluxo de tráfego, tipos de veículos e deteção de incidentes. Adicionalmente, as estações meteorológicas rodoviárias, equipadas com sensores que medem temperatura, humidade, vento e precipitação, permitem detetar riscos como acumulação de gelo ou vento forte em regiões montanhosas (National Traffic Designs, n.d.).

Estes dados são guardados e processados em centros de gestão de tráfego e disponibilizados, muitas vezes, através de APIs públicas ou acordos privados com fornecedores de serviços, sendo enviados para servidores centrais, onde são processados utilizando técnicas de inteligência artificial e aprendizagem automática. Os algoritmos interpretam os dados, identificam padrões, calculam tempos estimados de viagem (ETA), detetam incidentes e geram sugestões de rotas alternativas com base nas condições atuais de trânsito (Suday, 2024b). Após o processamento, os dados estruturados e analisados são disponibilizados às aplicações de navegação por meio de APIs.

Por si só, estes dados apresentam utilidade limitada se não forem devidamente analisados e interpretados de forma a produzir informação relevante e passível de ser utilizada pelo utilizador final. A integração de inteligência artificial, algoritmos preditivos e dados de várias fontes permite a previsão de acidentes e a adaptação em tempo real das rotas às necessidades individuais. Os algoritmos de otimização assumem um papel central na

definição dos itinerários, incorporando não apenas dados dinâmicos, mas também inerentes ao serviço e as limitações de autonomia associadas às baterias de veículos elétricos.

II.7 Algoritmos de otimização

A necessidade de promover uma mobilidade mais sustentável tem impulsionado o estudo de novas abordagens para a gestão eficiente das rotas. Neste contexto, a otimização de rotas assume um papel crucial, permitindo minimizar custos de operação, reduzir emissões e melhorar a qualidade do serviço prestado aos utilizadores. Com a transição para a mobilidade elétrica, surgem novos desafios associados às restrições de autonomia, à necessidade de recarga e à capacidade de carga limitada dos veículos. Estas particularidades exigem a adaptação dos modelos tradicionais de otimização, de forma a garantir soluções eficientes que consideram variáveis dinâmicas e restrições energéticas (Kuo, 2025).

Para os condutores de motociclos elétricos, esta capacidade significa uma redução significativa do tempo de deslocação, num menor consumo de energia e redução do *stress* associado à incerteza do itinerário. A otimização de rotas contribui igualmente para mitigação da poluição atmosférica e sonora, ao permitir uma redistribuição mais eficientes dos fluxos de tráfego e a redução da circulação em zonas sensíveis, como áreas residenciais, zonas escolares e proximidades de unidades de saúde, onde a diminuição de tráfego assume particular relevância do ponto de vista ambiental e de saúde pública (Bagabaldo et al, 2024).

A integração de algoritmos inteligentes, incluindo fatores como tempo de viagem, custos de carregamento e condições associadas a postos de carregamento, possibilita a determinação de percursos mais curtos, rápidos e eficientes para veículos elétricos (Rajesh et al., 2022). Em contexto urbano, caracterizado por elevada densidade de tráfego e por restrições frequentes à circulação, a otimização de rotas permite contornar zonas congestionadas e áreas restritas com agilidade, aproveitando a flexibilidade do veículo. Além disso, a escolha automática de percursos com menor intensidade de acelerações e desacelerações, pode contribuir para prolongar a autonomia da bateria, um fator crítico para utilizadores dependentes do veículo para deslocações diárias (Zhao et al., 2025; Vigário et al, 2025c).

A aplicação de técnicas avançadas de otimização de rotas constitui um elemento central para um melhor planeamento logístico, mais sustentabilidade e uma experiência de

condução mais inteligente. A otimização de rotas possibilita a adaptação contínua do comportamento da navegação às preferências do condutor, às condições operacionais do veículo e ao contexto envolvente, promovendo uma experiência de condução simultaneamente mais segura, eficiente e personalizada.

II.8 Resumo do capítulo

O capítulo analisou o papel dos motociclos elétricos, das tecnologias digitais e das aplicações de logística e navegação, através de uma revisão abrangente da literatura e do estado da arte no contexto da mobilidade urbana. Foram analisados os benefícios funcionais, económicos e ambientais dos veículos de duas rodas, destacando-se a elevada flexibilidade operacional, a utilização do espaço viário e o contributo para uma maior mobilidade de última milha com um crescente papel em serviços em áreas urbanas, contribuindo para a redução de congestionamento e emissão de poluentes atmosféricos e sonoros.

Os desafios associados à apresentação de informação durante a condução, evidenciando a relevância dos fatores visuais e cognitivos dos condutores de motociclos e a necessidade de *interfaces* que minimizem a carga mental e disponibilizem uma interpretação rápida e inequívoca da informação. Neste âmbito, foram enquadrados os princípios de UX e UI, mostrando a importância do DCU na conceção de sistemas interativos adaptados às limitações humanas e às exigências de ambientes dinâmicos.

Caracterizou-se as principais aplicações de logística e de navegação disponível no mercado, identificando as suas funcionalidades, limitações e grau de adequação ao contexto dos motociclos elétricos. Abordou-se igualmente a arquitetura dos sistemas de aquisição, processamento e distribuição de dados de tráfego em tempo real, destacando o papel das APIs e das múltiplas fontes, incluindo sensores, sistema GPS, *crowdsourcing* e entidades governamentais, na construção de serviços de navegação inteligentes e atualizados.

Por fim, foram analisadas abordagens recentes de otimização de rotas baseadas em algoritmos, mostrando as suas contribuições para a redução de fatores como o tempo de deslocação, consumo energético e impactos ambientais, bem com a melhoria da experiência de condução e condição do ambiente envolvente.

Em conjunto, este enquadramento teórico e tecnológico, permite identificar de forma objetiva a lacuna existente no desenvolvimento de soluções digitais especificamente concebidas para apoiar a utilização profissional de motociclos elétricos em ambiente urbano. Evidencia-se o potencial contributo destas soluções para a otimização da mobilidade urbana sustentável e para o aumento da eficiência e fiabilidade dos serviços de logística. Neste sentido, o presente enquadramento estabelece uma base sólida para os capítulos seguintes, onde será apresentado a metodologia de desenvolvimento, na secção III e orientando a implementação da aplicação, apresentado na secção IV.

III. ARQUITETURA DA SOLUÇÃO

Neste capítulo é apresentada a descrição do problema, bem como da solução proposta. A metodologia adotada neste trabalho baseia-se numa abordagem de desenvolvimento iterativa e centrada no utilizador, combinando princípios de engenharia de *software*, HCI e sistemas de informação geográfica. O processo metodológico encontra-se estruturado em várias etapas fundamentais, nomeadamente: o levantamento de requisitos funcionais e não funcionais do sistema, a definição dos casos de uso, o modelo de base de dados, modelos de integração e fluxo de integração, a conceção inicial da *interface* do utilizador de acordo com os princípios do DCU, recorrendo à elaboração de *mockups*, a inclusão de rotas otimizadas, e, por fim, a definição da arquitetura global da solução.

III.1 Descrição do problema

A Agenda A-Mover (A-Mover, 2026), visa promover uma transformação substancial no modelo de desenvolvimento industrial na Região de Trás-os-Montes e Alto Douro. Alinhado a responsabilidade ambiental, os avanços na tecnologia, sustentabilidade, procurou-se desenvolver uma solução capaz de comutar os problemas de navegação na condução de um veículo de duas rodas nos serviços de logística, para a utilização de motas elétricas seja em serviços urgentes e não urgentes.

Embora a implementação de uma *interface* para os motociclos elétricos apresente um grande potencial para a melhoria da segurança e da eficiência nos serviços urbanos, enfrenta desafios significativos decorrentes das características intrínsecas deste tipo de veículo. Desafios como a sua menor estabilidade e a necessidade permanente de o condutor manter o foco na estrada e no meio envolvente, são especialmente relevantes. A condução de motociclos constitui uma atividade complexa, envolvendo múltiplas tarefas simultâneas, o que exige uma gestão rigorosa dos recursos para manobrar o veículo em segurança (Gupta & Velaga, 2025). Neste contexto, a utilização de dispositivos de apresentação de informação, como pequenos ecrãs acoplados ao guiador, que poderiam auxiliar na navegação e na obtenção de informações, podem igualmente representar uma fonte de distração se não for devidamente concebida.

Fase a estas limitações, é proposto o desenvolvimento de um *frontend* de mobilidade compacta, funcional e eficiente, especificamente para motociclos elétricos dotado de uma

interface otimizada que permita aos utilizadores aceder, de forma prática e segura, aos serviços essenciais durante a condução. Para lidar com o dinamismo das informações, a interface deve ser flexível, intuitiva e responsiva, capaz de se adaptar a atualizações contínuas e de reagir eficazmente a eventos imprevistos, tais como desvio de rota, alterações nas condições de tráfego ou modificações na disponibilidade de pontos de recarga.

A solução proposta assenta no desenvolvimento um *frontend* orientado para apoio à utilização profissional do motociclo, integrado a um *backend* robusto, responsável pela gestão e fornecimento de dados atualizados através de uma base de dados centralizada. O objetivo consiste na criação de uma *interface* adaptativa e centrada no utilizador, implementada nos dispositivos smartphones com posterior conexão ao *display* da moto, onde será apresentado somente a navegação, com capacidade de apresentar os resultados de modelos de otimização de rotas de forma clara, transparente e intuitiva, baseadas na interação, nos padrões de utilização e perfil do condutor.

III.2 Requisitos funcionais e não funcionais

Os requisitos funcionais definem o que o sistema deve realizar para responder às necessidades do utilizador, especificando as funcionalidades, serviços e comportamentos que a aplicação deve disponibilizar. Estes requisitos descrevem, de forma objetiva, as ações que o sistema deve executar. Por sua vez, os requisitos não funcionais especificam como o sistema deve atuar, representando atributos e características que o define. Os requisitos não funcionais influenciam diretamente a experiência do utilizador e a eficiência global da aplicação, garantindo que o sistema não só cumpre as suas funções, mas o faz de forma adequada e consistente. O levantamento e a definição destes requisitos permitem obter uma compreensão clara das metas definidas para o processo de desenvolvimento.

III.2.1 Requisitos Funcionais (RF)

Os requisitos funcionais definem as funcionalidades essenciais que a aplicação deve disponibilizar. Sendo eles, neste projeto:

[RF1] – Autenticação e sessão

- RF1.1 Login do condutor – através de *email* e password.
- RF1.2 Manter sessão ativa
- RF1.3 Logout.

[RF2] – Gestão de tarefas

- RF2.1 Visualizar detalhes da tarefa associada à rota.
- RF2.2 Atualizar estado da tarefa (Concluída/Não Concluída).
- RF2.3 Registar ocorrências (ex.: cliente ausente, morada incorreta, etc.)
- RF2.4 Registar prova de entrega (assinatura, fotografia, PIN/QR – conforme política)
- RF2.5 Pré-visualizar rota antes de iniciar.

[RF3] – Rota diária e otimização de rotas

- RF3.1 Apresentar rota diária – lista de tarefas atribuídas no dia.
- RF3.2 Visualização de atualização de rota otimizada quando alteração em tarefa – concluída, cancelada, não concluída.

[RF4] – Integração com serviços de mapas

- RF4.1 Visualização de rotas otimizadas na aplicação de navegação

[RF5] – Histórico

- RF5.1 Apresentar o histórico de rotas e tarefas.

[RF6] – Apoio serviço

- RF6.1 Contactar cliente/central a partir da tarefa.
- RF6.2 Centro de ajuda e reporte de problemas técnicos.

[RF7] – Navegação

- RF7.1 Apresentar notificações/alertas

III.2.1 Requisitos Não-Funcionais (RNF)

Os requisitos não funcionais garantem que a aplicação tenha um desempenho adequado e seja segura e escalável.

[RNF1] – Usabilidade e segurança na condução

- RNF1.1 Interface “modo condução” com baixa carga cognitiva e apresentação somente da navegação.
- RNF1.2 Interface redimensionada para desktop e smartphone.

[RNF2] – Desempenho

- RNF2.1 Tempo de abertura da aplicação ≤ 3 segundos.
- RNF2.2 Atualização do estado em $\leq 2-5$ segundos após receção de dados.
- RNF2.3 Interface responsiva para desktop e smartphone.
- RNF2.4 Renderização de mapa ≤ 1 segundo.

[RNF3] – Fiabilidade e disponibilidade

- RNF3.1 Backend com disponibilidade alvo $\geq 99\%$.
- RNF3.2 Tolerância a falhas: tentativas, filas offline.

[RNF4] – Segurança e privacidade

- RNF4.1 Autenticação segura e comunicação HTTPS.
- RNF4.2 Proteção de dados pessoais: minimização, retenção, consentimento.
- RNF4.3 Criptografia de dados sensíveis.

[RNF5] – Escalabilidade

- RNF5.1 Suporte crescimento de utilizadores/tarefas sem degradar a resposta.

[RNF6] – Manutenibilidade e extensibilidade

- RNF6.1 Código modular.
- RNF6.2 API versionada e documentação.
- RNF6.3 Testes (unitários e integração) para rotas, regras e sincronização.

[RNF7] – Compatibilidade

- RNF7.1 Android com versões mínimas definidas
- RNF7.2 Adaptado a diferentes tamanhos de ecrã.
- RNF7.3 Integração com serviços externos via APIs substituíveis

[RNF8] – Acessibilidade

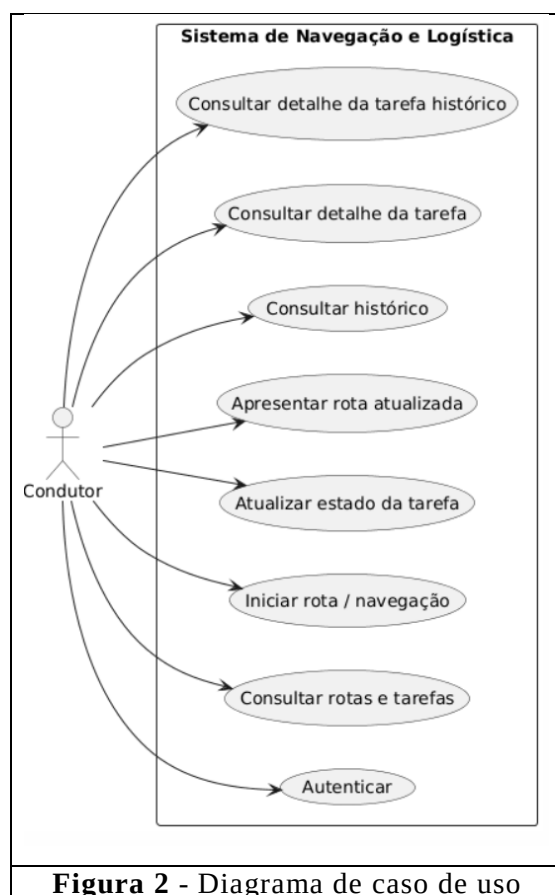
- RNF8.1 *Design* centrado no utilizador (UX/UI).
- RNF8.2 *Design Jetpack Compose*.

[RNF9] – Conformidade e ética

- RNF9.1 Respeito por regras de utilização durante condução
- RNF9.2 Transparência: quando a rota muda e porquê.
- RNF9.3 Língua adaptada ao utilizador.

III.3 Casos de uso

Um caso de uso, **Figura 2**, é uma representação abstrata da interação entre um sistema e os seus utilizadores, descrevendo a forma como o sistema responde às suas ações. O principal objetivo de um caso de uso é relatar a interação entre o sistema e os utilizadores, identificando funcionalidades específicas e orientando o desenho e a arquitetura do *software*. Os elementos incluem atores (utilizadores ou sistemas externos), descrição, fluxos (primário e secundário), pré-condições e pós-condições. A seguir, é apresentado o diagrama e as tabelas referentes a cada caso.



O caso de uso 1 (**Tabela 4**) apresenta o fluxo de autenticar de um utilizador no sistema, o caso de uso 2 (**Tabela 5**) apresenta o fluxo de rotas e tarefas a apresentar ao utilizador, o caso de uso 3 (**Tabela 6**) a iniciar rota/navegação, o caso de uso 4 (**Tabela 7**) atualiza estado da tarefa, o caso de uso 5 (**Tabela 8**), apresenta a rota otimizada, o caso de uso 6 (**Tabela 9**) apresentação detalhes da tarefa caso de uso 7 (**Tabela 10**) apresenta o caso de uso histórico de tarefas e, por fim, o caso de uso 8 (**Tabela 11**) apresenta o caso de uso detalhe da tarefa no histórico. Caso de uso descreve os passos envolvidos, as ações dos utilizadores e as respostas esperadas do sistema.

TABELA 4 - CASO DE USO - AUTENTICAÇÃO DO UTILIZADOR

Caso de uso: Autenticar do utilizador ID:UC1
Descrição: Realizar a autenticação do utilizador no sistema, permitindo às funcionalidades da aplicação.
Atores: Condutor (Motociclista)
Pré-condições: • O utilizador possui uma conta registada no sistema; • Existe ligação à internet.
Fluxo Primário: 1. O sistema apresenta o ecrã de autenticação; 2. O utilizador introduz o endereço eletrónico e a palavra-passe; 3. O sistema envia os dados para o serviço de autenticação; 4. O backend valida as credenciais; 5. O sistema confirma a autenticação; 6. O utilizador acede à aplicação.
Fluxo Secundário: A. Credenciais inválidas 1. O utilizador introduz dados incorretos; 2. O sistema informa que a autenticação falhou; 3. O utilizador é convidado a tentar novamente.
Pós-condições: • O utilizador inicia a sessão com sucesso. • O sistema redirecionado utilizador para a página inicial da aplicação.

TABELA 5 - CASO DE USO - ROTAS E TAREFAS

Caso de uso: Consultar rotas e tarefas ID:UC2
Descrição: Receber automaticamente a rota e as tarefas atribuídas ao condutor, permitindo a sua consulta e posterior execução através da aplicação móvel.
Atores: Condutor (Motociclista)
Pré-condições: • O utilizador encontra-se autenticado; • O sistema de backend encontra-se operacional; • Existe rota atribuída ao condutor para o dia.
Fluxo Primário: 1. O sistema consulta a rota atribuída ao condutor; 2. O sistema envia os dados da rota e respetivas tarefas para a aplicação; 3. A aplicação processa a informação recebida; 4. A aplicação apresenta a rota do dia e a lista de tarefas associadas ao utilizador.
Fluxo Secundário: A. Sem rota atribuída 1. O sistema verifica que não existe rota associada ao condutor; 2. O sistema apresenta uma mensagem informativa ao utilizador.
Pós-condições: • A rota e as tarefas ficam disponíveis para consulta e execução na aplicação; • O utilizador pode selecionar tarefas individuais ou iniciar a navegação da rota.

TABELA 6 - CASO DE USO - INICIAR ROTA/NAVEGAÇÃO

Caso de uso: Iniciar rota/navegação ID: UC3
Descrição: Iniciar rota atribuída ao condutor, ativando a navegação assistida através de uma aplicação de mapas com base na rota otimizada calculada pelo sistema.
Atores: Condutor (Motociclista)
Pré-condições: • O utilizador encontra-se autenticado; • Existe uma rota atribuída; • A rota encontra-se previamente calculada e otimizada pelo sistema; • A ligação ao backend (FastAPI) está disponível. • A ligação à API de mapas/navegação está disponível; • O dispositivo possui uma aplicação de navegação instalada.
Fluxo Primário: 1. O sistema envia para a aplicação a rota otimizada; 2. O condutor seleciona a opção “iniciar rota”; 3. A aplicação solicita a abertura da aplicação de navegação; 4. A aplicação de navegação é aberta com a rota criada pelo sistema; 5. A navegação assistida é iniciada.
Fluxo Secundário: A. Falha ao aceder aplicação de navegação 1. O sistema tenta abrir a aplicação de navegação; 2. A abertura falha (aplicação inexistente ou erro do sistema); 3. O sistema apresenta uma mensagem de erro ao utilizador informando que não foi possível iniciar a navegação.
Pós-condições: A navegação encontra-se ativa e o condutor pode iniciar o percurso da rota atribuída.

TABELA 7 - CASO DE USO - ATUALIZAR ESTADO DA TAREFA

Caso de uso: Atualizar estado da tarefa ID:UC4
Descrição: Atualizar o estado de uma tarefa atribuída ao condutor (por exemplo, Concluída ou Não Concluída), permitindo manter o progresso das entregas sincronizado com o sistema central.
Atores: Condutor (Motociclista)
Pré-condições: • O utilizador encontra-se autenticado; • Existe uma tarefa ativa associado ao utilizador; • A navegação encontra-se terminada ou temporariamente suspensa para permitir a interação com a aplicação; • Existe ligação ao backend; • O utilizador deve sair da navegação.
Fluxo Primário: 1. O condutor acede à lista de tarefas ou ao detalhe da tarefa atual; 2. O condutor seleciona a tarefa pretendida; 3. O condutor escolhe o novo estado da tarefa (ex.: Concluída ou Não Concluída); 4. A aplicação envia a atualização para o sistema backend; 5. O backend regista a alteração na base de dados; 6. O sistema confirma a atualização ao utilizador.
Fluxo Secundário: A. Falha de rede 1. A aplicação deteta indisponibilidade de ligação; 2. O sistema guarda a atualização localmente; 3. Quando a ligação é restabelecida, o sistema sincroniza automaticamente os dados com o backend; 4. O utilizador é informado da sincronização concluída (opcional).
Pós-condições: • O estado da tarefa é atualiza com sucesso no sistema; • A informação fica refletida na aplicação e na base de dados central.

TABELA 8 - CASO DE USO - APRESENTA ROTA ATUALIZADA

Caso de uso: Apresentar rota atualizada ID:UC5
Descrição: Receber e apresentar ao condutor, uma rota atualizada e otimizada, recalculada pelo sistema após a alteração do estado de uma tarefa ou ocorrência relevante, permitindo a continuação da navegação de forma eficiente.
Atores: Condutor (Motociclista)
Pré-condições: • O utilizador encontra-se autenticado; • Existe uma rota atribuída ao utilizador; • Existe uma rota previamente calculada; • O estado de pelo menos uma tarefa foi atualizado; • A ligação ao serviço FastAPI está disponível; • A ligação à API de mapas/navegação está disponível; • O dispositivo possui uma aplicação de navegação instalada.
Fluxo Primário: 1. O sistema deteta a atualização do estado de uma tarefa 2. A FastAPI solicita ao backend a nova rota otimizada; 3. O backend devolve a nova rota otimizada à FastAPI; 4. A FastAPI envia a nova rota para a aplicação; 5. A aplicação apresenta a rota atualizada; 6. O condutor seleciona a opção iniciar a navegação; 7. A aplicação abre a aplicação de navegação com a rota definida pelo sistema.
Fluxo Secundário: A. Falha ao aceder à aplicação de navegação 1. O sistema tenta abrir a aplicação de navegação; 2. A abertura falha; 3. O sistema apresenta uma mensagem de erro ao utilizador informando que não foi possível iniciar a navegação. B. Falha no serviço FastAPI 1. A aplicação não recebe resposta da FastAPI; 2. O sistema apresenta uma mensagem de erro ao utilizador; 3. O utilizador pode tentar novamente mais tarde.
Pós-condições: A rota otimizada fica definida.

TABELA 9 - CASO DE USO - DETALHE DA TAREFA

Caso de uso: Consultar detalhe da tarefa ID:UC6
Descrição: Apresentar os detalhes da tarefa selecionada na rota do dia, permitindo ao condutor consultar informação relevante para a execução da entrega (ex.: morada, contacto, notas e estado).
Atores: Condutor (Motociclista)
Pré-condições: • O utilizador encontra-se autenticado; • Existe uma rota atribuída ao utilizador para o dia em curso; • O dispositivo possui acesso à internet.
Fluxo Primário: 1. O condutor seleciona uma tarefa na rota do dia; 2. O sistema apresenta a lista de tarefas associadas à rota; 3. O condutor seleciona uma tarefa; 4. O sistema solicita ao backend (FastAPI) os detalhes da tarefa selecionada; 5. O backend devolve os dados da tarefa; 6. O sistema apresenta o ecrã de detalhe da tarefa.
Fluxo Secundário: A. Falha ao aceder detalhe da tarefa 1. O sistema apresenta uma mensagem de erro; 2. O utilizador pode tentar novamente ou regressar à lista.
Pós-condições: Os detalhes da tarefa selecionada são apresentados ao condutor.

TABELA 10- CASO DE USO - CONSULTAR HISTÓRICO DE TAREFAS

Caso de uso: Consultar histórico tarefa ID:UC7
Descrição: Apresentar ao utilizador a lista de tarefas atribuídas e permitir a consulta do histórico, com possibilidade de aplicar filtros (data, estado, entre outros).
Atores: Condutor (Motociclista)
Pré-condições: • O utilizador encontra-se autenticado; • Existe tarefas associadas ao utilizador (podem ser tarefas ativas, não concluídas e/ou concluídas); • Existe ligação à Internet • O serviço de tarefas (backend/BD) encontra-se acessível.
Fluxo Primário: 1. O utilizador acede ao ecrã Histórico; 2. O sistema solicita ao backend a lista de tarefas do utilizador; 3. O backend devolve as tarefas (ex.: por defeito, ordenadas por data/mais recentes); 4. O sistema apresenta a lista de tarefas no ecrã; 5. O utilizador seleciona uma tarefa; 6. O sistema apresenta os detalhes da tarefa selecionada.
Fluxo Secundário: A. Aplicar filtros (data, estado, etc.) 1. O utilizador seleciona a opção Filtrar; 2. O sistema apresenta os critérios de filtragem (ex.: data, estado, intervalo, tipo); 3. O utilizador define os filtros pretendidos; 4. O sistema aplica os filtros (via backend); 5. O sistema atualiza a lista com os resultados filtrados. B. Limpar filtros 1. O utilizador seleciona Limpar filtros; 2. O sistema repõe a listagem padrão (sem filtros). C. Não existem tarefas para os critérios selecionados 1. O sistema apresenta a mensagem “Sem tarefas para os filtros aplicados”; 2. O utilizador pode alterar/remover filtros. D. Falha de comunicação com o backend 1. O sistema informa que não foi possível obter as tarefas; 2. Se existir cache local, o sistema apresenta os últimos dados disponíveis; 3. O utilizador pode tentar novamente.
Pós-condições: As tarefas atribuídas ao utilizador ficam visíveis no ecrã Histórico; Caso existam filtros aplicados, a lista apresenta apenas os resultados correspondentes.

TABELA 11 - CASO DE USO - DETALHE TAREFA HISTÓRICO

Caso de uso: Consultar detalhe da tarefa no histórico ID:UC8
Descrição: Apresentar os detalhes da tarefa selecionada no histórico, permitindo ao condutor consultar informação.
Atores: Condutor (Motociclista)
Pré-condições: • O utilizador encontra-se autenticado; • Existe tarefas associadas ao utilizador (podem ser tarefas ativas, não concluídas e/ou concluídas); • Existe ligação à Internet; • O serviço de tarefas (backend/BD) encontra-se acessível.
Fluxo Primário: 1. O condutor seleciona histórico; 2. O sistema apresenta a lista de tarefas associadas; 3. O condutor seleciona uma tarefa; 4. O sistema solicita ao backend os detalhes da tarefa selecionada; 5. O backend devolve os dados da tarefa; 6. O sistema apresenta o ecrã de detalhe da tarefa.
Fluxo Secundário: B. Falha ao aceder detalhe da tarefa 1. O sistema apresenta uma mensagem de erro; 2. O utilizador pode tentar novamente ou regressar à lista.
Pós-condições: Os detalhes da tarefa selecionada são apresentados ao condutor.

Os casos de uso apresentados permitem caracterizar as principais interações entre o condutor e a aplicação, contribuindo para uma compreensão clara do comportamento esperado, servindo como base para o desenvolvimento da arquitetura, implementação das funcionalidades e validação dos requisitos.

III.4 Modelo da base de dados

No modelo de base de dados foram identificadas e modeladas as principais entidades do sistema, nomeadamente *users*, *motorcycles*, *tasks*, *services*, *alerts* e *routes*, bem como entidades associativas, tais como *user_motorcycles*, *user_alerts* e *motorcycle_alerts*, destinadas a suportar relações do tipo muitos-para-muitos (N:N). As ligações entre as tabelas foram definidas através de chaves primárias e estrangeiras, respeitando os princípios da normalização, com o objetivo de evitar redundância de dados e garantir a integridade referencial.

Este modelo permite a gestão completa dos utilizadores, das tarefas atribuídas, das motas associadas, das rotas planeadas, dos alertas e das operações de manutenção, incluindo ainda funcionalidades adicionais como notificações, registo de eventos (*logs*) e monitorização

de desempenho do sistema. Na **Figura 3** é apresentada a parte relevante do modelo entidade-relacionamento para o contexto do trabalho, representando a estrutura da base de dados desenvolvida para suportar todas as funcionalidades do sistema de gestão de motas elétricas, tendo como base os requisitos funcionais previamente definidos. O modelo entidade-relacionamento completo encontra-se em apêndice a esta dissertação.

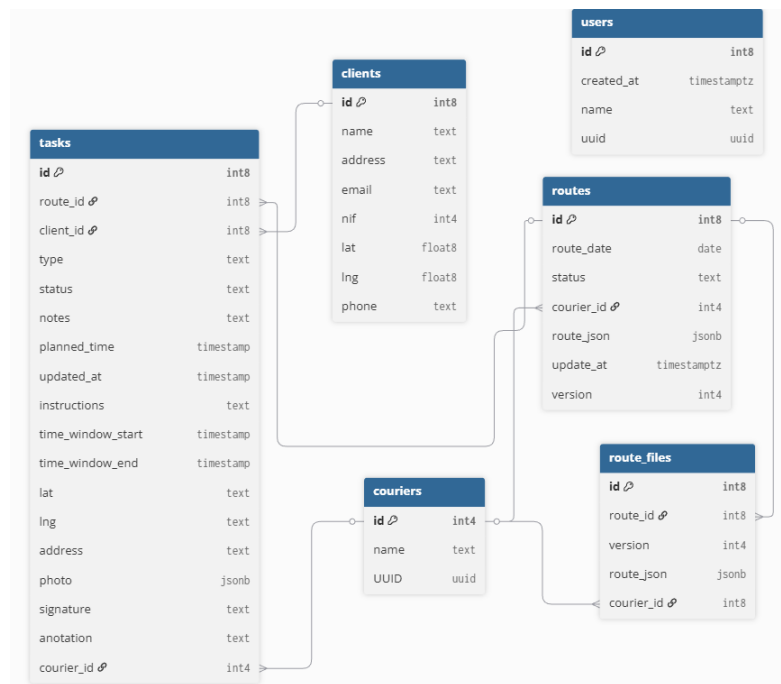


Figura 3 - Diagrama entidade-relacionamento base de dados

A modelação foi concebida de forma flexível e escalável, possibilitando futuras adaptações e extensões sem comprometer a estrutura existente ou a consistência dos dados. Adicionalmente, o modelo encontra-se alinhado com a integração futura de mecanismos de *caching*, filtros avançados de pesquisa e exportação de relatórios, assegurando a evolução sustentada da plataforma e a sua adequação a cenários de maior complexidade operacional.

III.5 Arquitetura

A definição da arquitetura do sistema constitui um elemento central no desenvolvimento da aplicação, uma vez que estabelece a organização estrutural dos componentes, os mecanismos de comunicação entre camadas e os princípios técnicos que orientam a evolução do *software*. Nos subcapítulos seguintes são apresentados o estudo arquitetural que fundamenta as decisões adotadas e a descrição detalhada da arquitetura

implementada, evidenciando a organização em camadas e a interação entre os diferentes componentes do sistema.

III.5.1 Estudo Arquitetura

A arquitetura de *software* tem como principal objetivo definir a estrutura e o comportamento de um sistema, assegurando simultaneamente a maximização de indicadores como o desempenho, a segurança e escalabilidade. Assim, constituindo um guia para o desenvolvimento e a evolução do *software*, contribuindo para a mitigação de riscos técnicos e organizacionais e para a utilização eficiente dos diversos componentes. A arquitetura orienta a estrutura do sistema, escolhe as tecnologias e *frameworks* mais adequadas e garante que as decisões técnicas se encontrem alinhadas com os objetivos do negócio (Nivedhaa, 2024).

As arquiteturas utilizadas em aplicações como Glovo e a Uber Eats constituem exemplos representativos de sistemas complexos concebidos para gerir operações de entrega em tempo real. No caso da Uber Eats, a arquitetura de *software* encontra-se organizada segundo um modelo em camadas, baseada em microserviços distribuídos, constituídos por um conjunto de subsistemas coordenados por meio de APIs e mecanismos orientados a eventos. Cada aplicação cliente comunica com subconjuntos específicos de serviços de *backend*, expondo diferentes funcionalidades consoante o perfil do utilizador, nomeadamente restaurantes, estafetas e clientes finais.

Um componente central desta arquitetura é a camada de *API Gateway/ Edge*, responsável por funcionalidades como a autenticação, o controlo de taxa por utilizador (*rate limiting*), para os serviços nucleares e o encaminhamento de pedidos (*routing*) para os serviços internos ou malha de serviços correspondente, garantindo assim segurança, escalabilidade e isolamento funcional (Thangavelu et al., 2024; K, A. A., 2025).

Por sua vez, a Glovo desenvolveu uma arquitetura escalável destinada a suportar funcionalidades avançadas, como conteúdos multimédia, componentes sociais e listas personalizadas na sua aplicação de entregas. Esta solução baseia-se num *design* modular orientado a *plugins*, na utilização de camadas do tipo *backend-for-frontend (BFF)* e em interfaces de utilizador orientadas por servidor (*server-driven UI*). A arquitetura modular denominada *Store Wall* encontra-se estruturada em serviços específicos, organizados em camadas de *backend* especializadas antes da exposição das APIs ao cliente final. Paralelamente, são implementados mecanismos de segurança e *edge services* como parte da

arquitetura de APIs públicas, recorrendo a práticas de *Infrastructure as Code* e a mecanismos de proteção como *Web Application Firewalls*(WAF) (Perelló & Malatini, 2024).

Embora os detalhes internos completos destas plataformas não sejam públicas, é possível concluir que ambas seguem um padrão de arquitetura moderno típico de grandes plataformas *on-demand*. Este padrão caracteriza-se por arquiteturas baseadas em microserviços e/ou serviços modulares, que permitem independência funcional, fácil escalabilidade e *deploy* contínuo; por abordagens *API-Driven* com recursos a *gateways* centrais, que fornecem um *interface* unificada aos clientes enquanto o *backend* gere a lógica de negócio através de serviços especializados; e por uma organização em camadas funcionais, incluindo *backend-for-frontend*, lógica de domínio, segurança e dados, que abstrai a complexidade do sistema e facilita a evolução independente dos seus componentes.

III.5.2 Arquitetura do sistema

A arquitetura proposta neste trabalho adota uma abordagem orientada a serviços, baseada em comunicações através de API REST e troca de dados, encontrando-se dividida em camadas independentes e bem definidas. Esta solução inspira-se nas arquiteturas modernas analisadas anteriormente, embora implementada numa escala inferior e ajustada ao contexto específico da mobilidade elétrica e da logística urbana, preservando, ainda assim, os princípios fundamentais de modularidades, separação de responsabilidades e facilidade de evolução do sistema.

A camada dos dispositivos corresponde aos *smartphones* Android utilizados pelos condutores, onde é executada a aplicação. A camada das aplicações integra a própria aplicação móvel, responsável pela interação com o utilizador, apresentação da informação, gestão de sessão e recolha de eventos operacionais. A camada dos serviços é constituída pela API desenvolvida em FastAPI, na alteração do estado da tarefa e na transferência de JSONs com rotas otimizadas, pela API automática da Supabase, para acesso e gestão de dados e pela API de navegação/mapa da aplicação instalada no dispositivo. Por fim, a camada de dados corresponde à base de dados PostgreSQL disponibilizada pelo Supabase, onde são armazenadas informações relativas a utilizadores, tarefas, rotas e histórico.

Embora a camada de integração esteja prevista e representada na arquitetura geral do sistema, no âmbito do presente trabalho optou-se pela realização de comunicações diretas

entre a aplicação e os serviços disponibilizados pelas APIs da camada de serviços. Esta decisão deveu-se sobretudo a critérios de simplicidade de implementação, redução da complexidade arquitetural e adequada à escala atual do projeto, bem como a decisão de utilizar o Supabase na camada de Base de dados.

Importa, contudo, salientar que, no contexto global do projeto A-Mover, está prevista a futura implementação de uma camada de integração dedicada, que permitirá centralizar a comunicação entre os diferentes serviços, para além da criação de uma base de dados própria, de utilização exclusiva no ambiente do projeto. Deste modo, a arquitetura desenvolvida neste trabalho pode ser entendida como uma etapa intermédia, funcional e alinhada com os objetivos do projeto, mantendo-se compatível com a posterior incorporação da referida camada de integração (**Figura 4**).

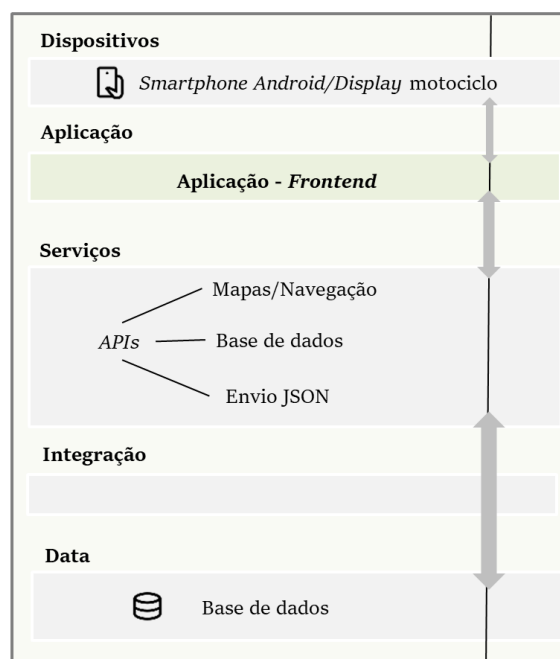


Figura 4 - Arquitetura do sistema

III.6 Mockups

Todos os ecrãs implementados na aplicação foram previamente concebidos sob a forma de *mockups*, com o objetivo de planejar, validar e otimizar os elementos da interface e os fluxos de navegação. Os componentes visuais necessários à conceção das diferentes interfaces foram definidos com base nos casos de uso identificados, permitindo assegurar a coerência funcional e a adequação às necessidades dos utilizadores.

Posteriormente, em colaboração com a equipa de *Front-End* da mota do projeto A-Mover, foram tomadas decisões relativas ao *design* gráfico, nomeadamente no que respeita à seleção de cores, disposição e dimensionamento dos elementos, estilos visuais e animações, procurando garantir uma experiência de utilização intuitiva, consistente e visualmente apelativa.

Numa fase subsequente, os ecrãs foram desenvolvidos recorrendo às ferramentas Figma e Glide. No Figma (ferramenta de design colaborativo) foram concebidos os layouts destinados tanto a smartphone como aos dispositivos de visualização integrados na mota elétrica, tendo em consideração as especificidades de cada contexto de utilização. Por sua vez, o Glide (plataforma *no-code*) foi utilizado para a criação de protótipos navegáveis, possibilitando a análise entre os diferentes ecrãs e a validação dos fluxos de interação.

A autenticação é efetuada através da inserção do *username* e da *password* do utilizador. Após a autenticação, a aplicação apresenta ao utilizador o menu principal, correspondente ao layout inicial (*Home*), no qual é disponibilizada as principais funcionalidades do sistema, nomeadamente, ao Histórico de atividade e à opção de encerramento de sessão (*logout*) (**Figura 5**).

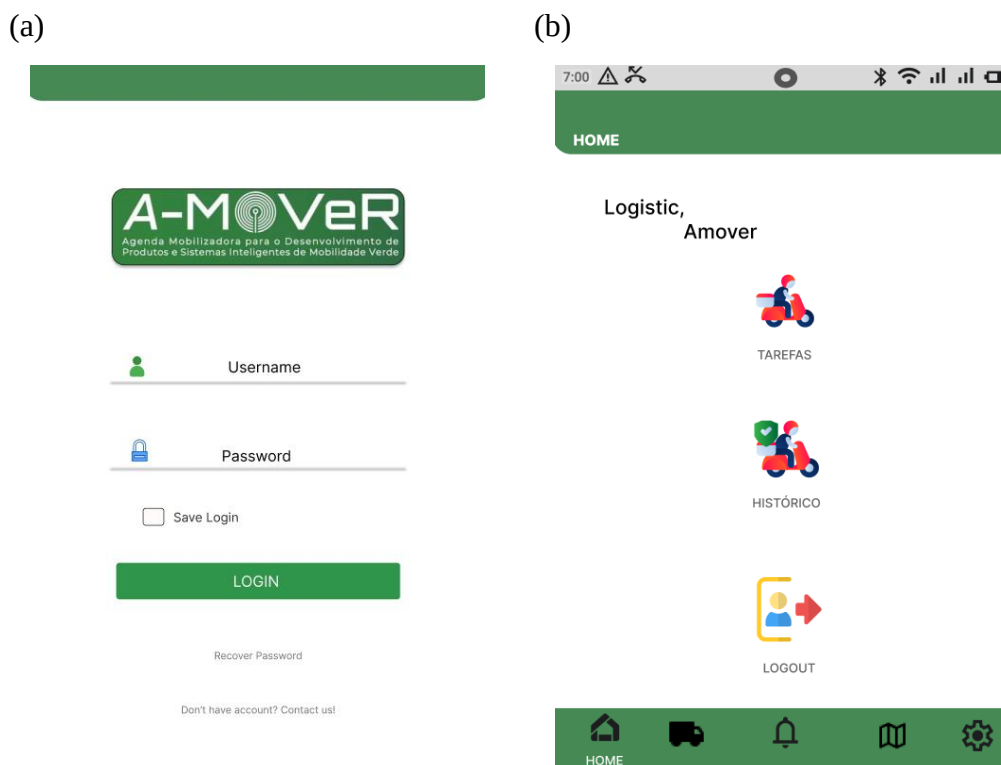


Figura 5 - Mockups (a) ecrã de autenticação e (b) Home

As interfaces apresentadas ao utilizador aquando da seleção das opções *Tarefas* ou *Histórico* exibem a lista das tarefas atribuídas. No ecrã *Tarefas* é apresentada a rota correspondente ao dia atual, contendo apenas tarefas agendadas para esta data. Por sua vez, o ecrã *Histórico* disponibiliza o conjunto completo de todas as tarefas associadas ao utilizador, independente da data de execução (**Figura 6**).

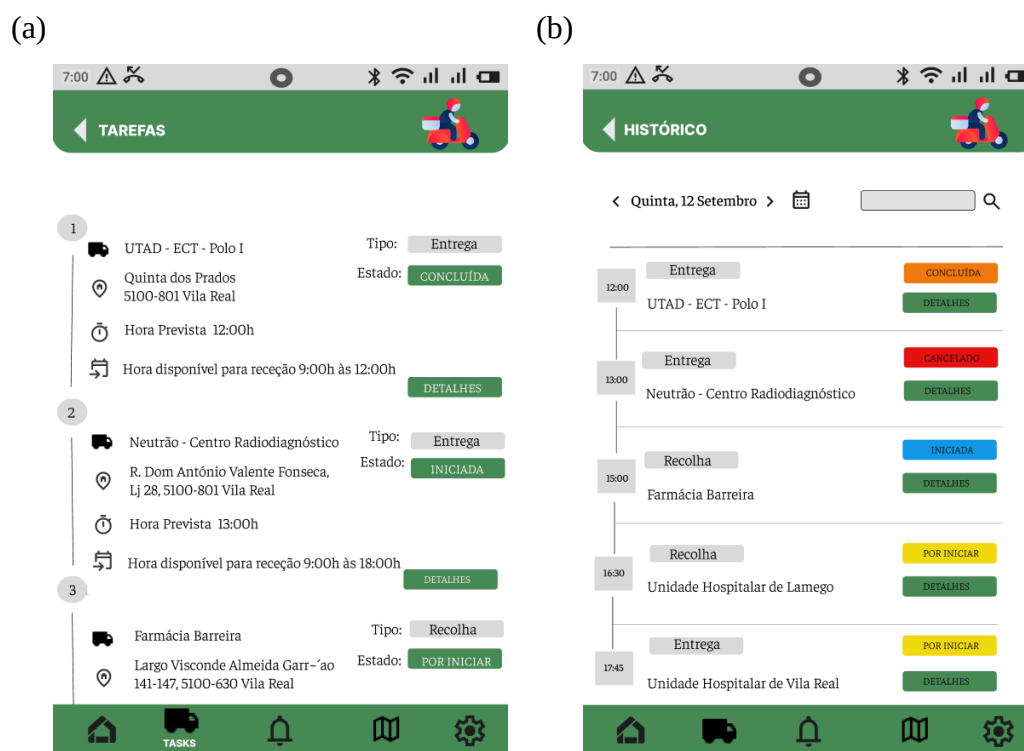


Figura 6 - Mockups das (a) tarefas e do (b) histórico

O ecrã apresentado aquando da seleção de uma tarefa corresponde ao detalhe da mesma, no qual são exibidas todas as informações relevantes para a sua execução, incluindo a morada de destino, a hora prevista de entrega, o tipo de encomenda, a tipologia da tarefa (entrega ou recolha) e o respetivo estado. Adicionalmente, ao selecionar a opção comprovativo, é apresentado um ecrã dedicado à sua visualização. Em Comprovativo são apresentadas as diferentes formas de confirmação da entrega, de acordo com a política definida pela organização. Em alternativa, o utilizador pode registar, no campo relatório/observações, o motivo pelo qual a tarefa não foi concluída, assegurando assim a documentação adequada da ocorrência (**Figura 7**).

No ecrã de detalhe da tarefa é disponibilizada a opção de iniciar a respetiva tarefa. Ao selecionar estado iniciar, concluir ou cancelar (**Figura 7**).

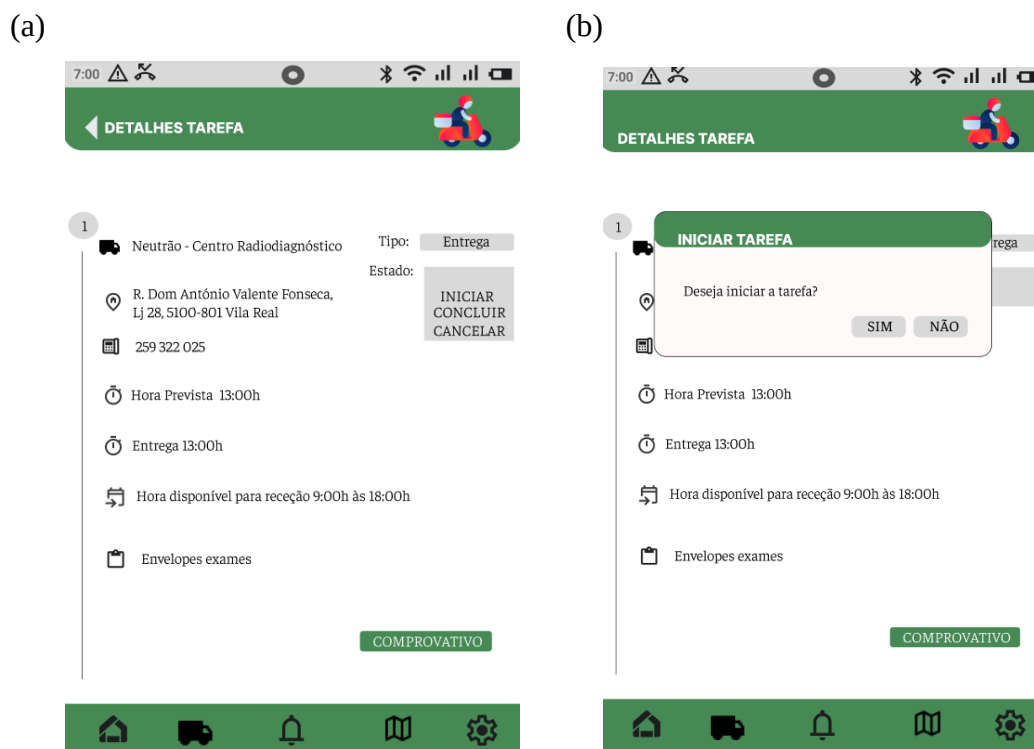


Figura 7 - Mockup de detalhe da tarefa

Durante a navegação, a aplicação apresenta um ecrã dedicado à visualização da rota até à morada de destino para cada tarefa seleccionada, conduzindo o utilizador por cada percurso, uma viagem de cada vez (Figura 8).

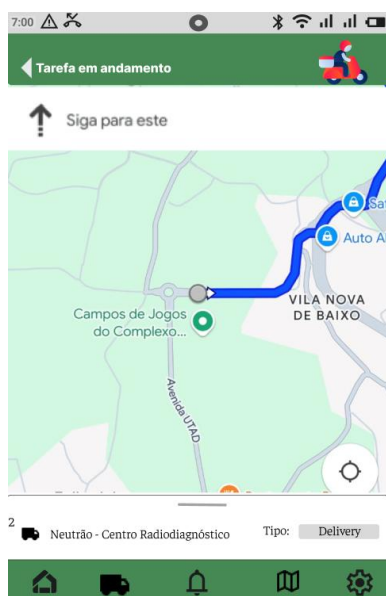


Figura 8 - Mockup navegação

Foi igualmente concebido uma interface destinada à visualização da rota no display da moto durante a condução. Este *layout* foi desenvolvido com vista a uma futura integração no âmbito da evolução do projeto A-Mover e, por esse motivo, não foi implementado na presente fase do trabalho (**Figura 9**).



Figura 9 - Mockup navegação display do motociclo

Os *mockups* desenvolvidos serviram de base ao design e às interações da *frontend* final, assegurando uma organização clara das funcionalidades essenciais e uma experiência de utilização simples e eficiente para o condutor.

III.7 Problemas de roteamento de veículos

Neste trabalho vários algoritmos foram explorados para a resolução do Problema de Roteamento de Veículos (VRP) (Chen, 2025), uma generalização do problema do caixeiro-viajante (Travelling Salesman Problem – TSP) (Hexaly, n.d.), especificamente adaptados ao contexto urbano. Foram considerados diversos cenários relevantes, tais como o Problema de Roteamento de Veículos com Capacidade (Capacitated Vehicle Routing Problem – CVRP) (Hexaly, n.d.), o Problema de Roteamento de Veículos com Janelas Temporais (Vehicle Routing Problem with Time Windows – VRPTW) (Google OR-Tools, 2024), o Problema de Roteamento de Veículos com Múltiplos Depósitos (Multi Depot Vehicle Routing Problem – MDVRP) (Hexaly, 2024) e o Problema de Roteamento Dinâmico de Veículos (Dynamic Vehicle Routing Problem – DVRP) (Kucharska, 2019), no qual podem surgir novos pedidos durante a execução das entregas (Fernandes et al., 2025).

O sistema desenvolvido permitiu simular diferentes situações operacionais e comparar o desempenho de algoritmos exatos, como o Dijkstra (Patel, 2023) e o Branch and Bound (Lawler & Wood, 1966), heurísticos, como Savings (Toklu, 2022) e o Nearest Neighbor (Sadhukhan et al., 2025) e, por fim, os meta-heurísticos, como o GRASP (Greedy Randomized Adaptive Search Procedure) (Feo & Resende, 1995) e o Tabu Search (Tucson et al., 1999).

O algoritmo de Dijkstra baseia-se no princípio de um algoritmo ganancioso, selecionando em cada etapa o nó com a menor distância estimada a partir da origem. O processo inicia-se com a escolha de um nó inicial e o cálculo das distâncias mínimas para os restantes nós da rede. Iterativamente, o nó mais próximo é fixado e utilizado para atualizar as distâncias dos restantes, sendo posteriormente excluído de novas iterações. Este procedimento repete-se até que todas as distâncias mínimas sejam determinadas, resultando na identificação das rotas mais curtas entre a origem e os demais nós (Vigário et al., 2025; Fernandes et al., 2025).

O algoritmo Branch and Bound é utilizado para resolver instâncias de menor dimensão do TSP, VRP e CVRP. Este método exato explora sistematicamente todas as rotas possíveis, dividindo o problema em subproblemas e eliminando aqueles que não podem produzir soluções melhores do que a melhor solução conhecida. O Branch and Bound garante a obtenção da solução ótima, mas apresenta um elevado custo computacional, tornando-se impraticável para instâncias de grande dimensão (Vigário et al., 2025; Fernandes et al., 2025).

O Algoritmo de Poupança de Clarke-Wright Savings é aplicado na resolução de instâncias maiores do VRP, como o CVRP. Este método heurístico calcula as poupanças resultantes da combinação de diferentes rotas de entrega e procede à sua fusão de forma iterativa, de modo a criar rotas mais eficientes. Apesar de ser rápido e computacionalmente eficiente, o algoritmo de poupança não garante o ótimo das soluções, sobretudo quando enfrenta restrições complexas, como o recarregamento de baterias (Vigário et al., 2025b; Fernandes et al., 2025).

A heurística do Nearest Neighbor é um método eficiente, especialmente aplicado para TSP. O algoritmo parte de um ponto inicial (sede ou depósito) e, em cada etapa, escolhe o vértice mais próximo do último ponto visitado, adicionando-o à rota. Este processo repete-se até que todos os vértices tenham sido visitados, concluindo com o retorno ao ponto de origem.

Visto ser semelhante ao algoritmo Savings, este apenas encontra soluções perto do ótimo e é mais lento. No entanto, em comparação com os algoritmos dos métodos exatos, este é bastante mais simples (Vigário et al., 2025b; Fernandes et al.,2025).

A Tabu Search é aplicada a problemas como o DVRP, nos quais as rotas necessitam de ser ajustadas dinamicamente, utilizando uma estrutura de memória, designada por lista tabu, que impede o algoritmo de regressar a soluções recentemente exploradas, promovendo assim uma exploração mais eficaz do espaço de soluções. Por sua vez, o GRASP é utilizado no problema VRPTW, onde é necessário respeitar restrições de janelas temporais e de carregamento, construindo soluções iniciais através de uma abordagem gulosa e aleatória, seguida de uma fase de busca local destinada a melhorar progressivamente a qualidade das soluções obtidas (Vigário et al., 2025b; Fernandes et al.,2025).

III.8 Resumo do capítulo

Este capítulo apresentou o problema abordado no âmbito do projeto A-Mover e a solução proposta para apoiar a navegação e a gestão de tarefas em serviços de logística com motociclos elétricos. Foram definidos os requisitos funcionais e não funcionais do sistema, bem como os principais casos de uso, permitindo estruturar o comportamento esperado da aplicação.

Apresentaram-se ainda o modelo da base de dados e a arquitetura do sistema, organizada em camadas e baseada em serviços. A conceção da interface foi realizada através de *mockups* e protótipos desenvolvidos em Figma e Glide, com enfoque na usabilidade e na redução da carga cognitiva do condutor, incluindo a integração com o Google Maps para navegação, bem como levantamento dos algoritmos para a resolução do problema de roteamento de veículos. No próximo capítulo é discutido o desenvolvimento e a implementação da aplicação.

IV. IMPLEMENTAÇÃO DA APLICAÇÃO

Após a consolidação dos mockups, e a definição dos requisitos e casos de uso, damos início à fase de implementação do protótipo. Assim, nesta secção são descritas as tecnologias utilizadas, apresentadas as decisões relativas ao *design* da interface e as principais funcionalidades implementadas. É feita uma abordagem do processo de desenvolvimento do protótipo, bem como das alterações de algumas das funcionalidades. A presente dissertação engloba apenas os serviços e bases de dados para a sua execução, uma vez que estes componentes serão posteriormente implementadas e integradas no âmbito do projeto A-Mover, com o objetivo de criar um ecossistema unificado, assente em uma base de dados única, API dedicada à execução dos algoritmos de otimização de rotas, autenticação dos utilizadores e a gestão de permissões asseguradas através da integração com o sistema de código aberto Keycloak e integração com o display do motociclo.

O capítulo está dividido nas seguintes secções: a secção IV.1 processo e metodologia de desenvolvimento, a secção IV.2 apresenta as tecnologias utilizadas, a secção IV.3 aborda a os testes efetuados na otimização de rotas. A secção VI.4 de uma base de dados, simulando o contexto de uma empresa de logística e na secção seguinte, o desenvolvimento de uma API simples, para a interoperabilidade de ficheiros JSON e alteração do estado da tarefa na base de dados, por fim, a secção IV.6 apresenta o processo de desenvolvimento do sistema.

IV. 1 Metodologia e processo de desenvolvimento

O desenvolvimento de sistemas de *software* envolve um conjunto estruturado de processos que visam a entrega de uma solução final alinhada com as necessidades dos utilizadores. É comum que, ao longo do desenvolvimento, surja a necessidade de introduzir novas funcionalidades ou de ajustar as já existentes de acordo, pois muitas vezes o cliente não consegue identificar claramente as suas necessidades.

No decorrer deste trabalho, cujo foco principal é o desenvolvimento de uma frontend, foram adotados processos de desenvolvimento de *software* ágeis. Os processos de *software* ágeis permitem um desenvolvimento incremental e facilita a adaptação contínua dos requisitos de acordo com as necessidades do cliente. Entre as principais metodologias associadas aos processos ágeis, destacam-se a Extreme Programming (XP), Scrum ou Feature Driven Development (FDD) (Livermore, 2007).

O método de desenvolvimento utilizado neste trabalho não segue em rigor nenhuma das metodologias ágeis, e é uma adaptação das metodologias XP, que é caracterizada pela incorporação de mudanças de requisitos em qualquer fase de desenvolvimento, e FDD que é caracterizada pela organização do desenvolvimento por funcionalidades. Na **Figura 10** é apresentado o processo de desenvolvimento utilizado.

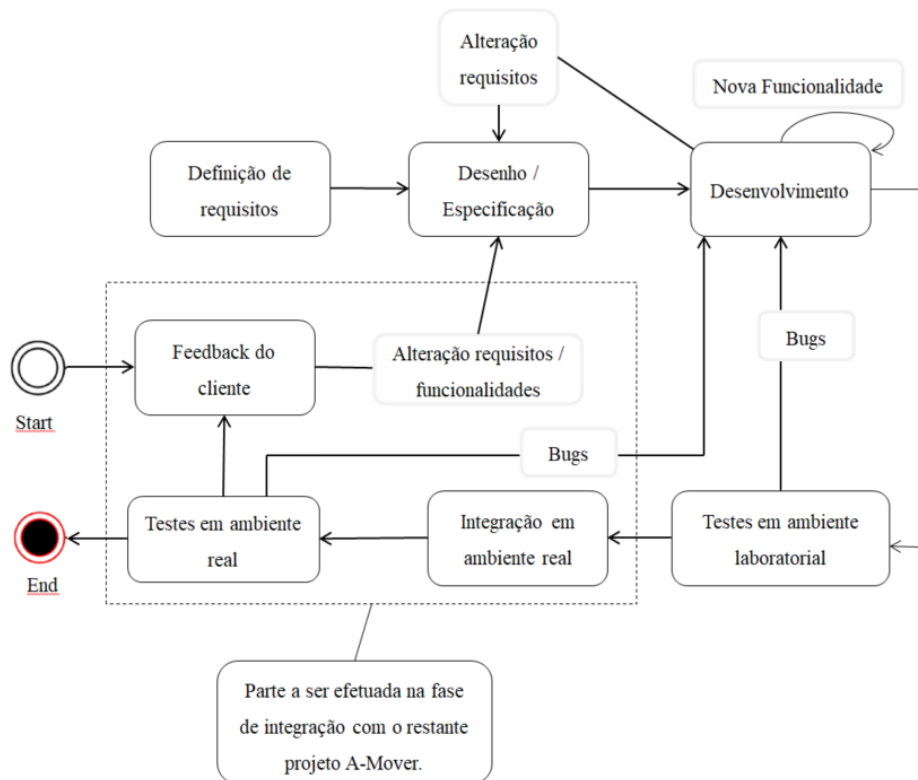


Figura 10 - Metodologia de desenvolvimento

De acordo a **Figura 10**, o processo inicia-se com a definição dos requisitos, seguido do desenho e especificações desses requisitos. De seguida, ocorre o processo de desenvolvimento das funcionalidades identificadas. Após o desenvolvimento de uma funcionalidade, são realizados testes em ambiente laboratorial. Caso ocorra erros (*bugs*), a funcionalidade retorna ao processo de desenvolvimentos. No âmbito desta dissertação, foi realizado, até o momento, este ciclo.

A integração da solução no ambiente real do Projeto A-Mover, bem como a recolha de feedback do cliente sobre eventuais alterações necessárias, será realizada em fase posterior. Sempre que ocorram alterações de requisitos e funcionalidades, o processo é reiniciado a partir da fase de *desenho/especificações*.

IV. 2 Tecnologia utilizadas

Tendo em conta as especificações descritas no capítulo III, arquitetura da solução, é necessário definir as tecnologias a serem utilizadas. A **Figura 11** apresenta as tecnologias utilizadas em cada camada da arquitetura desenvolvida que será descrita em seguida.

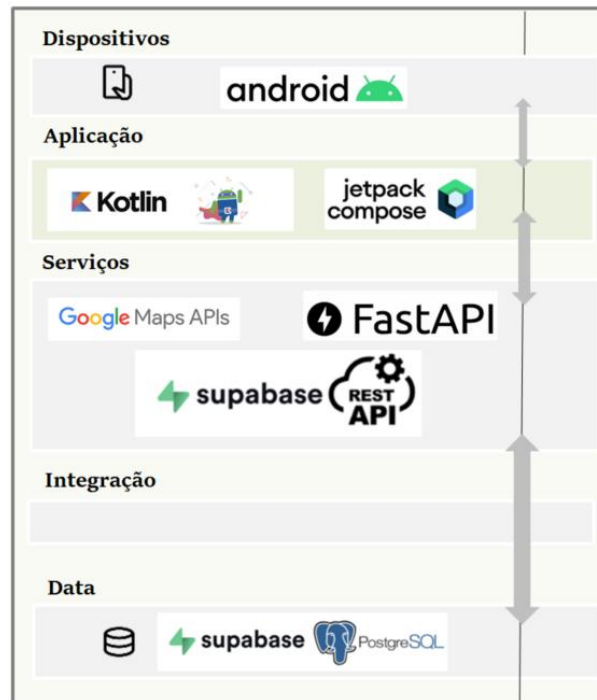


Figura 11 - Arquitetura: tecnologias utilizadas

Para cada camada foi selecionada uma ferramenta, após estudos e comparações efetuadas, tendo por base a arquitetura geral definida e a natureza do projeto.

IV.2.1 Android

Android é um sistema operativo móvel, baseado no núcleo Linux e desenvolvido pela Google, responsável por gerir e executar todos os recursos de *hardware* e *software* em um dispositivo móvel com ecrã tátil, como *smartphones* e *tablets*. Este sistema operativo *open-source* suporta um vasto ecossistema de aplicações e serviços, sendo utilizado em dispositivos móveis a nível global (Android, 2025).

Atendendo à opção pelo desenvolvimento da aplicação para a plataforma Android, o Android Studio foi utilizado como ambiente de desenvolvimento integrado (*Integrated Development Enviroment* - IDE) principal, para a implementação da aplicação móvel (Android Developers, 2025). Esta plataforma oficial do Google fornece um conjunto de

ferramentas para programação, depuração, testes e análise de desempenho, permitindo um ciclo de desenvolvimento eficiente e controlado. Entre as suas principais vantagens destacam-se o emulador, o sistema de gestão de dependências Gradle, apoio de tecnologia de inteligência artificial, Gemini, e o suporte nativo às bibliotecas modernas do ecossistema Android.

IV.2.2 Kotlin

Kotlin é um projeto de código aberto, distribuído sob a licença Apache 2.0, desenvolvido pela JetBrains. Trata-se de uma linguagem que executa sobre a Máquina Virtual Java (JVM), com linguagem de forma estática e suporte à programação funcional. Apresenta uma sintaxe mais expressiva e concisa quando comparada com a linguagem Java. A linguagem incorpora conceitos da programação funcional, como expressões *lambda*, e, por ser multi-paradigma, permite ao programador adotar tanto o paradigma funcional como o paradigma orientado a objetos no desenvolvimento de aplicações. O Kotlin é utilizado em diversos contextos, como no desenvolvimento Android, *web* e *backend*, em aplicações *desktop* e ciências de dados, enquanto linguagem multiplataforma, possibilita a partilha de código entre diferentes plataformas, como iOS e web, promovendo a reutilização e a eficiência no desenvolvimento de *software* (JetBrains, 2020).

IV.2.3 Jetpack Compose

O Jetpack Compose é um *kit* de ferramentas moderno desenvolvido pela Google com o objetivo de facilitar o desenvolvimento de interfaces de utilizador em aplicações Android. Este *framework* baseia-se num modelo de programação declarativa, no qual a *interface* é descrita através de funções que transformam dados numa hierarquia de componentes visuais. Sempre que os dados sofrem alterações, o sistema é responsável por atualizar automaticamente a *interface*, garantindo consistência e reatividade.

O *compose* foi concebido para ser utilizado com a linguagem de programação Kotlin, tirando partido da sua sintaxe. A sua base assenta nas denominadas funções *composable*, que correspondem a funções normais anotadas com `@Composable`, capazes de invocar outras funções do mesmo tipo, promovendo a criação de componentes reutilizáveis e modulares. Para além disso, o *framework* simplifica a gestão do estado da *interface*, permitindo o desenvolvimento de aplicações que reagem de forma eficiente às alterações dos dados.

Adicionalmente, o Jetpack Compose oferece integração nativa com os princípios do Material Design, disponibilizando componentes e temas pré-definidos que facilitam a criação de *interfaces* visualmente consistentes, atrativas e funcionais. A sua estrutura modular contribui ainda para um processo de desenvolvimento mais rápido e intuitivo, favorecendo a reutilização de código e a manutenção da aplicação ao longo do tempo.

Em síntese, o Jetpack Compose constitui uma solução moderna e robusta para o desenvolvimento de *interfaces* gráficas em Android, permitindo a criação de aplicações mais eficientes e com uma melhor experiência de utilização (Android Developers, 2026).

IV.2.4 API Google Maps

A API do Google Maps é uma *interface* de programação que permite os desenvolvedores integrar mapas, serviços de geolocalização, rotas e informações de locais em aplicações. Esta funciona por meio de requisições HTTP que retornam dados em Json ou XML, utilizando bibliotecas JavaScript para manipulação dos mapas. No contexto do Android, a integração com o Maps SDK for Android permite criar *interfaces* de mapas de forma declarativa, utilizando o Jetpack Compose. Esta biblioteca disponibiliza diversas funções e tipos de dados que facilitam a implementação de funcionalidades comuns, tais com a apresentação de marcadores, traçado de rota, visualização da posição atual do utilizador, cálculo de distâncias e interação com o mapa em tempo real (Google for Developers, 2006).

IV.2.5 FastAPI

A API local foi desenvolvida com o objetivo principal de assegurar a comunicação entre a aplicação móvel e o sistema *backend*, nomeadamente para o envio e atualização de rotas em formato JSON, bem como para a sincronização de estados e registos na base de dados. Esta abordagem permitiu testar e validar o funcionamento da aplicação de forma autónoma durante a fase de desenvolvimento. Adicionalmente, a arquitetura da API foi concebida de forma modular e escalável, considerando a futura integração com uma API generalizada no âmbito do projeto A-MoVer, possibilitando a expansão para um ecossistema mais amplo de serviços de mobilidade e logística urbana.

A escolha da *framework* baseou-se em fatores como desempenho, segurança e escalabilidade. Na **Tabela 12** é apresentado as *frameworks* estudadas.

TABELA 12 - COMPARAÇÃO DAS FRAMEWORKS ESTUDADAS

Framework	Performance	Facilidade de utilização	Escalabilidade	Suporte Assíncrono	Documentação Automática	Segurança	Recomendação Python
FastAPI	Excelente	Excelente	Bom	Excelente	Excelente	Excelente	√
Flask	Bom	Excelente	Bom	Via extensões	Via Swagger	Bom	√
Django Rest Framework (DRF)	Bom	Bom	Excelente	Via extensões	Excelente	Excelente	√
Express.js	Bom	Excelente	Excelente	Excelente	Via Swagger	Bom	X
Spring Boot	Excelente	Bom	Excelente	Excelente	Excelente	Excelente	X

Legenda:

Performance: Mede a eficiência na resposta aos pedidos.

Facilidade de uso: Avalia a curva de aprendizagem e simplicidade de implementação.

Escalabilidade: Indica a capacidade de lidar com o aumento do número de pedidos.

Suporte assíncrono: Determina se a framework suporta operações assíncronas de forma nativa.

Documentação automática: Indica se a framework oferece documentação interativa integrada.

Segurança: Avalia os recursos nativos para autenticação e proteção contra vulnerabilidades.

Recomendado para python: Indica se a framework é adequada para uso com Python.

Após a realização do estudo comparativo das diferentes tecnologias disponíveis, apresentado na **Tabela 12** e, considerando a experiência prévia com a linguagem de programação Python, foi selecionado o FastAPI como *framework* para o desenvolvimento da API do sistema. Esta escolha fundamentou-se não apenas na familiaridade com a linguagem, mas também nas vantagens técnicas do FastAPI, nomeadamente o elevado desempenho, o suporte nativo a operações assíncronas, a geração automática de documentação interativa e a sua adequação ao desenvolvimento de serviços escaláveis e modernos.

IV.2.6 Supabase

O *Supabase* é uma plataforma *open source* que fornece uma solução completa de *backend* baseada em PostgreSQL totalmente gerida na *cloud*. Para além da base de dados, inclui autenticação, armazenamento de ficheiros, funções (*serverless*) e APIs automáticas, permitindo que equipas de desenvolvimento criem aplicações modernas com facilidade (Supabase, 2025).

Destaca-se pela simplicidade de utilização, integração transparente com PostgreSQL, painel de gestão *web* muito intuitivo e funcionalidades como *backups* automáticos e gestão de permissões. É especialmente indicada para prototipagem rápida, projetos académicos, MVPs e até projetos em produção com necessidades moderadas, considerando a futura integração com uma base de dados generalizada no âmbito do projeto **A-MoVer**.

Nesta secção foi apresentado todas as ferramentas desenvolvidas na criação deste trabalho, dando seguimento na próxima secção ao estudo dos algoritmos de otimização.

IV. 4 Base de dados

Como referido anteriormente, a base de dados foi criada na ferramenta *online* Supabase. Tratando-se de uma base de dados PostgreSQL, isto é, relacional, organizada em tabelas com uma estrutura bem definida, utilizando linguagem SQL, suportando relacionamentos entre entidades, bem como chaves primárias e estrangeiras, garantindo uma forte consistência e integridade dos dados. Como referido anteriormente, foi criado uma base de dados funcional para a realização desta dissertação, com o objetivo de representar, de forma realista, os requisitos necessários ao desenvolvimento do sistema proposto, e que já foi apresentados nos diagramas de entidade e relacionamento.

As tabelas foram criadas para simular o funcionamento de uma empresa real de entregas e recolhas. Entre estas, destacam-se as tabelas ***route_files*** e a ***routes*** apresentadas na **Figura 12**.

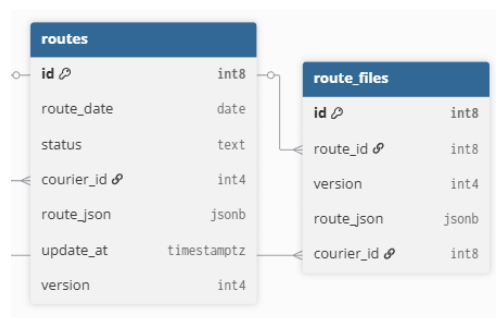


Figura 12 - Tabelas *routes* e *routes_files*

A principal função da tabela ***routes_files*** é armazenar ficheiros no formato JSON, contendo rotas otimizadas provenientes de uma API externa. Embora, no contexto deste trabalho, esta inserção seja realizada manualmente, a estrutura encontra-se preparada para uma futura implementação, através da integração com o ecossistema A-Mover. Estes ficheiros são seleccionados de acordo com a sua versão sempre que a aplicação solicita uma nova rota, sendo posteriormente o ficheiro JSON ativo associado à tabela ***routes***, substituindo o anterior.

IV. 5 API

A API primária, desenvolvida em FastAPI, foi criada como base para o desenvolvimento da aplicação. Implementada em Python, esta API é responsável por intermediar a comunicação entre a aplicação móvel e a base de dados, Supabase, bem como suportar operações do sistema, como apresentado no código da **Figura 13**. Em particular, permite a alteração do estado das tarefas (por exemplo, não concluída ou concluída) e a transferência de JSON contendo rotas otimizadas, como enxerto apresentado na **Figura 14**. Desta forma, a API assegura a atualização consistente da informação operacional e a disponibilização atempada dos percursos aos condutores.

```
# --- ENV / Supabase -----
load_dotenv()

SUPABASE_URL = os.getenv("SUPABASE_URL")
SUPABASE_KEY = os.getenv("SUPABASE_ANON_KEY")

if not SUPABASE_URL or not SUPABASE_KEY:
    raise RuntimeError("SUPABASE_URL ou SUPABASE_ANON_KEY não definidos no .env")

supabase: Client = create_client(SUPABASE_URL, SUPABASE_KEY)

# --- FastAPI -----

app = FastAPI(title="Amover")

app.add_middleware(
    CORSMiddleware,
    allow_origins=["*"],      # em produção deve restringir
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)
```

Figura 13 - Parte código-fonte da API

A **Figura 13** apresenta parte do código-fonte da API desenvolvida em FastAPI, responsável pela configuração inicial do serviço e pela ligação à base de dados Supabase (PostgreSQL). A API não é iniciada sem uma configuração válida, aumentando a segurança e robustez do sistema. É criado o cliente Supabase, que permite à API aceder à base de dados PostgreSQL, consultar e atualizar tabelas e gerir tarefas, rotas e ficheiros JSON, funcionando como ponte entre a API FastAPI e o *backend* de dados. A seguir, configura-se o **middleware** que permite que a API seja acedida pela aplicação Android, mesmo estando em domínios diferentes, que em produção devem ser restringida por motivos de segurança.

```

{
  "nodes": [
    {
      "x": -7.740830797169679,
      "y": 41.28684669406769,
      "id": 0,
      "is_depot": true
    },
    {
      "x": -7.735645509599732,
      "y": 41.29310917773415,
      "id": 1,
      "type": "delivery",
      "notes": "envelopes",
      "status": "pending",
      "id_task": 1,
      "client_id": "3",
      "name_client": "Utad - Universidade de Trás-os-Montes e Alto Douro",
      "address_task": "Lidl, Av. da Universidade 6, 5000-693 Vila Real",
      "instructions": "Entrega gerente loja",
      "time_window_end": "18:00:00",
      "time_window_start": "16:00:00"
    }
  ],
}

```

Figura 14 - Parte ficheiro JSON

A **Figura 14** apresenta parte do ficheiro JSON utilizado pelo sistema para representar uma rota otimizada e as tarefas associadas. Estruturado com um conjunto de nós geográficos, incluindo o depósito logístico e as tarefas de entregas associados, cada nó integra informação espacial, operacional e temporal, transferidas através da API FastAPI, permitindo à aplicação Android desenhar a rota no mapa, apresentar rotas do dia, apresentar detalhes da tarefa da rota do dia, ordenar os pontos de visita e validar as restrições temporais.

IV. 6 Desenvolvimento da aplicação

Tendo sido feito a especificação dos requisitos, a identificação das tecnologias a serem utilizadas, os testes aos algoritmos de otimização e a base de dados, começa a etapa de desenvolvimento do sistema. Nesta secção são descritas, em pormenor, as *interfaces*.

IV.6.1 Interface Login

O serviço de autenticação da aplicação foi implementado através do componente **SupabaseClientProvider**, o qual segue o padrão de projeto **Singleton**, sendo responsável por criar e fornecer uma única instância única do **SupabaseClient** em toda a aplicação. Este cliente é utilizado para: realizar operações de autenticação (login/logout) e aceder à base de dados PostgreSQL através da *interface* disponibilizada pelo Supabase.

Por razões de segurança, as credenciais de acesso (URL do projeto e chave pública) não se encontram codificadas diretamente no código-fonte. Estas são carregadas

dinamicamente a partir do ficheiro de configuração BuildConfig, definido no ambiente Gradle, evitando a exposição de informação sensível no repositório da aplicação.

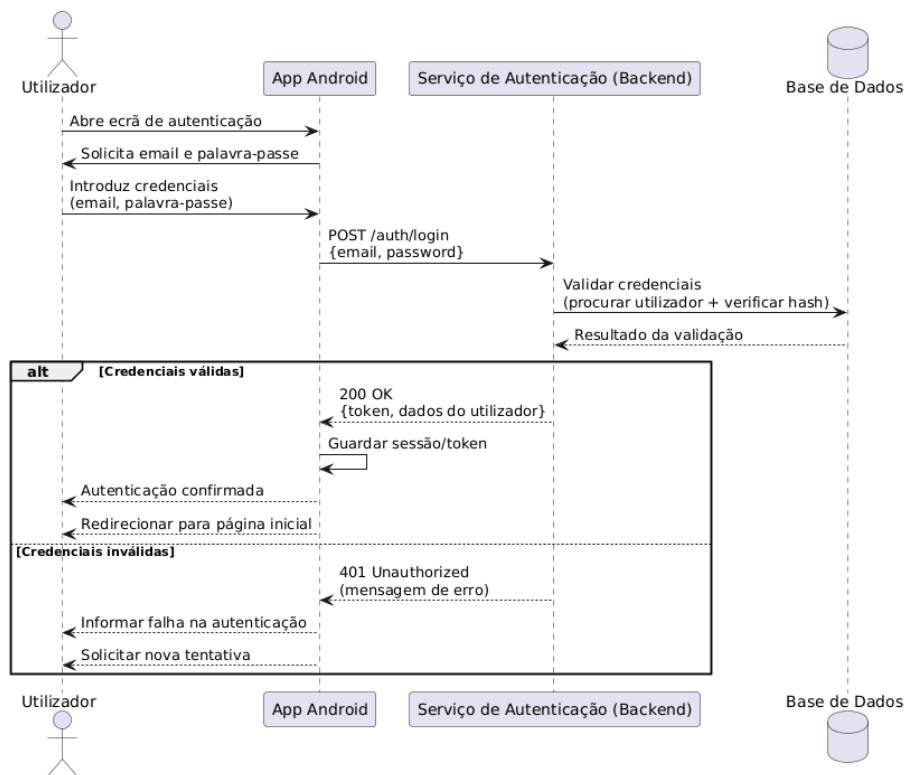


Figura 15 - Diagrama de sequência autenticar utilizador

O processo de autenticação, demonstrado através do diagrama de sequência (**Figura 15**), o processo inicia-se quando o utilizador introduz as suas credenciais na *interface* da aplicação. Estes dados são enviados de forma segura para o serviço de autenticação do Supabase. O Supabase valida as credenciais e, em caso de sucesso, devolve o *token* de sessão associado ao utilizador, a aplicação armazena esta sessão localmente, sendo por fim redirecionado para a página inicial, Home. Caso as credenciais sejam inválidas ou ocorra uma falha de comunicação, é devolvida uma mensagem de erro, sendo o utilizador notificado pela *interface*. Este fluxo possui uma abordagem segura e eficiente para autenticar utilizadores, autorizar o acesso a recursos protegidos e garantir a privacidade das informações.

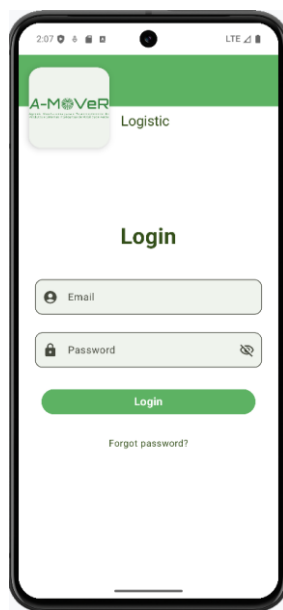


Figura 16 - Interface de autenticação da aplicação

A **Figura 16** apresenta a *interface* da autenticação, na qual o utilizador deve introduzir as suas credenciais para autenticar-se e aceder a aplicação. Após a validação dos dados, o sistema permite o acesso às restantes funcionalidades da aplicação.

IV.6.2 Interface Consultar Rotas e Tarefas

Para que o utilizador consultar a rota do dia, como demonstrado no diagrama de sequência (**Figura 17**), depois de autenticado, selecionando a opção de tarefas (Task), a aplicação envia um pedido (*request*) à *API backend* para obter a rota diária e as tarefas associadas. Ao receber o pedido, o *backend*, por sua vez, procede à consulta da base de dados de forma a verificar se existe uma rota atribuída ao condutor para a data em questão. Caso exista uma rota registada, a base de dados devolve à API a informação correspondente, através do ficheiro JSON.

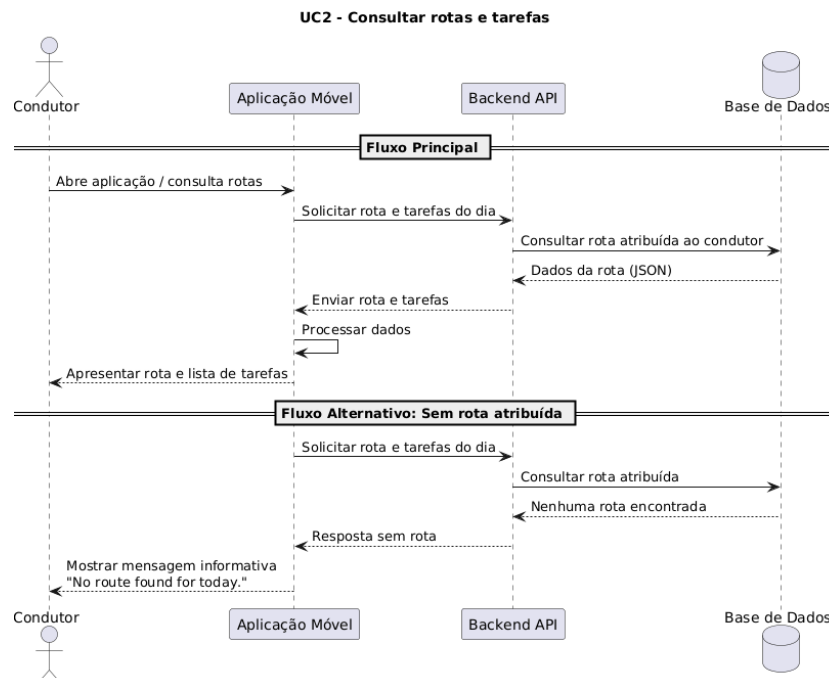


Figura 17 - Diagrama de sequência consultar rota e tarefas

De seguida, o *backend* envia o ficheiro JSON à aplicação. Após receção, a aplicação apresenta em seu *layout*, de forma organizada, a rota com as tarefas associadas. Caso não exista rota associada ao utilizador, em fluxo alternativo, é apresentado a mensagem “*No route found for today*”.



Figura 18 - Interface rota do dia

A **Figura 18** apresenta a *interface* da visualização da rota diária associada ao utilizador.

IV.6.3 Interface rota/navegação

Depois da interação descrita anteriormente, na *interface* Tasks, ao selecionar o botão “*Start navigation*”, a aplicação solicita a abertura da aplicação externa de navegação (Google Maps), recorrendo a um mecanismo de *deep linking* (intente), no qual é incluída a rota recebida, demonstrado no diagrama de sequência da Figura 20.

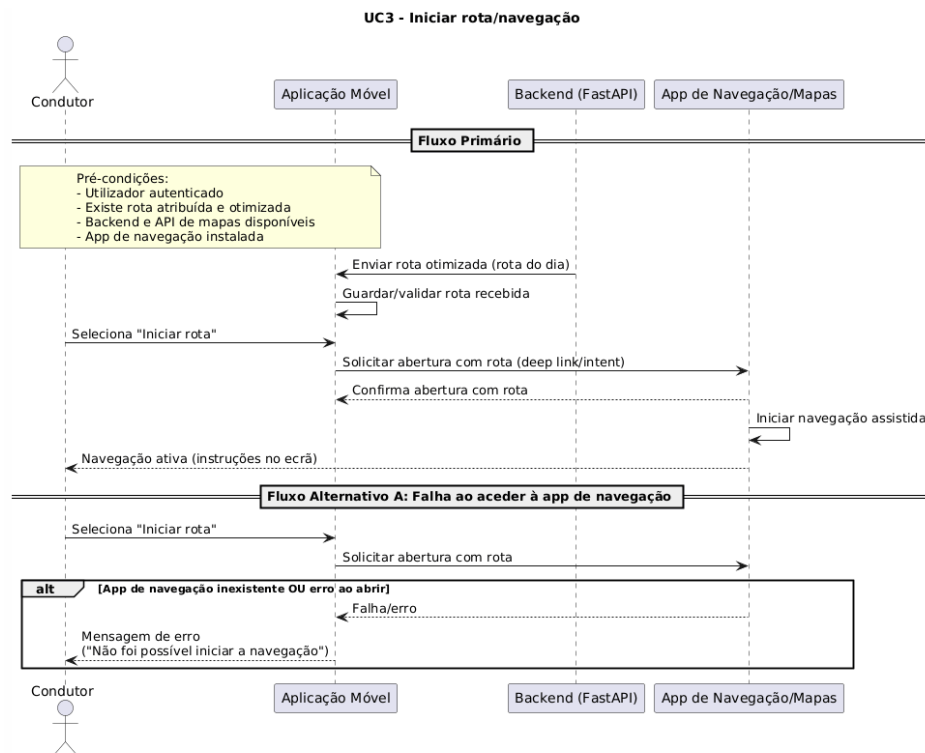


Figura 19 - Diagrama de sequência iniciar navegação/rota

A aplicação de navegação confirma a abertura com os dados da rota recebidos e inicia automaticamente o modo de navegação assistida, a partir deste momento o utilizador passa a visualizar a navegação, entrando no estado de navegação ativa até ao final do percurso ou interrupção pelo utilizador (**Figura 19**). Ocorrendo falha a aplicação apresenta uma mensagem de erro.



Figura 20 - Interface de navegação

A navegação pode ser iniciada ao clicar no botão *start*, com mostrado na **Figura 20** do lado inferior direito.

IV.6.4 Atualizar estado da tarefa

O processo inicia-se quando o utilizador, já com a navegação suspensa ou terminada, acede à lista de tarefas. Selecionando a tarefa pretendida, o utilizador seleciona o botão “*Complete Task*”, abrirá novo ecrã para selecionar se a tarefa foi “*Completed*” ou “*No completed*”. A aplicação envia então essa informação para o *backend*, identificando a tarefa e o respetivo novo estado. O *backend* regista a alteração na base de dados e devolve uma confirmação à aplicação, que por sua vez retorna à rota diária, como demonstrado na **Figura 21**.

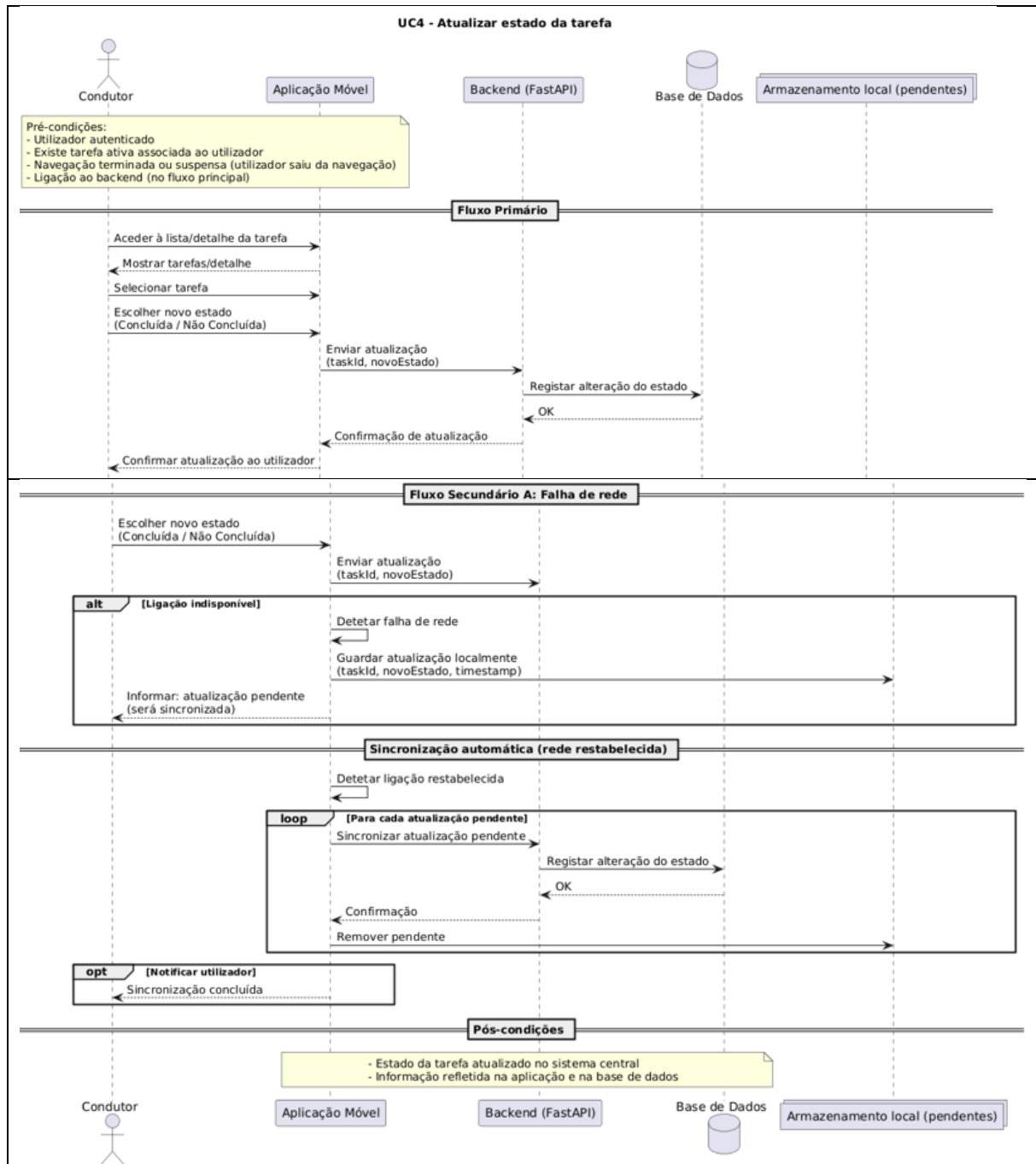


Figura 21 - Diagrama sequência alterar estado tarefa

Esta alteração do estado da tarefa é efetuada obrigatoriamente com a navegação pausada ou terminada, de forma garantir que o condutor não se encontra em condução no momento da interação com a aplicação, contribuindo assim para o aumento da segurança rodoviária.

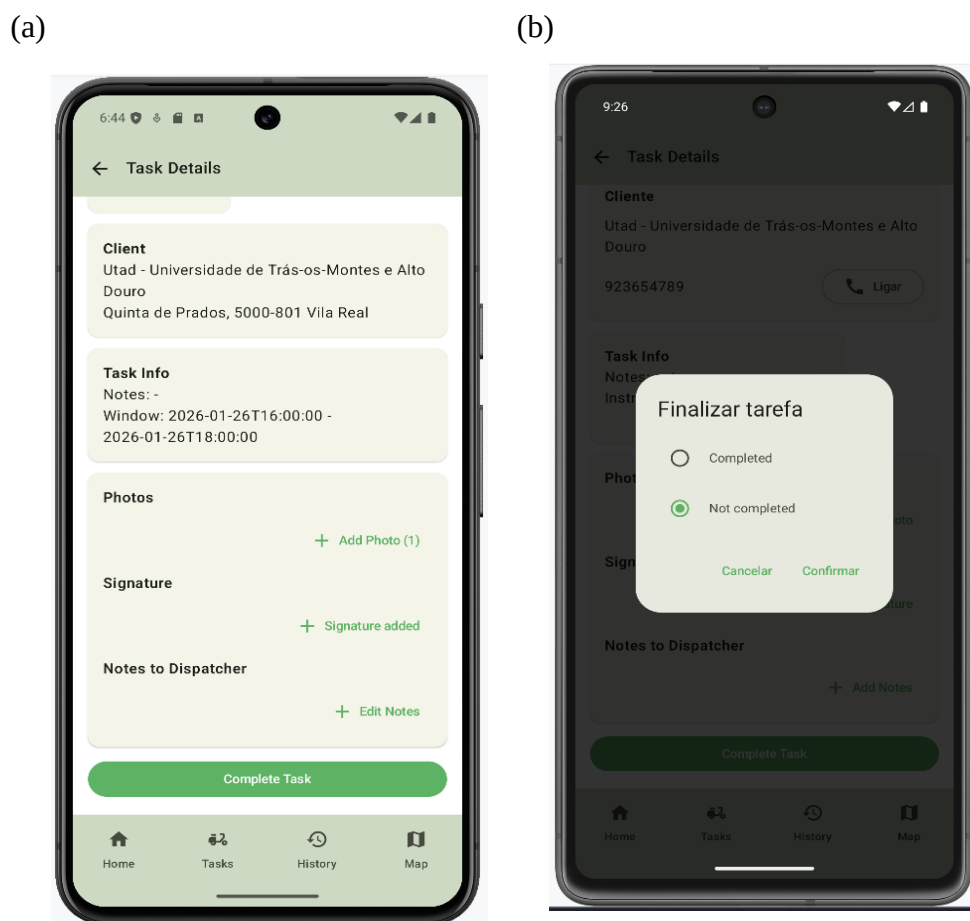


Figura 22 - Interfaces (a) detalhe da tarefa (b) finalizar tarefa

As interfaces apresentadas na **Figura 22** apresentam o detalhe da tarefa e, ao seleccionar “Complete task”, apresentará o interface para finalizar a tarefa, alterando o estado na base de dados.

IV.6.5 Apresentar rota atualizada

Em seguida ao processo indicado na secção anterior, a interface é automaticamente direcionada para a rota do dia atualizada.

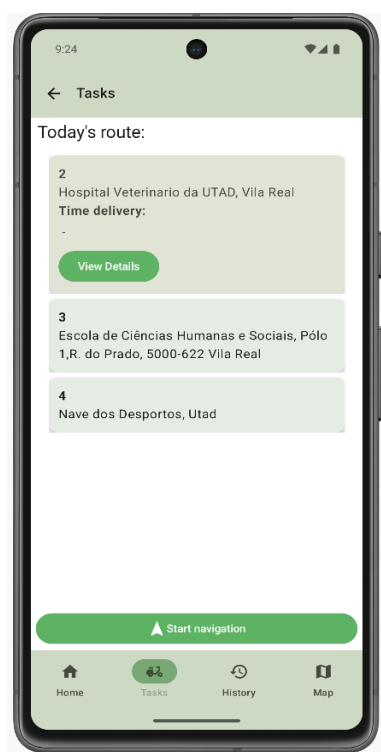


Figura 23 - Interface rota atualizada

A *interface* apresentada na **Figura 23** demonstra a dinâmica da aplicação na apresentação da rota otimizada, evidenciando uma visualização clara, eficiente e orientada para a utilização em contexto de condução.

IV.6.6 Consultar detalhe da tarefa

Sendo selecionado o botão “*View Detail*” será apresentada os detalhes da tarefa (**Figura 24**), em conjunto a todos os dados necessários para a alteração do estado da tarefa, bem como acesso ao botão de chamada, caso seja necessário entrar em contato com o cliente.

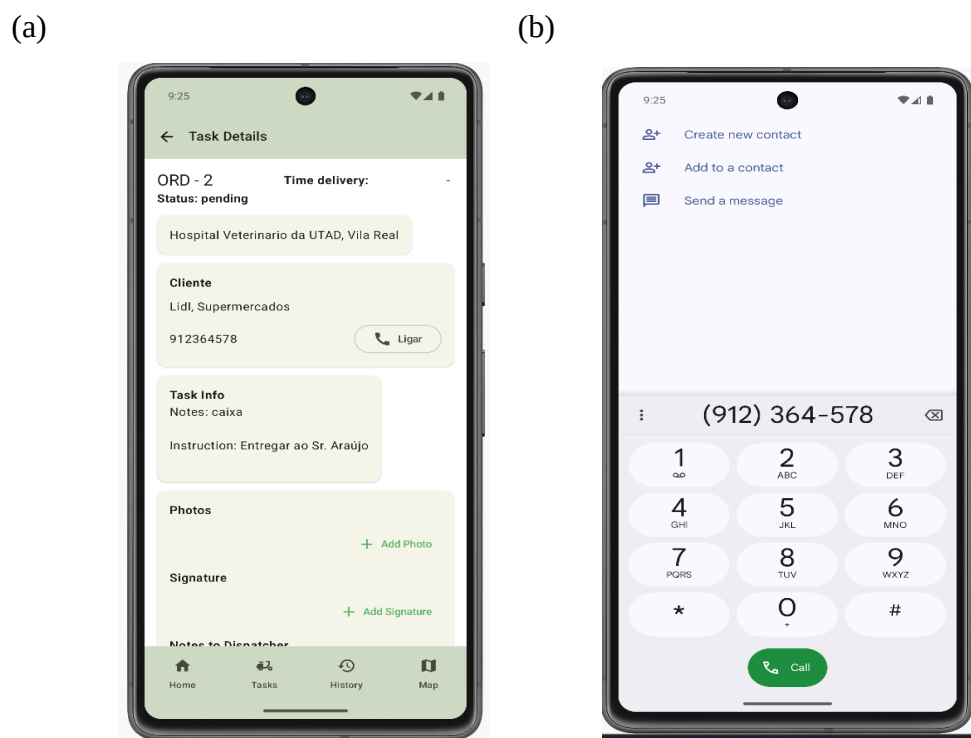


Figura 24 - Interfaces (a) detalhe da tarefa (b) chamada ao cliente

Essas *interfaces* foram criadas de modo a que o utilizador não encontre dificuldades significativas na sua utilização, permitindo que todo o processo seja realizado de forma simples, intuitiva e rápida.

IV.6.7 Consultar histórico da tarefa

A aplicação disponibiliza uma *interface* de histórico de tarefas, permitindo ao utilizador manter o registo das rotas e tarefas efetuadas. Esta funcionalidade possibilita a aplicação de filtros, nomeadamente por data, criação e estado da tarefa, proporcionando assim uma navegação mais organizada, fácil acesso à informação relevante e maior rapidez na consulta do histórico (**Figura 25**).

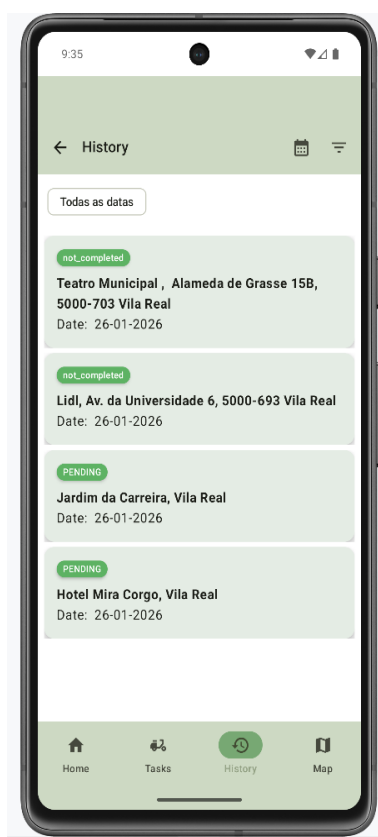


Figura 25 - Interface Histórico

IV. 7 Resumo do capítulo

Neste capítulo foi abordado todo o processo de desenvolvimento da aplicação, desde da implementação das tecnologias de apoio e desenvolvimento. Os primeiros resultados da aplicação são positivos pois, em ambiente laboratorial, permitiram responder aos requisitos definidos e identificar os erros e limitações do sistema. No capítulo seguinte serão introduzidos os testes realizados ao sistema em ambiente laboratorial. Estes testes permitirão validar todas as funcionalidades de cada serviço desenvolvido, assim como validar as funcionalidades possíveis.

V. TESTE E VALIDAÇÃO

O processo de testes e validação desempenham um papel fundamental no desenvolvimento de aplicações, garantindo não apenas que as funcionalidade adequada, mas também a confiabilidade e a eficiência do sistema. Neste capítulo, são descritos os testes realizados, nomeadamente, testes unitários que avaliam componentes individuais da aplicação, integração, que têm como objetivo avaliar a integração entre os diferentes componentes do sistema, aplicações e serviços e de usabilidade. Esses testes serão feitos através de caso de estudo para testes funcionais, do ambiente de testes do Android Studio, para os testes unitários e *checlist* da Qualidade - App quality da Android Developers adaptada a aplicação (Android Developers, 2026).

V.1 Testes funcionais - Caso de estudo

Para efetuar os testes deste trabalho, tendo em conta a impossibilidade de contacto direto com utilizadores finais, uma vez que o sistema se encontra ainda numa fase de desenvolvimento local, os mesmos foram efetuados através do emulador disponibilizado pelo Android Studio. Esta abordagem permitiu simular diferentes dispositivos e versões do sistema operativo Android, possibilitando a avaliação do comportamento funcional. Através de cenário, estes testes foram realizados manualmente tendo em conta os requisitos funcionais definidos após análise, exemplificado na **Figura 26**. Nos testes funcionais serão abrangidos os requisitos RF1, RF2, RF3, RF4, RF5, RF6 e RF7.

Os cenários realizados podem ser consultados detalhadamente no apêndice E. No capítulo E.1 são apresentados os testes da autenticação, o E.2 consultar rota do dia e tarefas, no E.3 consultar detalhe e atualizar estado, no capítulo E.4 visualização da rota atualizada, no capítulo E.5 será testado o histórico mais os filtros, no capítulo E.6, o recebimento de notificações e mensagens. No capítulo E.7 foi testado o recebimento de notificações e mensagens através do Firebase, o E.8 é apresentado testes instrumentados de interface e E.9 a compatibilidade da aplicação em diferentes tamanhos de ecrã.

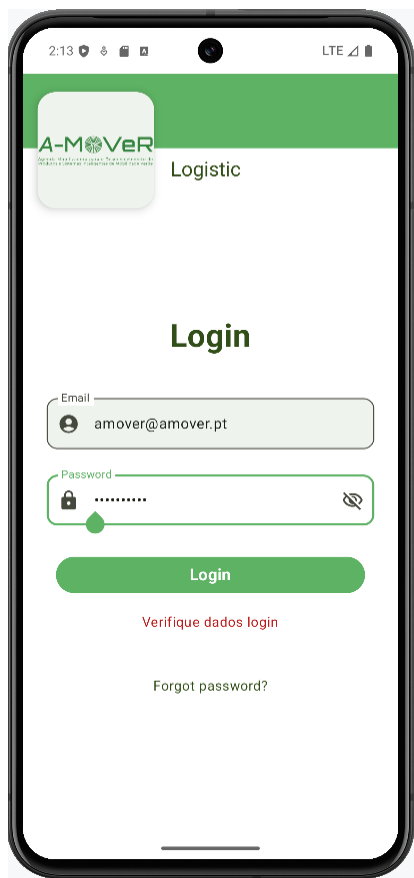


Figura 26 - Exemplo de teste efetuado - validação de credenciais

V.2 Teste de integração

Os testes de integração que serão realizados nos serviços são relacionados à verificação correta da comunicação entre diferentes serviços e aos formatos de resposta utilizados. Visa garantir a interoperabilidade entre as aplicações e os diversos serviços. Os testes realizados podem ser encontrados detalhadamente no apêndice F. No capítulo F.1 são apresentados os testes de integração relativos à autenticação e histórico de tarefas, através da própria plataforma *online* do Supabase, com apresentado na **Figura 27**. No capítulo F.2 os testes relativos aos serviços de otimização de rotas e alteração do estado da tarefa e por fim, no capítulo F.3 são apresentados os testes ao recebimento de mensagens e notificações.

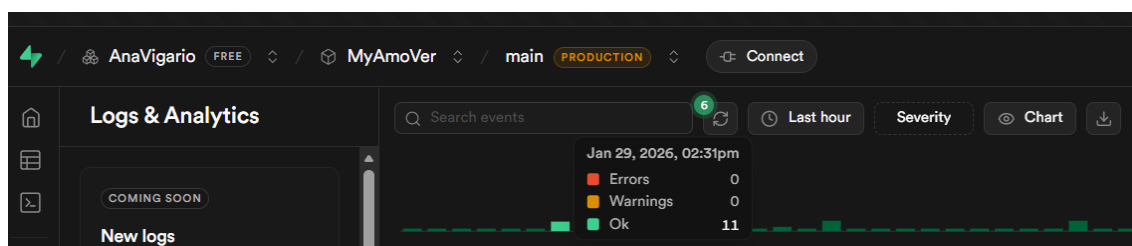


Figura 27 - Confirmação de *logs* ok do Supabase

V.3 Testes de performance/desempenho

Os testes de performance são focados no teste de componentes da aplicação, será aplicada aos serviços relevantes para a aplicação. Em relação aos testes de performance foram executados com recurso ao Android Profiler (**Figura 28**), integrado no Android Studio, analisando o comportamento da aplicação em tempo real ao nível de consumo e tempos de respostas às principais interações do utilizador. Por outro lado, foram realizados testes instrumentados de desempenho, utilizando a biblioteca Android Macrobenchmark, para avaliar o comportamento temporal da aplicação.

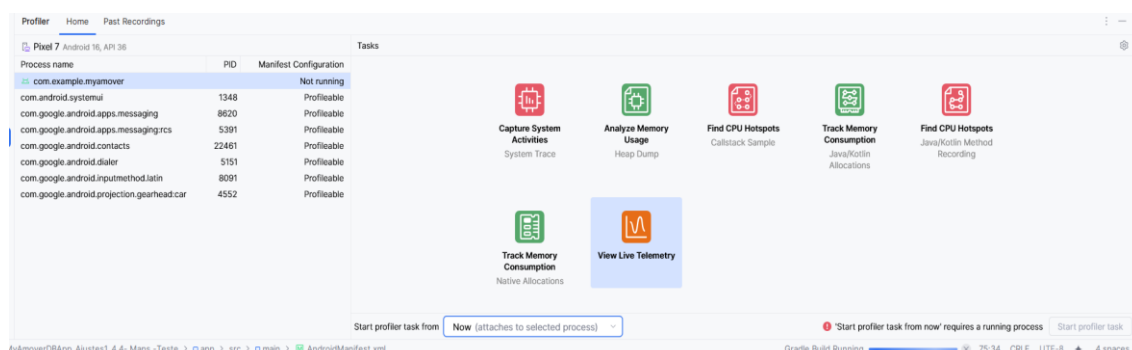


Figura 28 - Android Profiler

Os testes realizados podem ser encontrados detalhadamente no apêndice G. No capítulo G.1 são apresentados os testes de arranque da aplicação, no capítulo G.2 carregamento da lista de tarefas. No capítulo G.3 são apresentados os testes de apresentação das rotas e o capítulo G.4 o teste às atualizações do estado da tarefa, enquanto que, no G.5 é apresentado a avaliação de desempenho do tempo de arranque da aplicação.

V.4 Testes unitários

Os testes unitários foram criados com o objetivo de testar a lógica interna dos ViewModels de forma isolada, sem dependência de serviços externos. Estes testes foram executados em ambiente JVM, recorrendo a *mocks* e repositórios simulados, seguindo princípios de Clean Architecture no próprio Android Studio. Os testes realizados podem ser encontrados no apêndice H. No capítulo H.1 ao LoginViewModel (**Figura 29**), enquanto no capítulo H.2 apresenta testes efetuados à TaskViewModel.

Test Summary

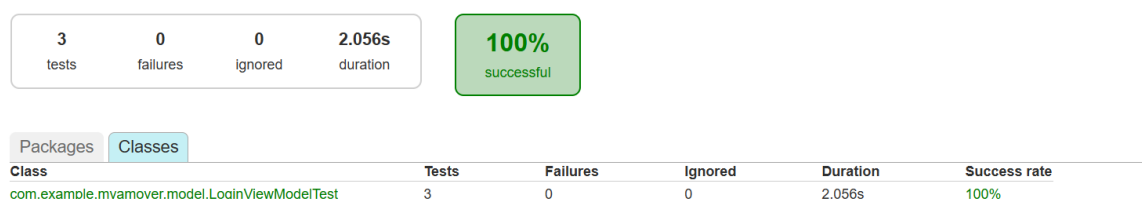


Figura 29 - Teste Summary LoginViewModel

V.5 Resultados dos testes - aplicação

De um modo geral, a aplicação apresentou resultados positivos em todos os testes realizados, demonstrando um comportamento estável e consistente face aos requisitos funcionais e não funcionais definidos. Os testes funcionais permitiram validar corretamente os principais fluxos de utilização, nomeadamente a autenticação do utilizador, a consulta e gestão de tarefas, a visualização e atualização de rotas, bem como a ativação de navegação externa, confirmando o correto funcionamento das funcionalidades implementadas.

No que respeita aos testes de integração, verificou-se uma comunicação eficaz entre a aplicação, a API desenvolvida em FastAPI e a plataforma Supabase. As requisições efetuadas apresentaram respostas bem-sucedidas, evidenciando a correta persistência e recuperação da informação na base de dados. Em complemento, os testes instrumentados de funcionalidade confirmaram que esta integração se reflete corretamente na interface do utilizador, assegurando que as operações remotas decorram sem falhas.

Relativamente ao desempenho, a análise revelou tempos de respostas adequados às interações do utilizador, com valores geralmente inferiores a 100ms para eventos de toque. Foram identificados picos pontuais durante as operações mais exigentes, comportamentos

temporários associados sobretudo à fase de composição gráfica, estabilizando posteriormente por valores compatíveis com a experiência de utilização fluida. Estes resultados são coerentes com os testes instrumentados de funcionalidades realizados sobre a interface principal, os quais confirmaram a estabilidade da navegação, a correta execução dos callbacks e a consistência do fluxo de interação do utilizador.

A monitorização da aplicação através do Firebase permitiu confirmar a estabilidade da versão testada, bom como a sua utilização ativa. A versão foi classificada como eficiente, o que reforça a recepção das mensagens e notificação por parte da aplicação.

Os testes unitários revelaram eficiência do sistema ao nível da lógica interna, confirmando a previsibilidade do comportamento dos componentes e a robustez da arquitetura adotada.

Em síntese, os resultados obtidos demonstram que a aplicação cumpre os objetivos propostos, apresentando um desempenho satisfatório, uma integração eficaz com os serviços e uma experiência de utilização adequada, validando assim as opções técnicas e arquiteturais adotadas no desenvolvimento do sistema.

V.6 Algoritmo de otimização

O objetivo principal do estudo dos algoritmos de otimização foi comparar soluções que maximizasse a eficiência na operação, sendo que cada uma das técnicas foi adaptada de forma a considerar as particularidades dos veículos elétricos, nomeadamente no que respeita à autonomia limitada e à necessidade de planeamento de recargas ao longo do percurso. Para responder eficazmente a essas diferentes variantes, foram utilizadas abordagens exatas, heurísticas e meta-heurísticas, incluindo técnicas como Branch and Bound.

Os modelos de otimização de rotas foram testados a partir de cenários criados para sua avaliação, tais como:

- *Cenário simples de entrega*: Uma frota de veículos elétricos deveria atender 6 clientes a partir de um único depósito, veículos com capacidade limitada de carga e bateria, tendo como objetivo minimizar a distância total percorrida;

– *Cenário com Janelas de Tempo*: Veículos com capacidade limitada de carga e clientes com janelas de tempo para serem atendidos, influenciando o tempo de viagem entre locais;

– *Cenário dinâmico com recarga*: Veículos com capacidade limitada de carga, novos pedidos surgiam enquanto as entregas eram realizadas e necessidade de recarga da bateria, obrigando incluir paragens em pontos próprios de abastecimento;

– *Cenário de múltiplos depósitos*: Dois depósitos atendem uma cidade, cada um com sua frota, os clientes são atribuídos ao depósito mais próximo, minimizando a distância total percorrida.

Estes, entre outros cenários explorados, permitiram avaliar a eficácia dos algoritmos propostos em diferentes condições operacionais. Para avaliar o desempenho dos algoritmos aplicados, foram definidas as seguintes métricas:

- Distância total percorrida (em km): principal métrica de eficiência energética, representando o custo total das rotas calculadas;
- Tempo total de execução (em segundos): indicador fundamental em cenários dinâmicos, medindo a eficiência computacional do método;
- Número de visitas ao depósito: mede a necessidade de recarregamento ou reposição, refletindo a autonomia operacional;
- Respeito às restrições: avalia a capacidade do algoritmo em cumprir janelas de tempo, limites de carga e autonomia dos veículos;
- Robustez e escalabilidade: expressa a capacidade de adaptação do algoritmo a alterações de parâmetros e ao aumento da complexidade do problema.

Estas métricas foram aplicadas a um conjunto de cenários com diferentes características, baseados em situações logísticas comuns em uma cidade de média dimensão como Vila Urbana. Estes cenários foram construídos com dados representativos, incluindo coordenadas geográficas, demandas específicas, janelas temporais e restrições operacionais (capacidade de carga, autonomia energética e limites de tempo).

V.7 Resumo do capítulo

Este capítulo apresentou a metodologia de testes e validação da aplicação desenvolvida, evidenciando uma abordagem estruturada e multidimensional garantindo a

qualidade do sistema. Foram realizados testes funcionais, de integração, desempenho e testes unitários, cobrindo desde a validação de fluxos de utilização até à análise da lógica interna dos componentes.

Os testes funcionais permitiram confirmar o correto funcionamento das principais funcionalidades da aplicação, enquanto os testes de integração validaram a comunicação eficaz entre frontend, backend e base de dados. A avaliação de desempenho demonstrou tempos de resposta adequados e estabilidade operacional, mesmo em cenários de maior exigência e os testes unitários confirmaram a previsibilidade e robustez da arquitetura interna.

Para além dos testes referentes à aplicação, também, foram efetuados teste aos algoritmos de otimização. Foi demonstrado que a aplicação de algoritmos de otimização de rotas em cenários urbanos reais é não apenas viável, mas também altamente eficaz no apoio à operação profissional de motas elétricas.

No conjunto, os resultados obtidos demonstram que a aplicação apresenta comportamento estável, integração consistente entre serviços e uma experiência de utilização adequada, validando as decisões técnicas adotadas ao longo do desenvolvimento, bem como a inclusão da otimização de rotas. A seguir, será apresentado o capítulo V, conclusões e trabalhos futuros.

VI. CONCLUSÕES E TRABALHOS FUTUROS

Neste capítulo final, apresentam-se as conclusões decorrentes deste trabalho, juntamente com as limitações identificadas. Além disso, são delineadas algumas perspetivas

futuras, apontando a melhoria e expansão das funcionalidades destas aplicações, bem como o possível surgimento de novos projetos, impulsionados pela realização deste trabalho.

VI.1 Conclusões

O presente trabalho, teve como principal objetivo apresentar o desenvolvimento e validação de um *frontend* de plataforma de gestão e monitorização de rotas de motas elétricas em contexto de serviços urbanos. Para o desenvolvimento do trabalho, foi necessário conhecimento multidisciplinar, partindo da análise do estado da arte e dos desafios associados à experiência de utilização de veículos de duas rodas, foi possível conceber uma solução que integra funcionalidade de gestão de tarefas, otimização de rotas e navegação, suportada por uma arquitetura moderna e escalável. Estudos efetuados sobre a condução e a interação do motociclista, com os dispositivos de informação e a estrada, mostraram a necessidade e desafio na construção da *interface*.

A realização de estudos conceituais e a definição de especificações de *interfaces* contextuais para a condução de motociclos, suportadas por várias revisões da literatura e estudos sobre aplicações existentes no mercado, serviram de base na criação de várias soluções de aplicações enquadradas na condução desses veículos. Os estudos efetuados sobre: *design* e experiência do utilizador (UI/UX) constituíram um elemento fundamental para a construção da interface gráfica da aplicação, combinando modernidade, usabilidade e acessibilidade. Esses contributos servirão igualmente de suporte a melhorias futuras, quando disponibilizadas ao utilizador final, permitindo a evolução da *interface*.

A fase de implementação permitiu materializar os conceitos definidos, recorrendo a tecnologias atuais e amplamente adotadas, como o Jetpack Compose para o desenvolvimento da *interface*, algoritmos de otimização para o cálculo das rotas e serviços REST para a comunicação entre as diversas partes. A aplicação desenvolvida demonstra ser capaz de responder aos principais desafios identificados, assegurando uma experiência de utilização intuitiva e adequada ao contexto de utilização em mobilidade.

Após o desenvolvimento da aplicação, este foi testado e validado em ambiente laboratorial. Os testes mostraram o correto funcionamento e também o correto desenvolvimento tendo em conta os requisitos e os casos de uso definidos. Os resultados obtidos evidenciam que a aplicação cumpre os objetivos propostos, apresentando uma integração eficaz a serviços externos, tempos de resposta adequados às interações do

utilizador e um comportamento estável durante a execução das operações mais críticas, como a atualização do estado das tarefas e a visualização das rotas.

Neste trabalho foram alcançadas diversas contribuições importantes no campo da logística, destacando-se a necessidade da criação de um *frontend* capaz de integrar serviços de otimização, gestão de tarefas e navegação de apoio em operações de logística urbana sustentável, evidenciando o potencial das motas elétricas com alternativa eficiente e ambientalmente responsável. A solução desenvolvida constitui uma base sólida para a exploração de novas funcionalidades e para a adaptação a diferentes contextos operacionais.

Em conclusão, os objetivos definidos para esta dissertação foram alcançados, resultando numa solução funcional, validada e com potencial de aplicação prática, que contribui para a área das aplicações de logística e navegação em ambiente urbanos.

VI.2 Limitações e trabalhos futuros

Durante o desenvolvimento deste projeto, algumas limitações foram identificadas. Uma das principais limitações reside na ausência de validação em ambiente real de operação. Embora tenham sido realizados testes funcionais, de integração e de desempenho em ambiente controlado, a aplicação não foi testada de forma prolongada em contexto real. Esses testes permitiriam avaliar de forma mais precisa aspetos como a usabilidade durante a condução, a robustez da aplicação face a variações de rede e o impacto da utilização prolongada no desempenho do dispositivo.

No que respeita à otimização de rotas, os algoritmos implementados baseiam-se em pressupostos simplificados e em dados previamente definidos. Não foram considerados, de forma dinâmica, fatores como condições de trânsito em tempo real, acidentes ou condições meteorológicas, os quais podem influenciar significativamente o desempenho das rotas em contexto real. A inclusão de dados em tempo real, para a aplicação nos algoritmos estudados seria essencial para melhorar a precisão e a adaptabilidade das rotas geradas.

As limitações decorrentes da implementação do sistema no contexto do ecossistema A-Mover condicionaram a avaliação da solução a operações essencialmente locais. Não foi possível avaliar o comportamento fora da rede interna, a realização de testes em infraestruturas distribuídas e com maior volume de dados constitui uma etapa essencial para a consolidação e maturação da solução em contextos reais de utilização.

Para trabalhos futuros, destacam-se várias possibilidades de evolução da aplicação, tais como a melhoria dos algoritmos de otimização de rotas com base em tempo real, a

introdução de mecanismos de aprendizagem automática para previsão de tempos de entrega, a integração da aplicação ao ecossistema A-Mover e, e por fim, a integração da navegação, notificações e mensagens ao *display* da mota.

Estas direções para trabalho futuro têm como objetivo superar as limitações identificadas e introduzir melhorias significativas nas aplicações e a integração com sistemas reais. Por fim, este trabalho pode fornecer perspetivas interessantes para a continuidade da ideia aqui apresentada, e possivelmente para o surgimento de novos projetos.

VII. REFERÊNCIAS

A-MoVeR. (2026). *A-MoVeR – Site do projeto A-MoVeR*. Universidade de Trás-os-Montes e Alto Douro. <https://amover.utad.pt/>

Android Developers (2025). *Download Android Studio & App Tools - Android Developers*. Android Developers. <https://developer.android.com/studio?hl=pt-br>

Android. (2025). *O que é Android? Veja os recursos e modelos principais*. Android. https://www.android.com/intl/pt_br/articles/what-is-android/

Android Developers. (2026). *Desenvolver a interface | Jetpack Compose*. Android Developers. <https://developer.android.com/develop/ui?hl=pt-br>

Autobot. (2024). *How to use APIs for mobility*. <https://apibrasil.blog/como-usar-apis-para-mobilidade/>

Bagabaldo, A. R., Gan, Q., Bayen, A. M., & González, M. C. (2024). *Impact of navigation apps on congestion and spread dynamics on a transportation network*. *Deleted Journal*, 6(2). <https://doi.org/10.1007/s42421-024-00099-w>

Bernard, T. (2025). *Best navigation apps 2025*. *Auto Express*. <https://www.autoexpress.co.uk/product-group-tests/63573/best-navigation-apps-2025>

Black, J., Páez, A., & Suthanaya, P. (2002). *Sustainable urban transportation: Performance indicators and some analytical approaches*. *Journal of Urban Planning and Development*, 128(4), 184–209. [https://doi.org/10.1061/\(ASCE\)0733-9488\(2002\)128:4\(184\)](https://doi.org/10.1061/(ASCE)0733-9488(2002)128:4(184))

Chadalavada, S., Faust, O., Salvi, M., Seoni, S., Raj, N., Raghavendra, U., Gudigar, A., Barua, P. D., Molinari, F., & Acharya, R. (2025). *Application of artificial intelligence in air pollution monitoring and forecasting: A systematic review*. *Environmental Modelling and Software*, 185, Article 106312. <https://doi.org/10.1016/j.envsoft.2024.106312>

Chen, Y., Li, C., Wang, W., Zhang, Y., Chen, X. M., & Gao, Z. (2025). *The landscape, trends, challenges, and opportunities of sustainable mobility and transport*. *NPJ Sustainable Mobility and Transport*, 2(1), 1–14. <https://doi.org/10.1038/s44333-025-00026-8>

Chen, Z. (2025). *Algorithm and application in vehicle routing problem: A review*. *Journal of Electronic Research and Application*, 9(2), 166–174. <https://doi.org/10.26689/jera.v9i2.9939>

Chilro, G., Oliveira, P., Nunes, R., Barroso, J., & Rocha, T. (2024). *The role of UI/UX designs for enhancing safety and motorcycle riders' experience*. In *Lecture Notes in Computer Science* (pp. 13–22). https://doi.org/10.1007/978-3-031-76824-8_2

Connected Motorcycle Consortium. (2023). *CMC motorcycle simulator study on rider reaction time II* (Version 2.1). Connected Motorcycle Consortium.

Dias, D., Vigário, A., Pinto, T., Reis, A., & Barroso, J. (2025). *Smart logistics management for electric motorcycles*. In *Proceedings of the 2025 IEEE Vehicle Power and Propulsion Conference (IEEE VPPC 2025)*.

Emma, L. (2024, December 2). *User-centered design to enhance accessibility and usability in digital systems*. ResearchGate. https://www.researchgate.net/publication/386339454_User-centered_design_to_enhance_accessibility_and_usability_in_digital_systems

Falcone, K. C., & Miola, G. A. R. (2024). *User experience (UX) e seu impacto no desenvolvimento de sistemas: A importância do design centrado no usuário*. https://ric.cps.sp.gov.br/bitstream/123456789/28893/1/ads_2024_2_kemellyncfalcone_userexperienceuxeseuimpacto.pdf

Feo, T. A., & Resende, M. G. C. (1995). *Greedy randomized adaptive search procedures*. *Journal of Global Optimization*, **6**, 109–133.

Fernandes, R., Vigário, A., Catarino, P., Vasco, P., Reis, A., Pires, E.S., Barroso, J. & Pinto, T. (2025). *Route Optimization Problems and Algorithms for Urban Service Electrical Motorbike Fleets*. In *Proceedings Distributed Computing and Artificial Intelligence, 21st International Conference (DCAI2025)*. Lille, França: Springer (Lecture Notes in Networks Systems).

Gebretsadik, K. H., & Suzuki, K. (2024). *Riding behavior-based HMIs for motorcycle safety: A comparative study of alert and real-time feedback systems*. *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, **18**(5), JAMDSM0055. <https://doi.org/10.1299/jamdsm.2024jamdsm0055>

Google for Developers. (2006). *Biblioteca do Maps Compose*. <https://developers.google.com/maps/documentation/android-sdk/maps-compose?hl=pt-br>

Google OR-Tools. (2024). *Problema de definição de trajeto de veículos com janelas de tempo*. <https://developers.google.com/optimization/routing/vrptw?hl=pt-br>

Glovo. (2025). *Home – Glovo corporate site*. <https://about.glovoapp.com/>

Gong, L., & Ding, M. (2025). *Urban traffic digital twin system development in Unity*. *Scientific Reports*, **15**(1). <https://doi.org/10.1038/s41598-025-23943-7>

Goodwin, M. (2024). *What is an API?* IBM. <https://www.ibm.com/think/topics/api>

Grupo Ageas Portugal. (2023). *Estudo sobre a micromobilidade nos centros urbanos*. <https://www.grupoageas.pt/sobre-o-grupo-ageas/noticias-do-grupo/estudo-sobre-a-micromobilidade-nos-centros-urbanos>

Gupta, M., & Velaga, N. R. (2025). *Impact of psychological factors and individual characteristics on advanced warning design for motorized two-wheelers*. *Transportation Research Part F: Traffic Psychology and Behaviour*, **109**, 94–112. <https://doi.org/10.1016/j.trf.2024.11.025>

Hexaly. (n.d.). *Traveling salesman problem (TSP)*. Hexaly. <https://www.hexaly.com/templates/traveling-salesman-problem-tsp>

Hexaly. (n.d.). *Capacitated vehicle routing problem (CVRP)*. <https://www.hexaly.com/docs/last/exampletour/vrp.html>

Hexaly. (2024, November 18). *Multi-depot vehicle routing problem (MDVRP)*. <https://www.hexaly.com/templates/multi-depot-vehicle-routing-problem-mdvrp>

Holiday Extras. (2025). *Best navigation apps [Travel advice]*. <https://www.holidayextras.com/travel-advice/best-navigation-apps.htm>

International Energy Agency. (2025). *Trends in electric car markets – Global EV Outlook 2025*. <https://www.iea.org/reports/global-ev-outlook-2025/trends-in-electric-car-markets-2>

International Motorcycle Manufacturers Association. (2025). *Safer motorcycling: The global motorcycle industry's approach to road safety*. https://www.immamotorcycles.org/wp-content/uploads/2024/12/IMMA_White-Paper_2025.pdf

Ito, K., Nishimura, H., & Ogi, T. (2019). *Motorcycle head-up display: Design of presenting navigation information*. *IEEE Consumer Electronics Magazine*, **8**(5), 74–78. <https://doi.org/10.1109/MCE.2019.2923934>

JetBrains. (2020). *Kotlin programming language*. Kotlin. <https://kotlinlang.org/>

Kasalawat, N., & Shahi, T. C. (2024). *Determination of motorcycle equivalent unit and road capacity: A case study of selected road sections of Kathmandu Valley*. *International Journal on Engineering Technology*, **1**(2), 59–73. <https://doi.org/10.3126/injet.v1i2.66705>

K, A. A. (2025). *How Uber Eats architecture works: A deep dive into building a global, real-time food delivery system*. Medium. <https://medium.com/@aliaftabk/how-uber-eats-architecture-works-a-deep-dive-into-building-a-global-real-time-food-delivery-222dde27666f>

- Kucharska, E.** (2019). *Dynamic vehicle routing problem—Predictive and unexpected customer availability*. *Symmetry*, **11**(4), 546. <https://doi.org/10.3390/sym11040546>
- Kuo, M.** (2025). *Ultimate guide to route optimization*. Routific. <https://www.routific.com/route-optimization>
- Lawler, E. L., & Wood, D. E.** (1966). *Branch-and-bound methods: A survey*. *Operations Research*, **14**(4), 699–719.
- Ledesma, R. D., Padilla, J.-L., Tosi, J. D., Sánchez, N., & Castro, C.** (2023). *Motorcycle rider error and engagement in distracting activities: A study using the Attention-Related Driving Errors Scale (ARDES-M)*. *Accident Analysis & Prevention*, **187**, 107069. <https://doi.org/10.1016/j.aap.2023.107069>
- Livermore, J. A.** (2007, March 1). *Factors that impact implementing an agile software development methodology*. *IEEE Xplore*. <https://doi.org/10.1109/SECON.2007.342860>
- National Traffic Designs.** (n.d.). *Traffic data collection and analysis by consultants*. <https://www.nationaltrafficdesigns.com/traffic-data-collection-and-analysis-by-consultants>
- Nivedhaa, N.** (2024). *Software architecture evolution: Patterns, trends, and best practices*. *International Journal of Computer Sciences and Engineering (IJCSE)*, **1**(2), 1–14.
- Objetivos de Desenvolvimento Sustentável.** (2025, 23 de julho). *Nações Unidas – ONU Portugal*. <https://unric.org/pt/objetivos-de-desenvolvimento-sustentavel/?nowprocket=1>
- Onfleet.** (2024). *Getting started with the Onfleet driver app* [Vídeo]. YouTube. <https://www.youtube.com/watch?v=zeqNJ3GC3g0>
- Onfleet.** (2025). *Onfleet* [Página institucional]. <https://onfleet.com/>
- OptimoRoute.** (2025). *Help center* [Página de suporte]. <https://help.optimoroute.com/hc/en-us>
- Parlamento Europeu.** (2023). *2023/1058 (OJ C, 1058)*. EUR-Lex.
- Patel, R.** (2023, September 19). *What is Dijkstra’s algorithm? A detailed guide*. <https://www.upperinc.com/glossary/route-optimization/dijkstras-algorithm/https://data.europa.eu/eli/C/2023/1058/oj>
- Perelló, V., & Malatini, H.** (2024). *How we engineered a scalable architecture to power videos, social, and picks in our delivery app*. *The Glovo Tech Blog*. <https://tech->

blog.glovoapp.com/blog/how-we-engineered-a-scalable-architecture-to-power-videos-social-and-picks-in-our-delivery-app/

Rajesh, K., Jain, E., & Kotecha, P. (2022). *A multi-objective approach to the electric vehicle routing problem*. arXiv. <https://doi.org/10.48550/arXiv.2208.12440>

Renninger, A., Noman, S. A., Atkison, T., & Sussman, J. (2024). *Live intersection data acquisition for traffic simulators (LIDATS)*. *Sensors*, **24**(11), 3392. <https://doi.org/10.3390/s24113392>

Routific. (2025). *Routific help center* [Página de suporte]. <https://help.routific.com/en>

Sadhukhan, T., Paul, M., & Dwivedi, R. (2025). *Adaptively-weighted nearest neighbors for matrix completion (arXiv:2505.09612)*. arXiv. <https://arxiv.org/abs/2505.09612>

SoftCircles. (2024). *Transportation API: Building mobility-aware applications*. SoftCircles. <https://softcircles.com/blog/transportation-api-building-mobility-aware-applications>

Sunday, V. (2024a). *How real-time data transforms navigation apps: From traffic to weather alerts*. MapMetrics. <https://mapmetrics.org/pt/blog/how-real-time-data-transforms-navigation-apps/>

Sunday, V. (2024b). *How navigation apps provide live traffic updates on the road*. MapMetrics. <https://mapmetrics.org/pt/blog/live-traffic-updates-on-the-road>

Supabase. (2025). *Supabase docs*. Supabase. <https://supabase.com/docs>

Teledjieu, I. D., & Shafique, I. (2025). *CityPulse: Real-time traffic data analytics and congestion prediction*. arXiv. <https://arxiv.org/abs/2506.01971>

Thangavelu, M., Parwal, A., & Patali, R. (2021). *The architecture of Uber's API gateway*. Uber Engineering. <https://www.uber.com/en-PT/blog/architecture-api-gateway/>

Toklu, M. C. (2022). *A fuzzy multi-criteria approach based on Clarke and Wright savings algorithm for vehicle routing problem in humanitarian aid distribution*. *Journal of Intelligent Manufacturing*, **34**(5), 2241–2261. <https://doi.org/10.1007/s10845-022-01917-0>

TomTom. (2025). *Introduction | Traffic stats API* [Documentação]. TomTom Developer Portal. <https://developer.tomtom.com/traffic-stats/documentation/product-information/introduction>

Tucson, A., Glover, F., & Laguna, M. (1999). *Tabu search*. *Journal of the Operational Research Society*, **50**(1), 106. <https://doi.org/10.2307/3010402>

Uber Eats. (2025). Sobre o Uber Eats [Página institucional]. <https://about.ubereats.com/pt-pt/>

United Nations. (2025). *World urbanization prospects 2025: Summary of results* (UN DESA/POP/2025/TR/No. 12). United Nations.

Useche, S. A., Lijarcio, I., & Montoro, L. (2025). **The rise of the “technological motorcyclist”: Exploring the predictors of digital distractions in motorcycle commuting.** *Revista Iberoamericana de Psicología y Salud*, 16(2), 44–60. <https://doi.org/10.70478/rips.2025.16.06>

UserTesting. (2023, August 18). *Automotive UX/UI: User research and interface design for modern vehicles.* UserTesting. <https://www.usertesting.com/blog/automotive-UX-UI>

Vigário, A., Teixeira, B., Penelas, G., Pinto, N., Reis, A. & Pinto, T. (2024). Plataforma de serviços de gestão e informação para a mobilidade urbana verde - Management and information services platform for green urban mobility. In *Proceedings 19th Iberian Conference on Information Systems and Technologies (CISTI'2024)*. Salamanca, Espanha: Springer (Lecture Notes in Networks Systems).

Vigário, A., Dias, D., Pinto, T., Reis, A., & Barroso, J. (2025a). *The role of real-time navigation APIs in green and smart mobility.* In *Proceedings of the PAMDAS 2025 – International Conference on Physical Asset Management and Data Science* (pp. 343–353).

Vigário, A., Fernandes, R., Pinto, T., Rocha, T. & Reis, A. (2025b). Otimização de Rotas para Motas Elétricas na Ótica do Utilizador - Route Optimization for Electric Motorcycles from the User’s Perspective. In *Proceedings 20th Iberian Conference on Information Systems and Technologies (CISTI'2025)*. Lisboa, Portugal: Springer (Lecture Notes in Networks Systems).

Vigário, A., Fernandes, R., Pinto, T., Reis, A., Rocha, T., & Barroso, J. (2025c). User-centric route optimisation models for green mobility decision support. In *Proceedings of the 27th International Conference on Human-Computer Interaction (HCI International 2025)*. Gothenburg, Sweden: Springer.

Vigário, A., Oliveira, J., Fernandes, R., Pinto, T., Reis, A., Rocha, T., Barroso, J. (2026). Adaptive User Interface for Electric Vehicle Route Information in Urban Mobility Services. In: *Tsihrintzis, G.A., Virvou, M., Marinakis, V., Jain, L.C. (eds.) Human-Smart City*

Interactions and User-Citizen Experiences. Learning and Analytics in Intelligent Systems, 58, 157-180. Springer, Cham. https://doi.org/10.1007/978-3-032-06364-9_8.

World Health Organization. (2025). *World health statistics 2025: Monitoring health for the Sustainable Development Goals*. Geneva: World Health Organization

Zhao, J., Zeng, Z., & Liu, Y. (2025). *Electric vehicle routing problem considering traffic conditions and real-time loads. Transportation Research Part C: Emerging Technologies*, 176, 105150. <https://doi.org/10.1016/j.trc.2025.105150>

APÊNDICES

Apêndice A – Publicações

Esta dissertação resultou nas seguintes publicações:

- Vigário, A., Teixeira, B., Penelas, G., Pinto, N., Reis, A. & Pinto, T. (2024). Plataforma de serviços de gestão e informação para a mobilidade urbana verde - Management and information services platform for green urban mobility. In *Proceedings 19th Iberian Conference on Information Systems and Technologies (CISTI'2024)*. Salamanca, Espanha: Springer (Lecture Notes in Networks Systems).
- Vigário, A., Penelas, G., Teixeira, B., Pinto, N., Fernandes, R., Reis, A. & Pinto, T. (2024). Information and Management Platform for Green, Sustainable and Intelligent Mobility. In *Proceeding Distributed Computing and Artificial Intelligence, Special Sessions I, 21st International Conference (DCAI2024)*. Salamanca, Espanha: Springer (Lecture Notes in Networks Systems).
- Vigário, A., Fernandes, R., Pinto, T., Rocha, T. & Reis, A. (2025). Otimização de Rotas para Motas Elétricas na Ótica do Utilizador - Route Optimization for Electric Motorcycles from the User's Perspective. In *Proceedings 20th Iberian Conference on Information Systems and Technologies (CISTI'2025)*. Lisboa, Portugal: Springer (Lecture Notes in Networks Systems).
- Fernandes, R., Vigário, A., Catarino, P., Vasco, P., Reis, A., Pires, E.S., Barroso, J. & Pinto, T. (2025). Route Optimization Problems and Algorithms for Urban Service Electrical Motorbike Fleets. In *Proceedings Distributed Computing and Artificial Intelligence, 21st International Conference (DCAI2025)*. Lille, França: Springer (Lecture Notes in Networks Systems).
- Vigário, A., Fernandes, R., Pinto, T., Reis, A., Rocha, T., & Barroso, J. (2025). User-centric route optimisation models for green mobility decision support. In *Proceedings of the 27th International Conference on Human-Computer Interaction (HCI International 2025)*. Gothenburg, Sweden: Springer.
- Vigário, A., Dias, D., Pinto, T., Reis, A., & Barroso, J. (2025). The Role of Real-time Navigation APIs in Green and Smart Mobility. In *Proceedings of PAMDAS 2025 – International Conference on Physical Asset Management and Data Science*, 343-353. Coimbra, Portugal. ISBN 978-989-8331-19-9.
- Dias, D., Vigário, A., Pinto, T., Reis, A., & Barroso, J. (2025). Smart Logistics Management for Electric Motorcycles. *The 2025 IEEE Vehicle Power and Propulsion Conference (IEEE VPPC 2025)*. Hangzhou, China.

- Vigário, A. & Fernandes R. (2025). *Real-time Electric Motorbike Route Optimization for Urban Service Provision*. Comunicação apresentada no Workshop on Optimization, Dynamics and Convex Analysis (WODCA 2025). Aveiro, Portugal.
- Vigário, A., Oliveira, J., Fernandes, R., Pinto, T., Reis, A., Rocha, T., Barroso, J. (2026). Adaptive User Interface for Electric Vehicle Route Information in Urban Mobility Services. In: Tsihrintzis, G.A., Virvou, M., Marinakis, V., Jain, L.C. (eds.) *Human-Smart City Interactions and User-Citizen Experiences. Learning and Analytics in Intelligent Systems*, 58, 157-180. Springer, Cham. https://doi.org/10.1007/978-3-032-06364-9_8.

Apêndice B - Tabela síntese com as principais características das aplicações de navegação analisadas

É apresentado a **Tabela 13** com as principais características das aplicações de navegação estudadas nesta dissertação, tabela efetuada pela autora.

TABELA 13 - CARACTERÍSTICAS DAS APLICAÇÕES DE NAVEGAÇÃO

Características	Waze	Google Maps	Apple Maps	Maps.me	Here WeGo
Foco principal	Trânsito e rotas em tempo real. UX/UI	Navegação geral, pesquisa de locais. Explorar e navegar. UX	Navegação com integração com o ecossistema Apple. Aparência. UI	Mapas <i>offline</i> , viagens e trilhas	Navegação <i>offline</i> e planeamento de viagens
Mapas offline	Não	Sim (pode fazer <i>download</i>)	Sim. Pode efetuar <i>download</i> dos mapas ou área desejada e renomeá-los.	Baseados no OpenStreetMap ¹ , uma plataforma colaborativa que oferece dados detalhados e frequentemente atualizados por utilizadores de todo o mundo.	Permite fazer <i>download</i> de mapas de regiões, países ou continentes inteiros, baseados em uma plataforma de dados de localização própria.
Utilização offline	Precisa de internet para otimizar rotas e atualizar informações, mas pode continuar navegando em modo <i>offline</i> se a rota já foi carregada.	O app deixa de mostrar alertas de radares e acidentes, mas ainda permite ver toda a rota. Pode ser feito a rota na <i>web</i> , guardar e utilizar na <i>app</i> do telemóvel.	Pode obter detalhes adicionais sobre locais como direções, informações de contato, horários de funcionamento e muito mais.	Sim, é sua funcionalidade. Funciona a partir do GPS.	Sim, é sua funcionalidade. Funciona a partir do GPS.

1. O **OpenStreetMap (OSM)** é um **projeto colaborativo e de código aberto** que cria e mantém um **mapa digital gratuito e editável do mundo**, construído por uma comunidade global de voluntários. Site: <https://www.openstreetmap.org/#map=15/41.28597/-7.72648>

Características	Waze	Google Maps	Apple Maps	Maps.me	Here WeGo
Interface (visualização)	Simples, centrada na navegação e situações em tempo real	Simples	Simples	Simples, centrada em rotas diretas, sem otimização.	A aplicação possui uma interface moderna e intuitiva.
Personalização de rotas	Sim, guarda os destinos. Ao criar rota para saída em determinado horário, apresenta a hora de chegada de acordo com o trânsito referente ao momento da saída. Pode restringir rotas, como estradas de terra, para além das portagens e autoestradas.	Pode arrastar ponto da rota indicada de maneira a criar rota personalizada pela <i>web</i> . Guarda os destinos. Pode restringir rotas, como portagens e autoestradas.	Depois de adicionar e editar locais, pode criar rotas personalizadas. Apresenta várias rotas de acordo com o tempo/distância. Pode evitar autoestradas e portagens.	Pode escolher entre o tempo que falta ou a hora que chega. Evita estradas com portagens, estradas não pavimentadas.	Pode personalizar rotas adicionando paragens intermédias e ajustando preferências, como evitar portagens, autoestradas ou travessias em meio aquático.
Trânsito	Sim, indica.	Atualizações de trânsito em tempo real - sobre congestionamento, acidentes, corte de estradas e radares móveis. Pode manter os outros condutores informados ao sinalizar incidentes.	Mapa com indicação de trânsito é apresentado em tempo real.	Como é um mapa <i>offline</i> , não apresenta situações de trânsito.	Apresenta informações de trânsito em tempo real.

Características	Waze	Google Maps	Apple Maps	Maps.me	Here WeGo
Indicação de sentido navegação	Apresenta os próximos passos a serem seguidos.	Aviso atempado para aproximação de zonas perigosas, como curvas fechadas. Avisos de mudanças de direção simples.	Ao aproximar-se de um cruzamento complexo, o mapa altera para uma perspetiva 3D ao nível da estrada para simplificar o que está à frente.	A indicação de sentido e direção passo a passo, apresenta orientação com base na posição GPS. Possui algumas limitações, não apresenta mudança de faixa ou intersecções detalhadas.	Fornece instruções detalhadas.
Sinalização vertical	Não indica	Mostra localização de faixas de pedestre, sinalização e semáforos.	A exibição de sinalização está sendo implementada progressivamente e pode variar de acordo com a cidade ou país. Recursos como semáforos e STOP são mais comuns em locais onde a Apple lançou mapas redesenhados com mais detalhes.	Não exhibe sinalização vertical detalhada, como placas de trânsito, limites de velocidade ou avisos de direção obrigatória.	Exibe sinalizações verticais, como sinais de trânsito, diretamente no mapa.
Alertas em tempo real	Sim – <i>Crowdsourcing</i> , apresenta radar, acidentes, obras, polícia, entre outros.	Atualizações em tempo real sobre congestionamento, acidentes, corte de estradas e radares móveis, através da comunidade Waze.	Apresenta fluxo de trânsito, acidentes, fecho de vias e obras.	Como os mapas são <i>offline</i> , podem não refletir mudanças recentes, como corte de ruas, a menos que sejam atualizados manualmente.	Apresenta alertas sobre condições do trânsito, incluindo acidentes, obras e outras situações que afetem a rota.

Características	Waze	Google Maps	Apple Maps	Maps.me	Here WeGo
Alertas velocidade	Sim, ele emite alerta quando ultrapassado limite de velocidade local	Não apresenta grande alerta quando ultrapassado o limite.	Apresenta o limite de velocidade da via, mas não apresenta velocidade do veículo nem emite alertas de excesso de velocidade.	Apenas quando indicado radares pelos utilizadores, manualmente. Nesse caso, emite alerta quando acima do limite. Não apresenta radar fixo.	Exibe os limites de velocidade das vias e permite que configure-se alertas para notificar quando em excesso e o limite de velocidade do veículo.
Dados móveis	Necessidade constante	Necessário para alertas em tempo real.	Necessário para alertas em tempo real.	Não utiliza, utiliza sinal de GPS.	Necessário para alertas em tempo real. <i>Offline</i> utiliza GPS.
Navegação motociclos	Sim, oferece rotas otimizadas para motos, considerando vias menores e diferentes padrões de trânsito.	Sim, em alguns países, como Brasil, em países da Ásia e Índia. Esse recurso ainda não está amplamente disponível na Europa e nos EUA, mas pode ser expandido no futuro.	Não possui rotas específicas para motociclos.	Não possui rotas específicas para motociclos.	Oferece uma opção de navegação específica para motocicletas, mas a rota é criada com intuito de passeio, não apresenta a melhor rota tempo/distância.
Pontos de interesse	Apresenta, estabelecimentos, postos de abastecimento/carregamento, estacionamento.	Sugestões de pontos de interesse que estejam na rota.	Sugestões de pontos de interesse que estejam na rota e parques de estacionamento.	Possui a indicação de pontos de interesse inseridos manualmente pelos utilizadores, que podem proceder a essas atualizações.	Destaca pontos de interesse próximos, como restaurantes, postos de combustível, entre outros.

Características	Waze	Google Maps	Apple Maps	Maps.me	Here WeGo
Navegação veículos elétricos	Parcial, não apresenta rotas exclusivas, mas ao selecionar tipo de combustível e <i>plug</i> apresenta estações de carregamento, mas não possui um modo específico para EVs.	Mostra estações de carregamento, disponibilidade em tempo real e tipo de conector compatível. Permite filtragem por tipo de carregador.	Apresenta criação de rotas exclusivas para carros elétricos. Pode avaliar a carga, a elevação do percurso e os pontos de recarga compatíveis ao longo da rota.	Não oferece funcionalidades específicas para veículos elétricos, como a localização de estações de carregamento ou planeamento de rotas que considerem pontos de recarga.	Mostra a localização de pontos de carregamento ao longo da rota e no destino.
Navegação por voz	Sim, podendo alterar a voz. apresenta uma explicação +/- detalhada, como "Vire à direita logo após o posto de abastecimento".	Tem apenas dois modelos de voz em português e apresenta uma informação simples, como: "a 300 metros vire a direita".	Tem vários modelos de voz e apresenta uma informação antecipada da navegação, como: "Passe este semáforo e então vire à direita na próxima rua em 300 metros".	A funcionalidade permite que os utilizadores recebam instruções de direção por áudio. Possui em vários idiomas.	Oferece orientação por voz em vários idiomas.
Cursor	Mais rápido, a seta segue precisamente a direção do veículo. Possibilidade de alterar o aspeto do cursor.	O cursor move-se muito e continua a ajustar-se o tempo todo.	O cursor segue precisamente a direção do veículo.	Indica a direção por GPS, por vezes o cursor sai da trajetória.	Apresenta um cursor claro que indica a posição atual no mapa.

Características	Waze	Google Maps	Apple Maps	Maps.me	Here WeGo
Otimização de rotas	Calcula diferentes rotas e apresenta o tempo/ distância, tendo em conta estados da via, trânsito, entre outros. Apresenta as rotas no mapa.	Ao inserir um ponto de destino, o app automaticamente calcula a distância e os trajetos mais rápidos.	O planeamento de rotas oferece previsões. Em comparação com o Google maps, efetua melhor otimização de rota.	A criação de rotas é feita no momento de pesquisa, com indicação de tempo de viagem, km e pagamentos. Quando feito desvio, atualiza a nova trajetória. Na verdade não efetua otimização de rotas, apenas apresenta as opções de trajetos para as rotas escolhidas.	Calcula automaticamente a rota mais eficiente para o destino escolhido, considerando fatores como condições de trânsito em tempo real e preferências.
Estado do tempo	Exibi informações sobre o estado do tempo durante a navegação.	Apresenta dados atualizados.	Apresenta o índice qualidade do ar ou condições do tempo.	Não indica.	Fornece informações meteorológicas e temperatura.
Usabilidade	Simples e intuitiva	Escolha de tipo de mapa, padrão, satélite ou relevo. Detalhes do mapa, trânsito, etc.	<i>Street view</i> e <i>3D option</i> , com a vista de satélite têm uma visão foto realista. Integração com <i>Apple watch</i> .	Simples e usabilidade prática.	Escolhe tipo de mapa, padrão, satélite ou relevo.
Plataformas disponíveis	iOS, Android,web	iOS, Android,web	iOS, MacOS	iOS, Android	iOS, Android,web

Características	Waze	Google Maps	Apple Maps	Maps.me	Here WeGo
Acessibilidade	Pode personalizar as configurações da aplicação, ajustando a visualização do mapa, as instruções de voz e as preferências para melhor atender às necessidades e conforto individuais. Dispositivos ativados por voz, como Alexa² ou Siri³, podem ser utilizados para interagir com o Waze com o sistema de mãos livres.	Modo de alto contraste para utilizadores com deficiência visual. Informações sobre acessibilidade para deficientes em locais públicos. Modo de ecrã ampliado para facilitar a leitura durante a navegação.	O Modo Escuro reduz o brilho da tela e melhora a visibilidade, sendo útil para pessoas com sensibilidade à luz. Compatível com Ajustes de Acessibilidade do iOS, permitindo aumentar fontes e melhorar contraste para facilitar a leitura. Incorporados ao iPhone ou Ipad.	Não possui comandos de voz, pode alterar certas configurações da <i>app</i>, como, aparência e tamanho da letra.	Opções de contraste e modos de exibição que melhoram a legibilidade e a interação.
Immersive view	Em desenvolvimento	Na Europa está disponível em Paris. É possível visualizar, durante a navegação ou na planificação de uma viagem, o aspeto dos edifícios, estádios, parques e estradas, gerados de forma foto realista pela utilização de imagens reais, IA e visão computacional	3D <i>Option</i> – com dois dedos pode inclinar a visão em qualquer ângulo e pode girar para qualquer direção. Apresenta <i>street view</i> enquanto navega, para melhor perceção do local.	Não possui.	Não possui. Visualização 3D em desenvolvimento

2.Alexa: Assistente virtual com inteligência artificial desenvolvida pela Amazon, controlado principalmente por voz. <https://alexa.amazon.com/about>.

3.Siri: Assistente virtual com inteligência artificial desenvolvida pela Apple, que permite aos utilizadores interagir com dispositivos através de comandos de voz. <https://www.apple.com/br/sir>

Características	Waze	Google Maps	Apple Maps	Maps.me	Here WeGo
IA(interação com o utilizador/ conversação)	Com o apoio do Gemini ⁴ , a aplicação pode fornecer informações utilizando linguagem natural, de forma intuitiva e conversacional.	Ajuda mãos-livres com o Assistente Google O Assistente Google já está integrado no Google Maps para que possa enviar mensagens, fazer chamadas, ouvir música e obter ajuda mãos-livres enquanto conduz. Diga "Ok Google" para começar a utilizar.	Pode comunicar incidentes utilizando a Siri. Diga algo como: "Ei, Siri. Quero comunicar um acidente" ou "Ei, Siri. Está alguma coisa na estrada."	Não existem informações específicas que indiquem a utilização de recursos de inteligência artificial.	Não existem informações específicas que indiquem a utilização de recursos de inteligência artificial.
Principais Recursos com IA	Otimização de Rotas, Previsão de Trânsito, Detecção de Incidentes, Sugestão de Rota alternativa, Economia de Combustível, <i>Immersive View</i> .	Otimização de Rotas, Previsão de Trânsito, Detecção de Incidentes, Sugestão de Rota alternativa, Economia de Combustível, Immersive View.	Não possui recursos de Inteligência Artificial amplamente divulgados ou integrados de forma evidente em suas funcionalidades. No entanto, a Apple tem investido significativamente em IA em outras áreas de seu ecossistema, como a Apple Intelligence, em desenvolvimento.	Embora ofereça funcionalidades avançadas, não há detalhes claros sobre a implementação de recursos específicos de Inteligência Artificial.	Embora ofereça funcionalidades avançadas, não há detalhes claros sobre a implementação de recursos específicos de Inteligência Artificial.
Tempo de resposta	Tempo real	Tempo real	Tempo real	Como utiliza GPS, só depende do sinal.	Rápido e responsivo.

4. Gemini: Modelo de inteligência artificial desenvolvido pela Google. <https://gemini.google/about/>

Apêndice C – Diagrama entidade-relacionamento base de dados

No modelo de base de dados foram identificadas e modeladas as principais entidades do sistema, destinadas a suportar relações do tipo muitos-para-muitos (N:N). As ligações entre as tabelas foram definidas através de chaves primárias e estrangeiras, respeitando os princípios da normalização, com o objetivo de evitar redundância de dados e garantir a integridade referencial, como apresentado na **Figura 30**.

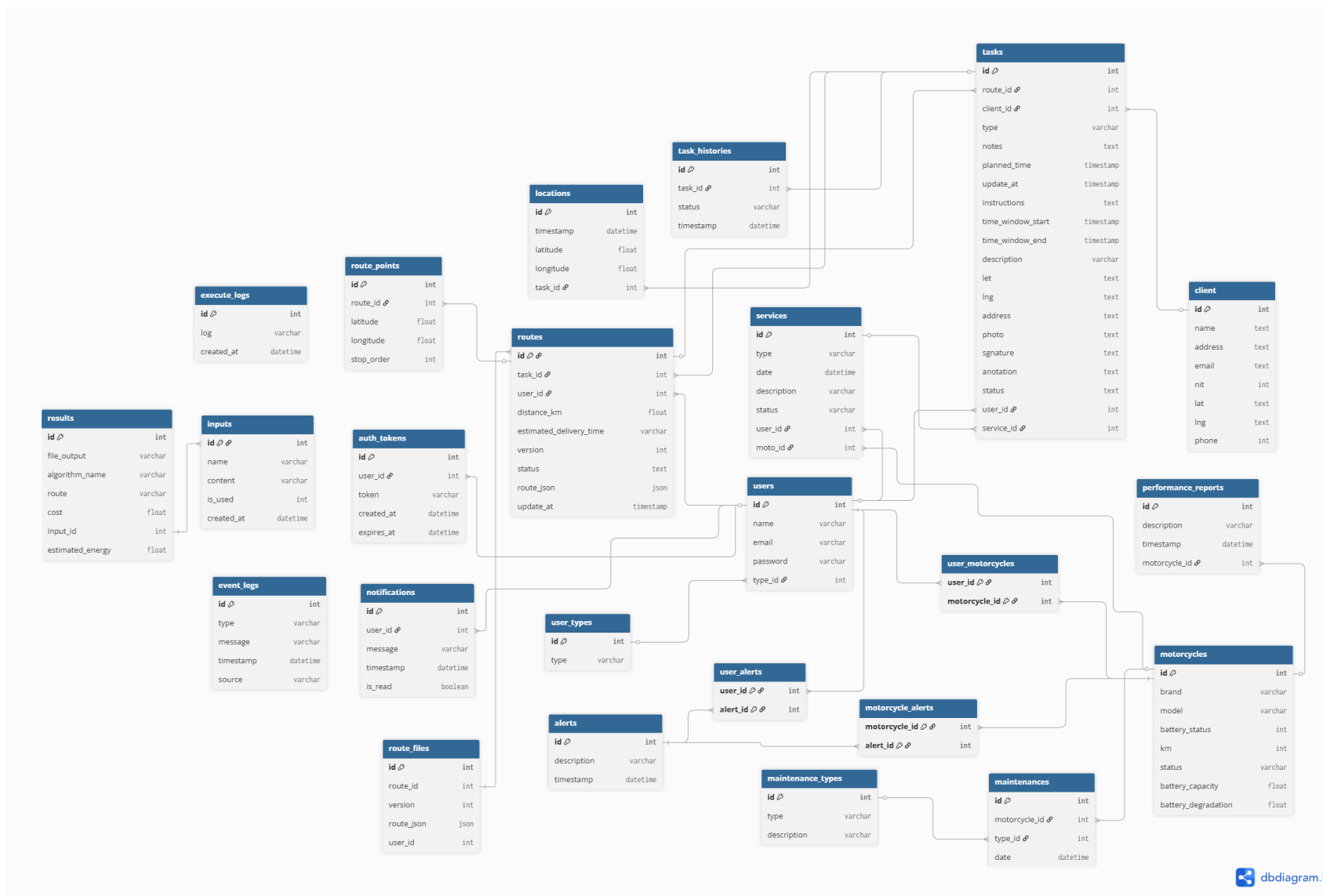


Figura 30 - Diagrama entidade-relacionamento

Apêndice D – Protótipo funcional e navegável da aplicação. Layouts criados:

De seguida são apresentados os mockups desenvolvidos na ferramenta online Glide, utilizados para estudar as diferentes formas de navegação da aplicação, Figuras de 31 a 38.



Figura 31 - Ecrã inicial

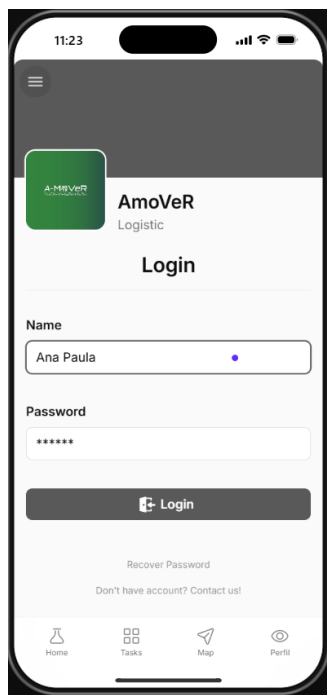


Figura 32 - Ecrã login

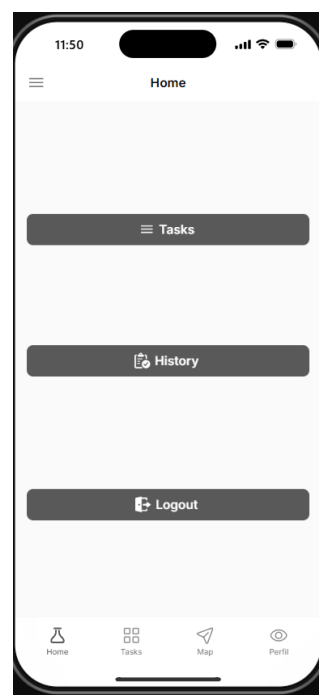


Figura 33 - Ecrã home

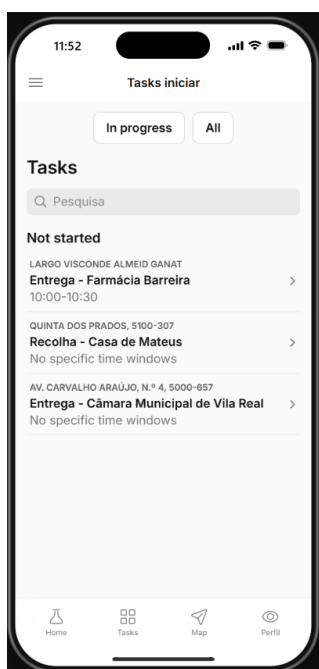


Figura 34 - Ecrã tarefas

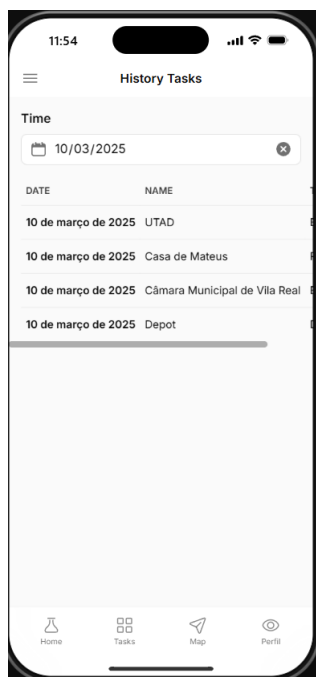


Figura 35 - Ecrã histórico

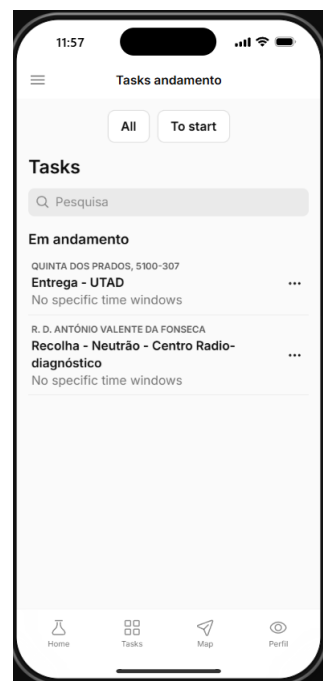


Figura 36 - Ecrã detalhe tarefa



Figura 37 - Ecrã navegação

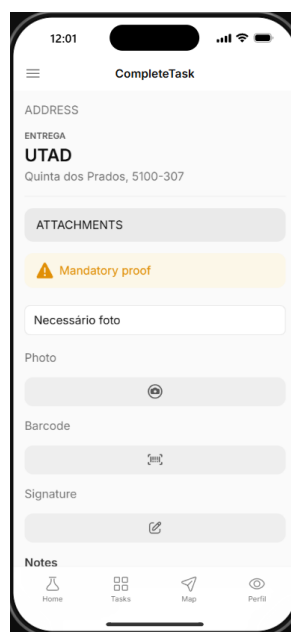


Figura 38 - Ecrã comprovativo

Apêndice E – Testes funcionais - Caso de estudo

Os testes funcionais foram elaborados a partir da utilização do emulador do Android Studio, utilizando vários modelos de dispositivos, através de um caso de estudo. De seguida são apresentados detalhadamente os testes realizados.

E.1 Autenticação

Este teste permite validar a autenticação e verificação do utilizador. Este teste tem associado o utilizador, amover@amover.pt. Este teste é caracterizado pelos seguintes testes:

1. Abrir a aplicação, inserir os dados do utilizador e carregar no botão do login;
2. Verificar que a autenticação ocorre esperado, senão tentar novamente;
3. Verificar a transição para a página inicial.

O utilizador abre a aplicação e é apresentado o ecrã de autenticação. Caso não seja introduzido qualquer dado nos campos obrigatórios, o sistema assinala-os a vermelho, solicitando ao utilizador a respetiva introdução antes de prosseguir (**Figura 39**). Caso algum dado seja inserido incorretamente, surgirá um aviso para que o utilizador verifique os dados. Dados corretos, o utilizador acederá à página principal, Home (**Figura 40**).

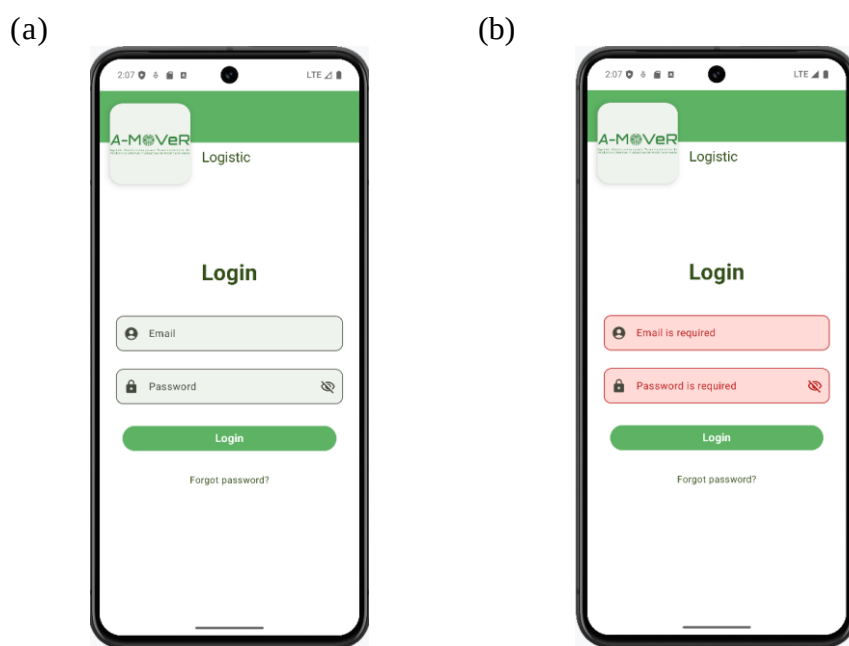


Figura 39 - (a)Layout login -> (b) Inexistência de credenciais inseridos

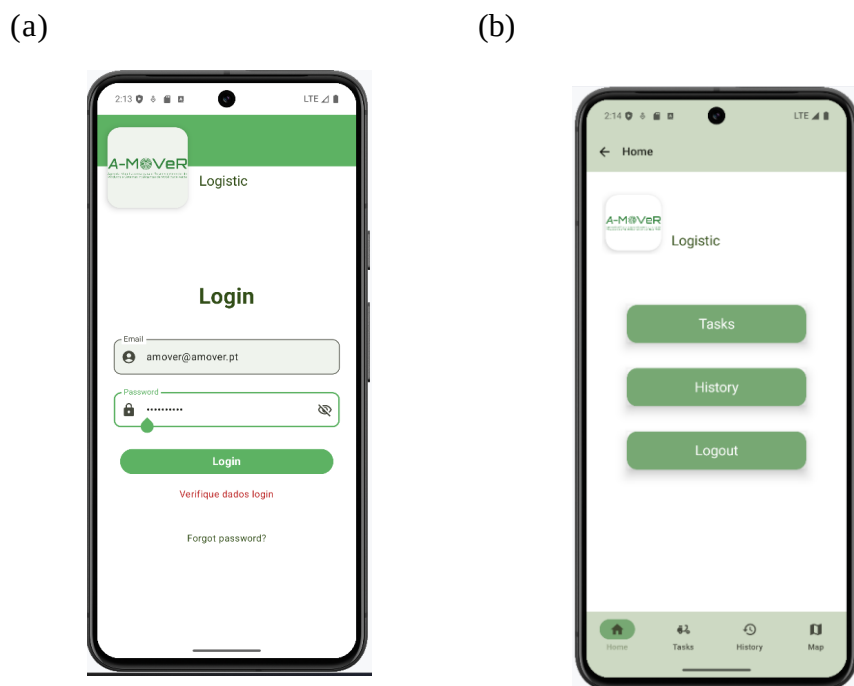


Figura 40 - (a)Layout login -> (b) Página inicial

Validando, assim, o requisito funcional de autenticação e sessão, efetuando o *login* do utilizador, mantendo a sessão ativa e efetuando o *logout*.

E.2 Consultar rota do dia e tarefas

Este teste serve para validar as funcionalidades associadas ao requisito funcional 2. Este teste será efetuado com a sessão do utilizador, mostrando a rota do dia associado ao utilizador amover@amover.pt, definido na base de dados com “Ana Vigário” (**Figura 41**). Este teste é caracterizado pelos seguintes passos:

1. Com a aplicação aberta e o utilizador autenticado, verifica a rota do dia e as tarefas associadas;
2. Verifica detalhes da tarefa e altera o seu estado.

☐	id int4	name text
☐	1	Ana Vigário
☐	2	Octávio Santos
☐	3	Rodrigo Soares

Figura 41 - Utilizadores-condutores

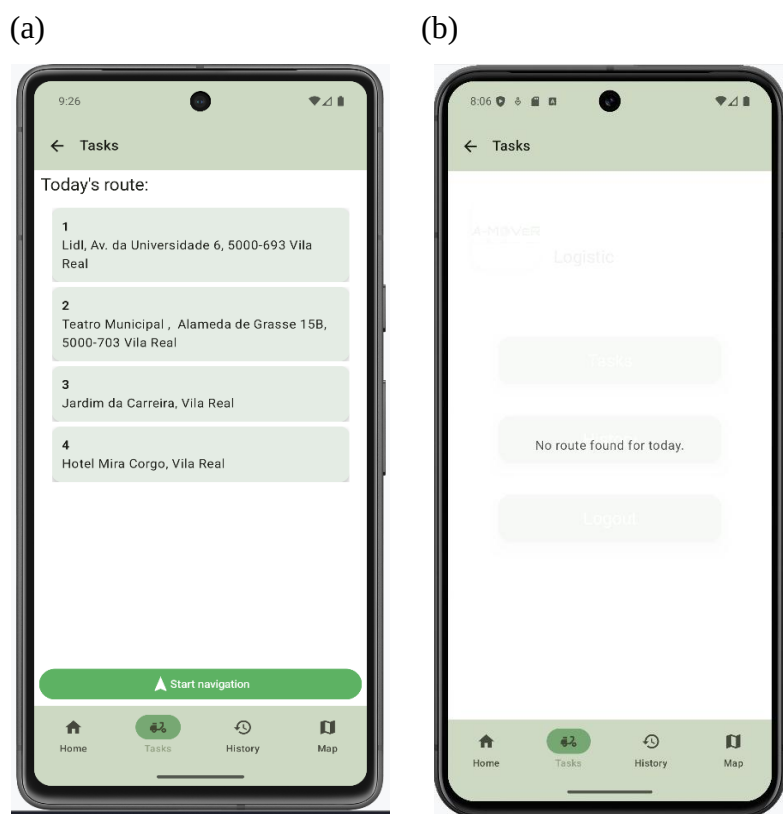


Figura 42 - (a) Interface rota diária (b) Interface sem rota

Depois de efetuado a autenticação, na *interface* de apresentação de rotas o utilizador poderá consultar as rotas e os detalhes das tarefas associadas a rota do dia. O condutor abre a aplicação, no ecrã Home seleciona “Tasks” sendo apresentado a rota do dia associada ao utilizador autenticado. Caso não exista rota associada ao utilizador, em fluxo alternativo, é apresentado a mensagem “No route found for today” (Figura 42).

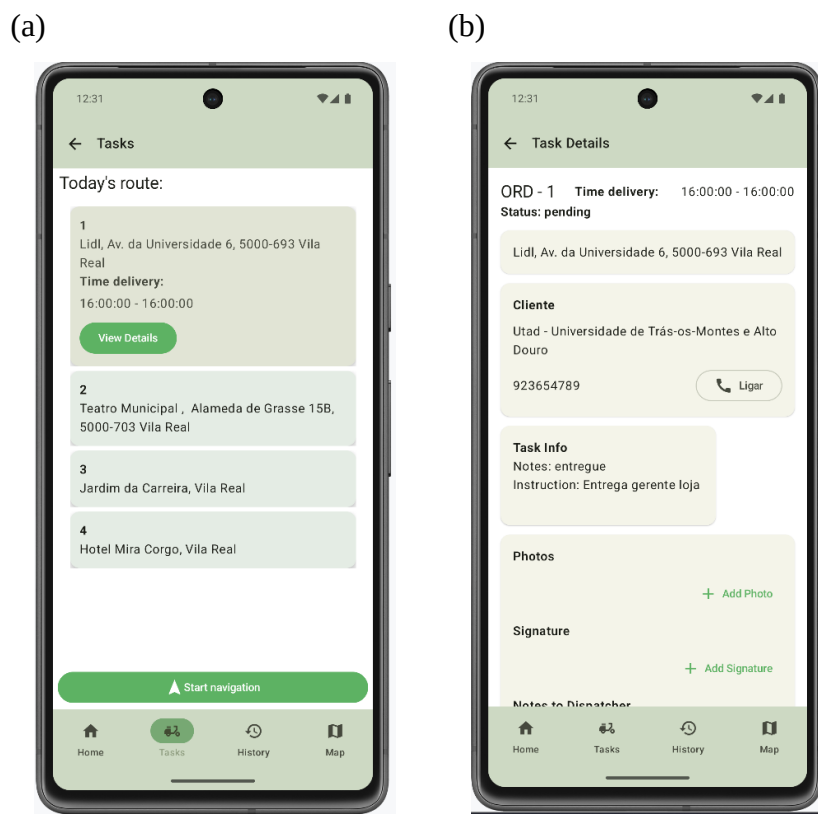


Figura 43 - (a) *Interface card* expandido (b) *Interface* Detalhe da tarefa

Ao seleccionar uma tarefa, o *card* será expandido e apresentado o botão “*View detail*”. Sendo seleccionado, abrirá nova *interface* com os detalhes da tarefa, ordem da tarefa, morada de serviço, nome cliente, entre outros dados inerentes ao serviço (**Figura 43**).

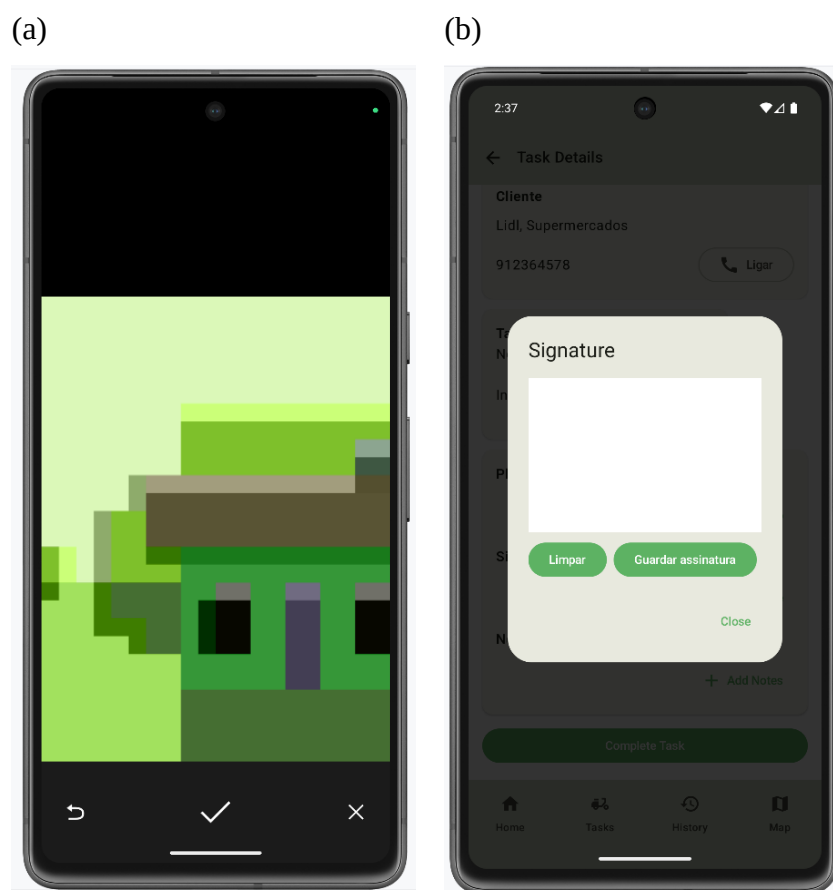


Figura 44 - (a) *Interface câmara fotográfica* (b) *Interface assinatura*

Nas **Figura 44** podem serem visualizados a câmara fotográfica, quando acionado “+ *Add Photo*” e acionado “+ *Add Signature*”. Ao seleccionar o botão “*Complete Task*”, abrirá um *pop-up*, de alteração de estado da tarefa (**Figura 45**).

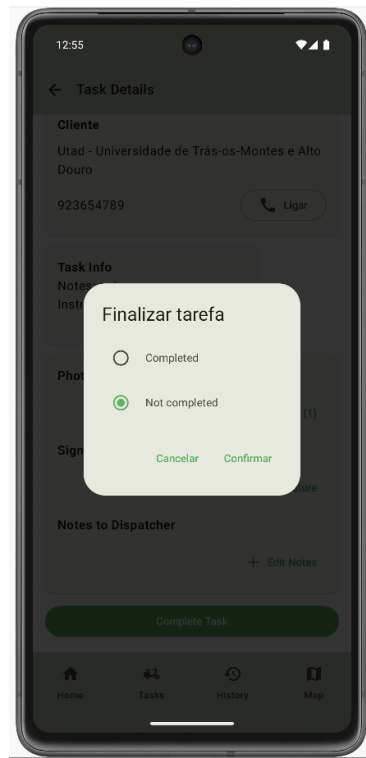


Figura 45 - Interface *pop-up* mudança de estado tarefa

Na *interface* de detalhe da tarefa é possível completar a tarefa, alterando assim o seu estado na base de dados, em “*completed*” ou “*no-completed*”, retornando à rota do dia. No detalhe da tarefa é apresentado os campos para preenchimento, obrigatórios, de acordo com a política da entidade prestadora do serviço.

Este teste foi realizado com sucesso, visto ser possível visualizar detalhes da tarefa associada, atualizar estado da tarefa, registar ocorrências e prova de entrega.

E.3 Rota diária e otimização de rotas

Este teste serve para validar a apresentação da rota diária e sua otimização, associadas ao requisito funcional 3. Este teste é caracterizado pelos seguintes passos:

1. Com a aplicação aberta e o utilizador autenticado navega até à página de tarefas;
2. Quando alterado do estado é apresentado nova rota atualizada;
3. Verificar se os dados são apresentados corretamente, conforme contexto do utilizador.

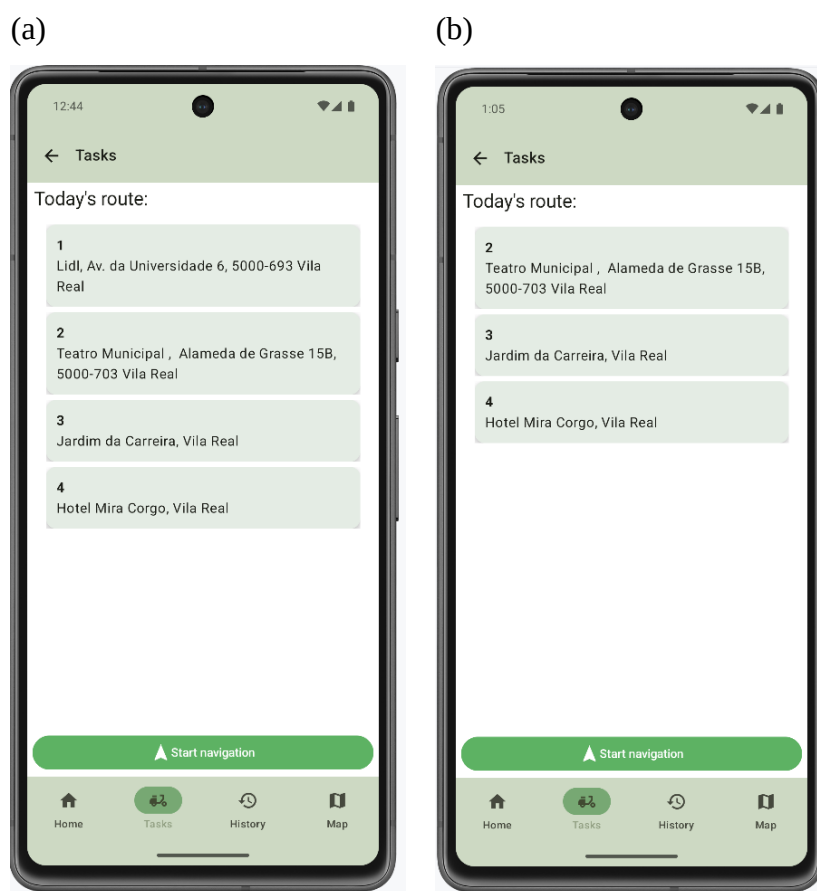


Figura 46 - (a) Interface rota inicial (b) Interface rota “otimizada”

Ao alterar o estado da tarefa, é automaticamente apresentada uma nova rota otimizada, enviada pela API. **(Figura 46)**. Este teste foi realizado com sucesso, visto ser possível visualizar a rota diária associada ao condutor, após atualização do estado da tarefa, ser apresentado nova rota otimizada.

E.4 Integração com serviços de mapas.

Este teste permite validar as funcionalidades associadas ao requisito funcional 4. Para este teste foi simulado a rota com saída e chega na Escola das Ciências e Tecnologia da UTAD, como depósito de saída e chegada. Este teste é caracterizado pelos seguintes passos:

1. Com a aplicação aberta e o utilizador autenticado, acede a página de tarefas;

2. Com a aplicação na página da rota diária, inicia-se navegação com pré visualização.

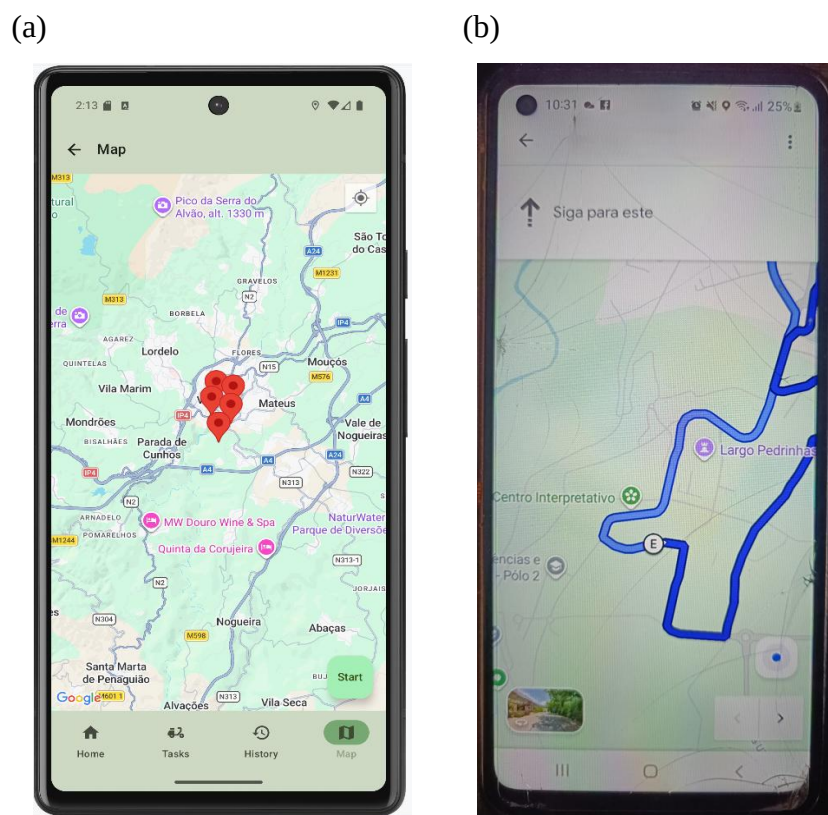


Figura 47 - (a) Interface visualização prévia (b) Interface navegação

Com o utilizador autenticado e navegado para a página de tarefas é apresentado a visualização prévia da rota. Acedendo ao botão “*Start navigation*”, apresentará a rota no mapa para navegação. Apresentado a visualização prévia da rota, e com o Google Maps instalado no dispositivo, pode iniciar a rota no Google Maps inserido na aplicação (**Figura 47**). Teste executado com sucesso é visualizado as rotas otimizadas na aplicação de navegação.

E.5 Histórico e filtros

Este teste serve para validar a apresentação do histórico e seus filtros, associadas ao requisito funcional 5. Este teste é caracterizado pelos seguintes passos:

1. Com a aplicação aberta e o utilizador autenticado, acede a página “History”;
2. Seleciona *card* para visualização dos detalhes da tarefa;
3. Selecionar filtro;
4. Verificar que a informação está conforme o esperado.

Neste seguimento, foi criada uma *interface* para que o utilizado pode visualizar as tarefas efetuadas (**Figura 48**). Foi criado, igualmente, uma *interface* de detalhe da tarefa, quando seleciona uma tarefa, o *card* expande-se onde pode selecionar o botão “View detail” e visualizar os dados da tarefa.

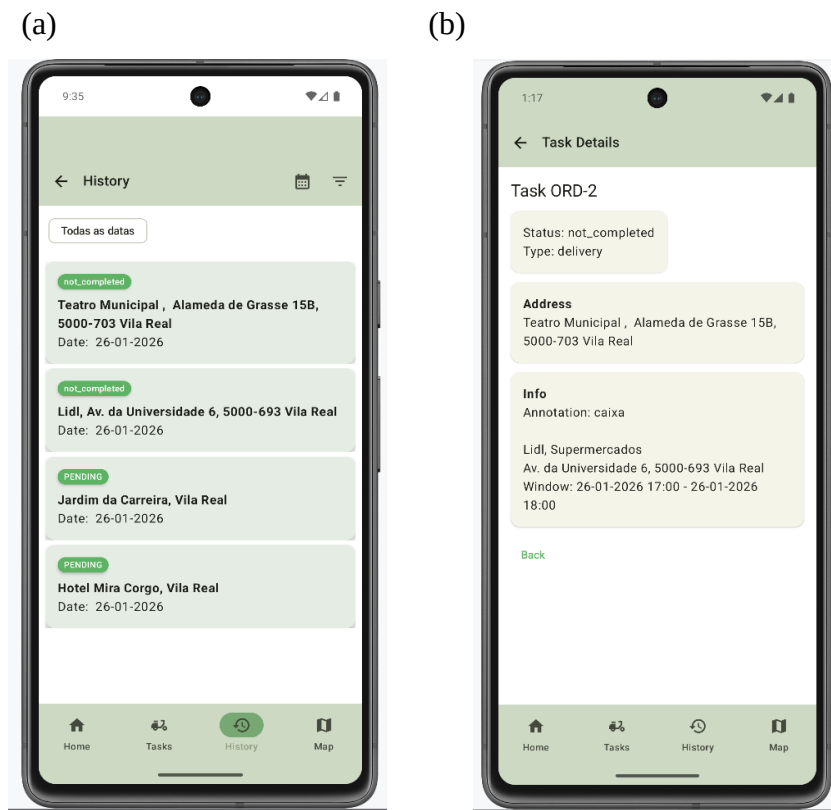
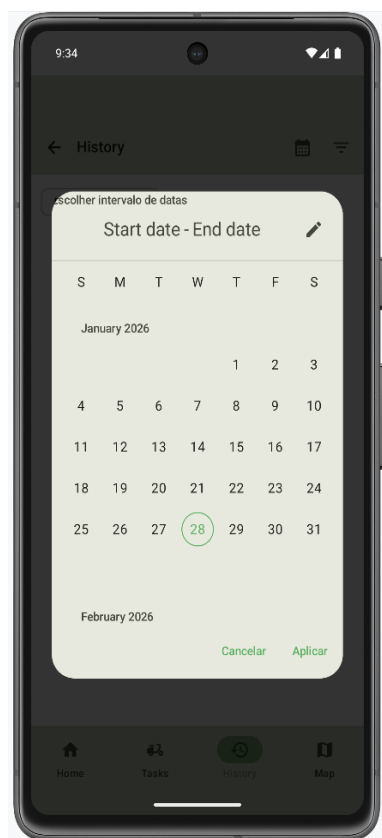


Figura 48 - (a) *Interface* histórico (b) *Interface* detalhe tarefa - histórico

Na *interface* histórico é apresentado as tarefas por ordem cronológica, existindo na parte superior direita botões para filtragem por data, onde selecionando é apresentado as tarefas realizadas nessa data que, neste caso, será o dia 28 de janeiro de 2026 (**Figura 49**).

(a)



(b)



Figura 49 - Interfaces (a) histórico (escolha data) (b) histórico filtrado por data

Selecionando os outros tipos de filtro, será apresentado em primeiro o selecionado pelo filtro, como apresentado na **Figura 50**, onde foi escolhido o filtro de estado.

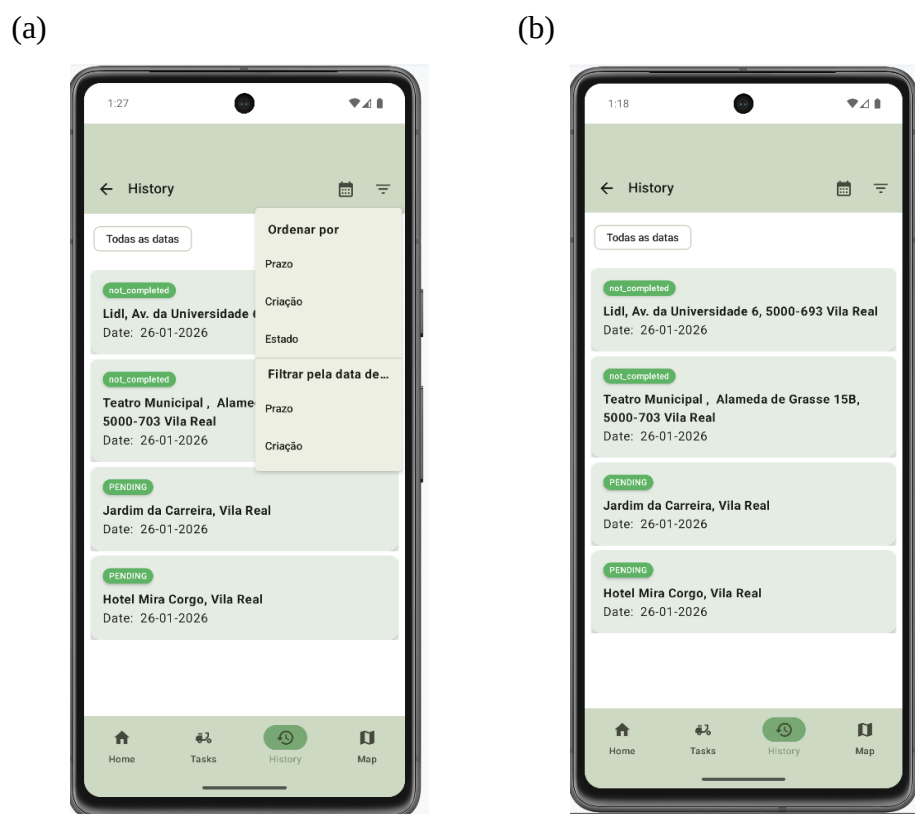


Figura 50 - Interfaces (a) histórico (escolha data) (b) histórico filtrado por data

Este teste foi realizado com sucesso. Com o acesso à página foi possível apresentar histórico de tarefas, visualizar detalhes da tarefa, aplicar filtros e visualizar listagem filtrada.

E.6 Apoio serviço

Este teste permite confirmar as funcionalidades do requisito funcional 6. Este teste é caracterizado pelos seguintes passos:

1. Com a aplicação aberta e o utilizador autenticado, acede a página “Task”;
2. Seleciona *card* para visualização dos detalhes da tarefa;
3. Seleciona o botão para efetuar a chamada;
4. Verificar se abre a área de chamada do dispositivo;
5. Retornar ao detalhe da tarefa onde seleciona para registo de ocorrências.

Na *interface* de detalhe da tarefa é apresentado um botão que permite realizar chamada telefónica ao cliente, caso necessário. Ao seleccionar este botão, é automaticamente aberta a aplicação de chamadas do dispositivo, possibilitando contato direto com o cliente (**Figura 51**).

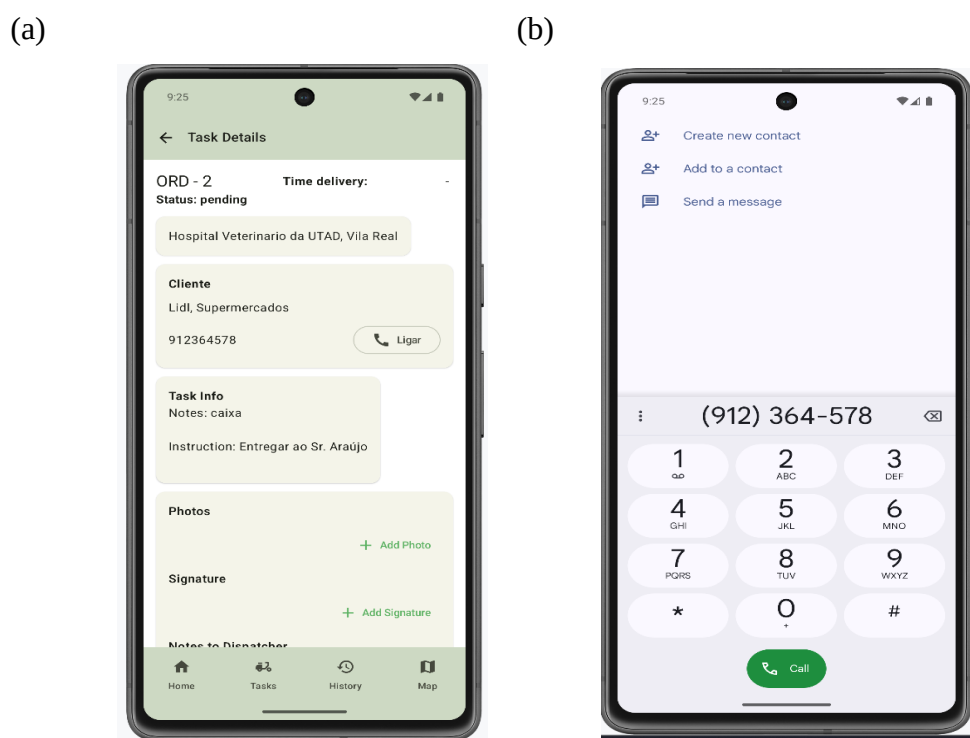


Figura 51 - *Interfaces* (a) detalhe da tarefa (b) Chamada ao cliente

Retornando a *interface* de detalhe de tarefa, pode-se efetuar o registo de ocorrências, quando acionado “+ *Edit Notes*” (**Figura 52**).

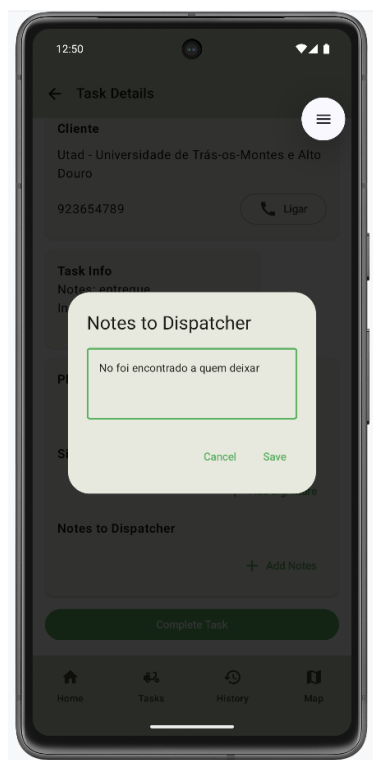


Figura 52 - Interface registar ocorrências

Este teste foi efetuado com sucesso. Com o acesso à página de detalhe da tarefa foi possível efetuar chamada ao cliente e preencher o registo de ocorrências.

E.7 Apresentação de notificações e mensagem

Este teste permite validar as funcionalidades associadas ao requisito funcional. Para sua execução, foi utilizado a ferramenta *online* Firebase para o envio de mensagens e notificações. Verificou-se que as notificações foram corretamente recebidas tanto com a aplicação em primeiro plano e mesmo que esta se encontrava encerrada, garantindo a receção contínua, conforme **Figura 53**.

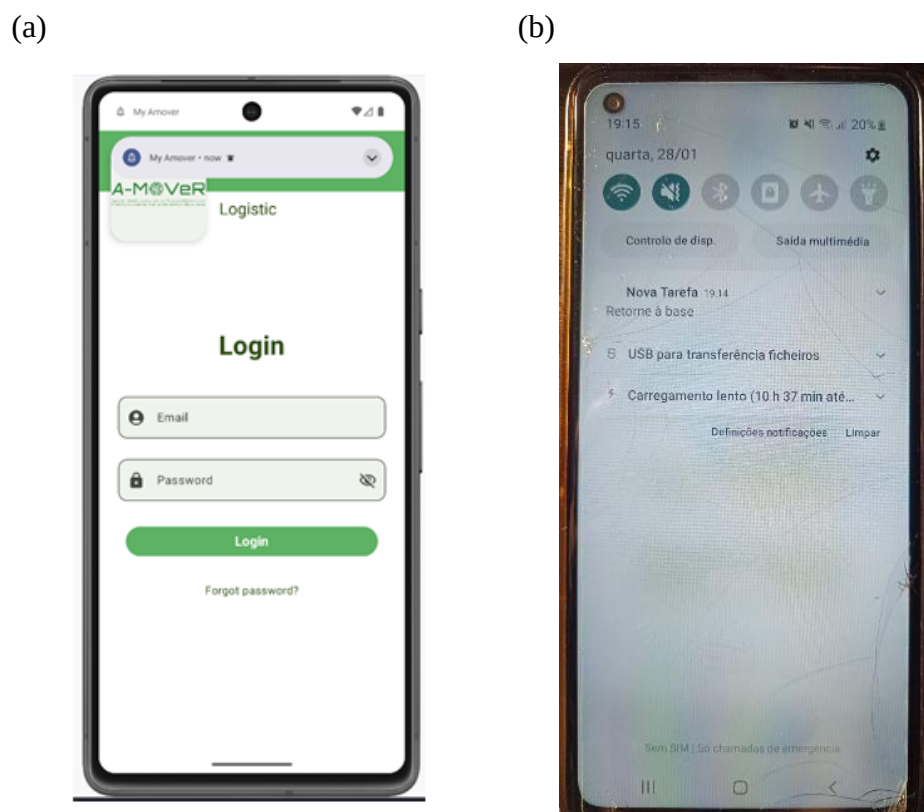


Figura 53 - Interfaces (a) notificação fora da aplicação (b) mensagem fora da aplicação

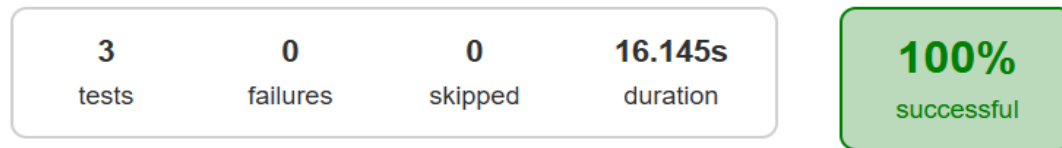
Este teste foi concluído com sucesso. As *interfaces* de apresentação de notificações e mensagens demonstram que a aplicação se encontra preparada para ser integrada com um sistema de envio de mensagens e notificações, quando ligada ao ecossistema A-Mover.

E.8 Testes instrumentados de funcionalidade

Os testes instrumentados foram utilizados para validar a navegação e os fluxos de interação do utilizador com a interface gráfica. Estes testes executaram-se em ambiente Android real e simulam ações com cliques, navegação entre ecrãs e encerramento de sessão. O objetivo principal foi confirmar que os *callbacks* de navegação são acionados corretamente, o grafo de navegação mantém consistência e a aplicação responde de forma estável a interações reais.

Class com.example.myamover.screens.HomeTest

all > com.example.myamover.screens > HomeTest



Tests

Test	Medium_Phone(AVD) - 16
clickHistory_callsCallback	passed (3.558s)
clickLogout_navigatesToLogin	passed (9.373s)
clickTasks_callsCallback	passed (3.214s)

Figura 54 - Resultados teste instrumentados de funcionalidades Home

O conjunto de testes executados sobre o ecrã principal da aplicação, **Figura 54**, apresentou taxa de sucesso de 100%, sem falhas ou teste ignorados, com duração total aproximada de 16,145 segundos. Estes testes incidiram sobre três interações críticas do fluxo principal da aplicação a navegação para o histórico, navegação para a rota diária e encerramento de sessão com retorno ao ecrã de autenticação.

Os resultados confirmam que o *callbacks*, associadas aos botões, são acionados corretamente e que o grafo de navegação mantém consistência durante a execução. O teste de navegação para o histórico e para a lista de tarefas apresenta tempos de execução semelhantes, refletindo interações simples e diretas. Já o teste de logout apresenta duração superior, o que é esperado, uma vez que envolve reconstrução do estado de navegação e limpeza da pilha de ecrãs, simulando um fluxo completo de encerramento de sessão.

A execução em ambiente instrumentado reforça que os comportamentos observados são representativos de utilização real, validando a robustez da interface e a estabilidade do sistema perante interações do utilizador. Estes testes asseguram que a aplicação responde de forma previsível a ações críticas, reduzindo o risco de regressões em atualizações futuras.

E.9 Compatibilidade

Este teste, apesar de averiguar um requisito não funcional, foi necessário para verificação de compatibilidade aos diferentes tamanhos de ecrãs. Apresentando adequadamente a informação em diferentes formatos, nas Figuras 55 a 58, efetuadas no emulador do Android.

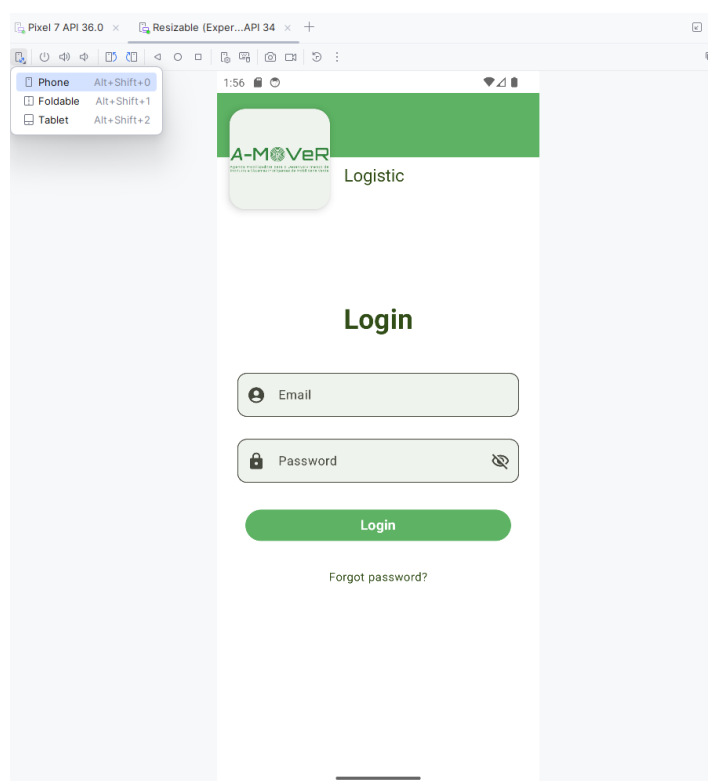


Figura 55 - Interface login em ecrã *phone*

A **FIGURA 55** apresenta a *interface* Login no emulador Resizable do Android Studio, em tamanho equivalente a um *smartphone*.

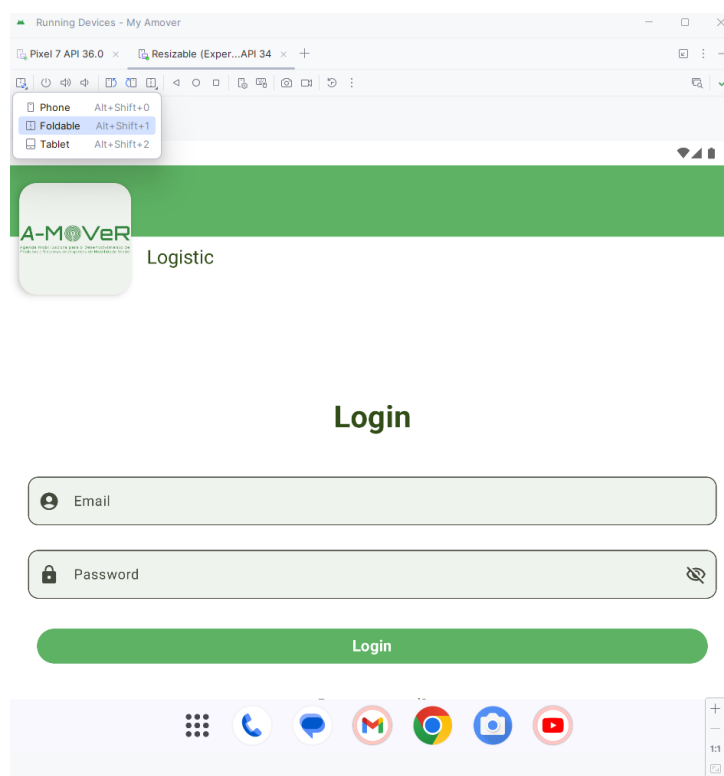


Figura 56 - Interface login em ecrã dobrável

A **Figura 56** apresenta a *interface* Login no emulador Resizable do Android Studio, em tamanho equivalente a um *smartphone* dobrável.

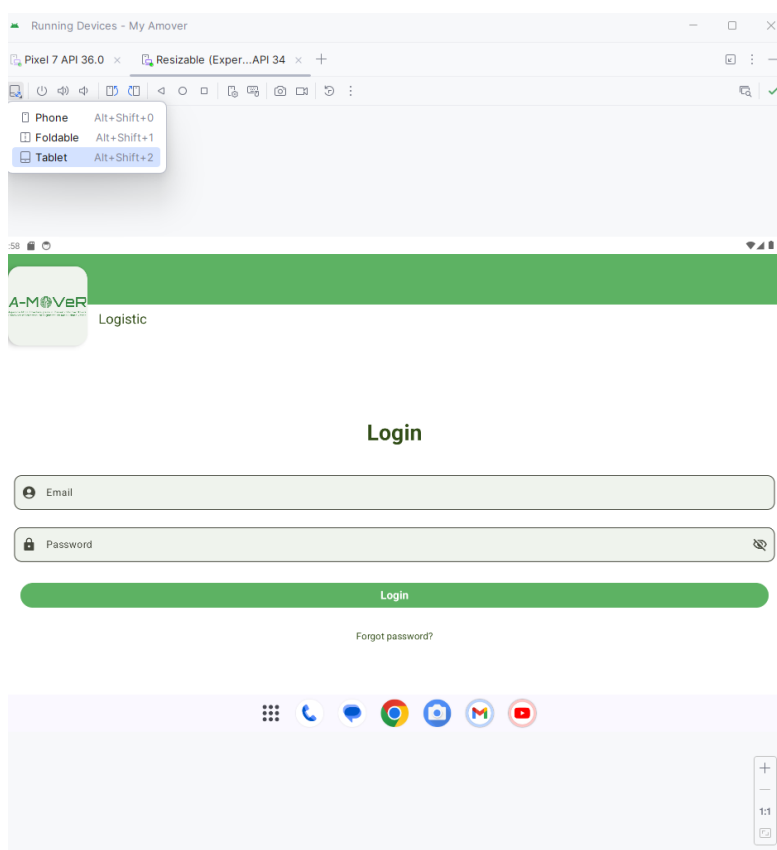


Figura 57 - Interface login em ecrã tablet

A **Figura 57** apresenta a *interface* Login no emulador Resizable do Android Studio, em tamanho equivalente a um *tablet*.

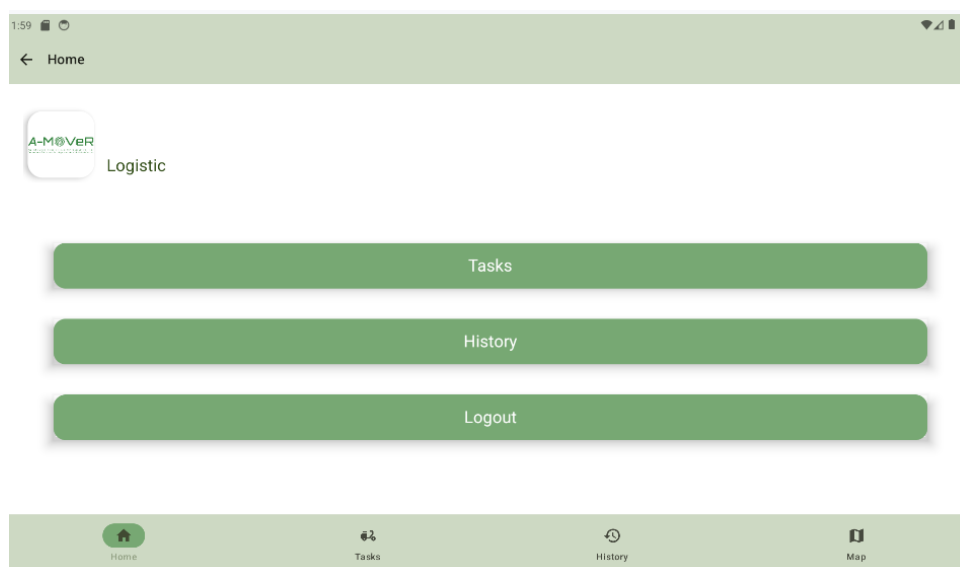


Figura 58 - Interface home em ecrã tablet

A **Figura 58** apresenta a *interface* Home no emulador Resizable do Android Studio, em tamanho equivalente a um *smartphone*. Este teste foi efetuado com sucesso, confirmando a compatibilidade da aplicação com diferentes dimensões e resoluções de dispositivos móveis.

Apêndice F – Testes de integração – APIs

Os testes de integração que foram realizados nos serviços estão diretamente relacionados à verificação correta da comunicação entre os diferentes serviços e as respostas.

F.1 Autenticação e histórico de tarefas

Os serviços de autenticação e histórico de tarefas estão associados a ligação à API do Supabase. Conforme evidenciado nas **Figura 59**, a comunicação com o *backend* é assegurada,

(a)

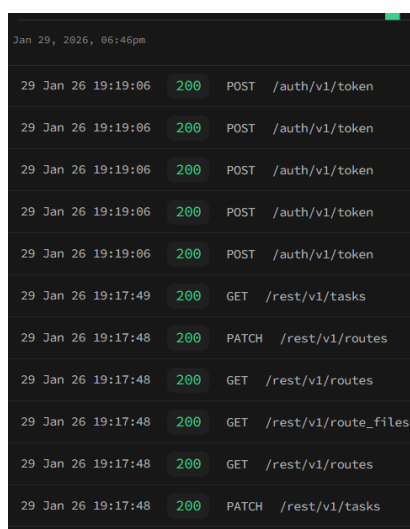
Jan 29, 2026, 04:45pm		
29 Jan 26 19:19:06	📢 INFO	Login
29 Jan 26 19:19:06	📢 INFO	/token request completed
29 Jan 26 19:19:06	📢 INFO	/token request completed
29 Jan 26 19:19:06	📢 INFO	Login

(b)

27 Jan 26 17:41:09	📢 INFO	/token request completed
27 Jan 26 17:41:09	📢 INFO	/token 400: Invalid login credentials

Figura 59 - API Gateway Supabase

Na **Figura 6059** são apresentados os registos de mensagens de log da aplicação durante a autenticação, confirmando conclusão dos pedidos `/token` e o correto funcionamento do mecanismo de *login*.



Jan 29, 2026, 06:46pm			
29 Jan 26 19:19:06	200	POST	/auth/v1/token
29 Jan 26 19:19:06	200	POST	/auth/v1/token
29 Jan 26 19:19:06	200	POST	/auth/v1/token
29 Jan 26 19:19:06	200	POST	/auth/v1/token
29 Jan 26 19:19:06	200	POST	/auth/v1/token
29 Jan 26 19:17:49	200	GET	/rest/v1/tasks
29 Jan 26 19:17:48	200	PATCH	/rest/v1/routes
29 Jan 26 19:17:48	200	GET	/rest/v1/routes
29 Jan 26 19:17:48	200	GET	/rest/v1/route_files
29 Jan 26 19:17:48	200	GET	/rest/v1/routes
29 Jan 26 19:17:48	200	PATCH	/rest/v1/tasks

Figura 60 - Registo das chamadas API Supabase durante processo de atualização do histórico

Na **Figura 60**, é apresentado a monitorização das requisições HTTP realizadas pela aplicação móvel à API Supabase durante os testes de integração, evidenciando chamadas de autenticação (`/auth/v1/token`) e operações *REST* sobre as tabelas tasks, onde se encontram os dados necessários para a criação do histórico de tarefas. Também fica evidenciado *REST* sobre as tabelas route e routes_files, fundamentais para a obtenção e construção das rotas utilizadas pela aplicação.

F.2 Otimização de rotas e alteração do estado da tarefa

As otimizações de rotas e alterações do estado das tarefas são efetuadas através da ligação entre a API local desenvolvida em FastAPI e a API do Supabase. Neste teste, tornou-se fundamental validar se a FastAPI acedia corretamente às informações disponibilizadas pelo Supabase, sendo, para este efeito, apresentados partes da *interface* Swagger da FastAPI, que evidenciam os *endpoints* responsáveis por estas operações.

```
GET /routes/today
```

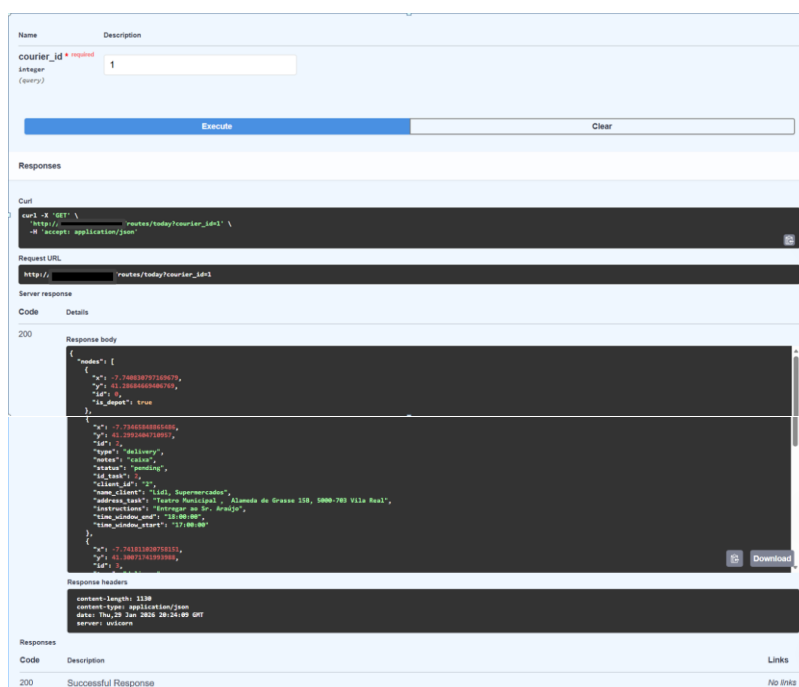


Figura 61 - Resposta do endpoint da API FastAPI - rotas

A **Figura 61** apresenta a resposta do *endpoint* FastAPI, apresentada na *interface* Swagger, contendo os dados das tarefas e respetivas coordenadas utilizadas no processo de otimização de rotas.

PATCH /tasks/{task_id}

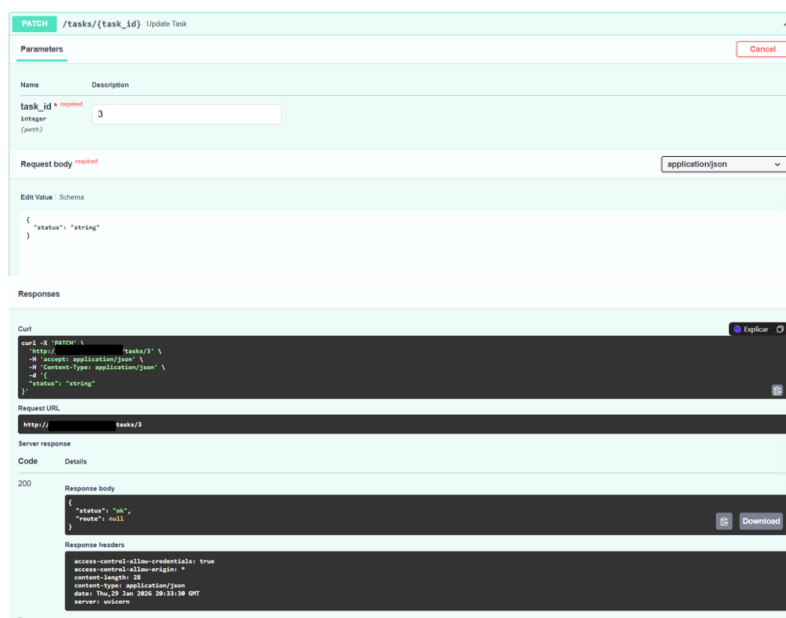
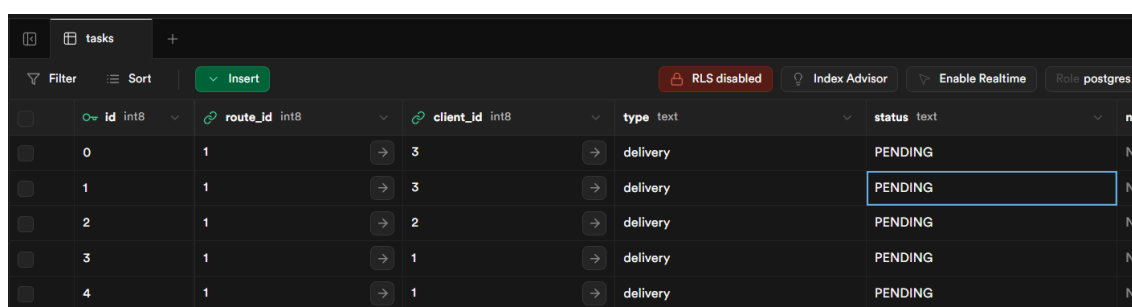


Figura 62 - Resposta do endpoint da API FastAPI - rotas

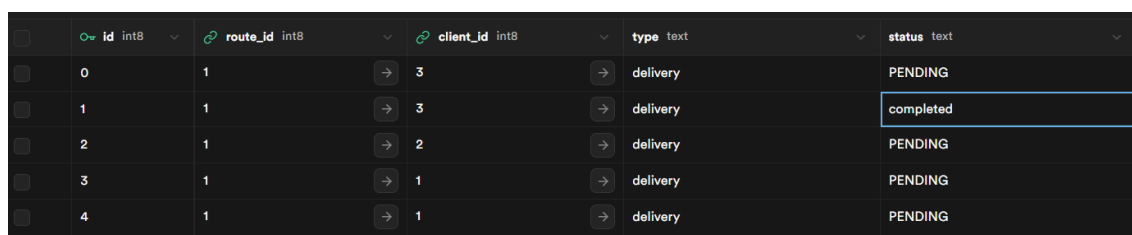
A **Figura 62** apresenta a documentação e execução do endpoint `PATCH /tasks/{task_id}`, disponibilizado pela API FatsAPI, responsável pela atualização do estado das tarefas. Através da *interface* Swagger, é possível observar os parâmetros necessários, bem como o corpo do pedido em formato JSON. A resposta do servidor, com código HTTP 200, confirma a execução bem-sucedida da operação, evidenciando a correta comunicação entre FastAPI e a base de dados Supabase. Este mecanismo é fundamental para suportar a funcionalidade de conclusão de tarefas na aplicação móvel.

(a)



	id int8	route_id int8	client_id int8	type text	status text
	0	1	3	delivery	PENDING
	1	1	3	delivery	PENDING
	2	1	2	delivery	PENDING
	3	1	1	delivery	PENDING
	4	1	1	delivery	PENDING

(b)



	id int8	route_id int8	client_id int8	type text	status text
	0	1	3	delivery	PENDING
	1	1	3	delivery	completed
	2	1	2	delivery	PENDING
	3	1	1	delivery	PENDING
	4	1	1	delivery	PENDING

Figura 63 - Tabela tasks confirmação de alteração de estado

Na **Figura 63** apresenta a alteração do estado da tarefa na base de dados Supabase, demonstrando a ligação entre a aplicação e a base de dados, com intermédio da API local.

F.3 Recebimento de mensagens e notificações

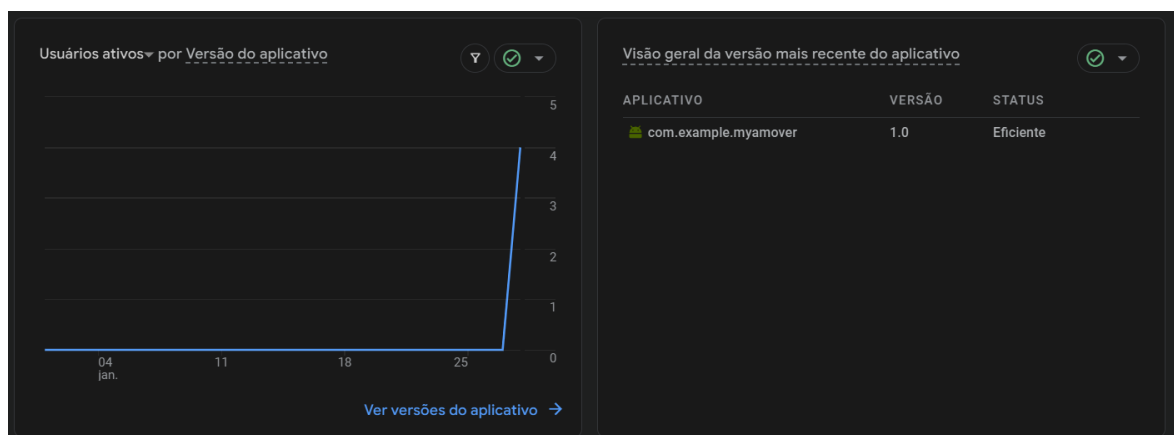


Figura 64 - Dashboard Firebase

A **Figura 64** apresenta a monitorização da aplicação através do Firebase, evidenciando o número ativos por versão e o estado da versão mais recente. Como visão geral é observado o estado com “Eficiente”, o que indica ausência de falhas críticas detetadas, funcionamento estável da versão monitorizada e desempenho adequado segundo os indicadores do Firebase.

Apêndice G – Testes de performance/desempenho

Os testes de performance permitem analisar o comportamento da aplicação em tempo real ao nível de consumo, memória e rede, bem como medir os tempos de resposta às principais interações do utilizador. Os testes incluíram o arranque da aplicação, carregamento da lista de tarefas, apresentação das rotas e as atualizações do estado da tarefa, garantindo que o sistema apresenta um desempenho adequado ao contexto de utilização em ambiente operacional real. Foram utilizados como ferramentas de apoio o Android Studio e o Android Profiler.

A avaliação do desempenho da aplicação nos principais fluxos operacionais, do condutor, medirá os seguintes aspetos:

- Tempo de arranque;
- Tempo de carregamento do ecrã *Home*;
- Tempo de carregamento da lista em *Tasks*;
- Tempo de abertura de *TaskDetail*;

- Tempo de Renderização da rota no ecrã *Map*; consumo de CPU, memória e rede durante operações.

É de observar que para uma UI fluida a 60Hz cada *frame* deve estar abaixo de 16,67ms, sendo maior ocorre jank(lag/stutter).

Nos testes instrumentados de desempenho, utilizando a biblioteca Andriid Macrobenchmark, com o propósito de avaliar o comportamento temporal da avaliação da aplicação, foram executados em dispositivo físico e desenvolveram interação direta cm a interface gráfica, medindo tempos de arranque, navegação entre ecrãs e carregamento de conteúdos.

G.1 Arranque da aplicação

Neste teste fui testado a capacidade de arranque da aplicação quando o utilizador aciona o botão “*Login*”, fazendo assim o arranque a aplicação. Para este teste foi utilizado as ferramentas Capture System Activities e View Live Telemetry, **Figura 65**.

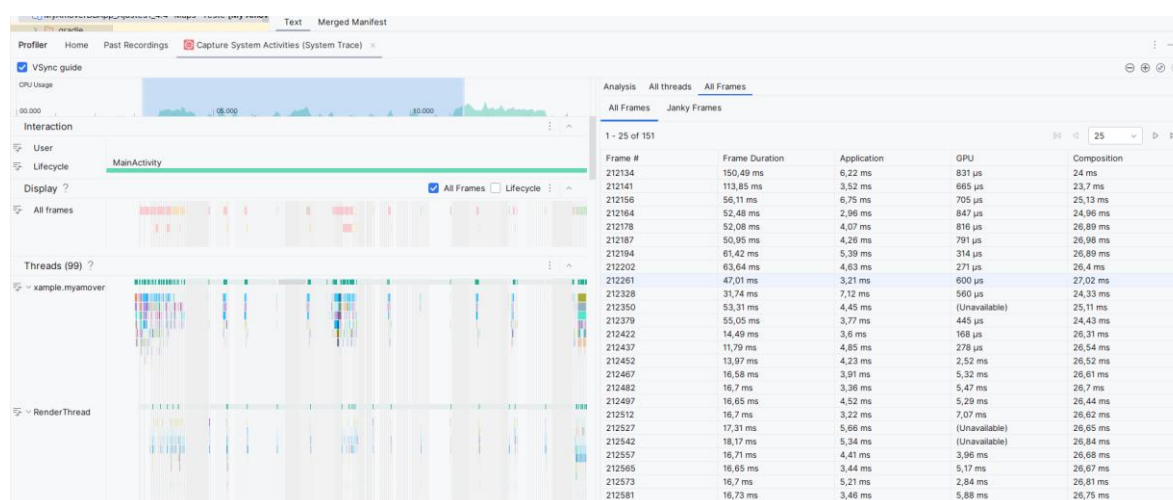


Figura 65 -Análise Capture System Activities - arranque aplicação

TABELA 14 - CUSTO ARRANQUE APLICAÇÃO

Componentes	Valores típicos
Application	3 -7 ms
GPU	< 1 ms
Composition	24-27 ms

Foram identificados *frames* com duração superior a 50ms, caracterizando ocorrências de *jank* visual, **Tabela 14**. A análise demonstra que o principal fator limitativo não se encontra na lógica da aplicação nem no processamento gráfico direto, mas sim na fase de composição da frame, realizada pelo sistema Android. O evento de interação associado ao botão de autenticação apresentou uma duração aproximada de 97ms, valor considerado adequado para aplicações móveis e correspondendo ao requisito não funcional, abertura da aplicação menor que 4s, **Figura 66**.

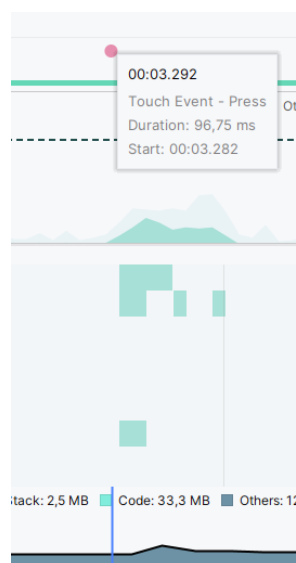


Figura 66 - Análise View Live Telemetry - arranque aplicação

Estes testes foram um sucesso, demonstrando que a aplicação responde de forma imediata à ação do utilizador, não sendo observados bloqueios significativos na interface gráfica durante o início do processo de autenticação e que o tempo de execução da lógica da aplicação se manteve reduzido (entre 3ms e 7ms), sendo o principal fator limitativo a fase de composição gráfica, com valores entre 24ms e 27ms por *frame*, comportamento esperado em aplicações com interfaces complexas e integrações de mapas.

G.2 Carregamento de lista de tarefas

Neste teste foi avaliado a capacidade de carregamento da lista de tarefas, quando o utilizador já se encontra na página principal e aciona o botão “*Task*”, abrindo a *interface* com a rota do dia/lista de tarefas. Para este teste foi utilizado as ferramentas Capture System Activities e View Live Telemetry, **Figura 67** e **Figura 68**.

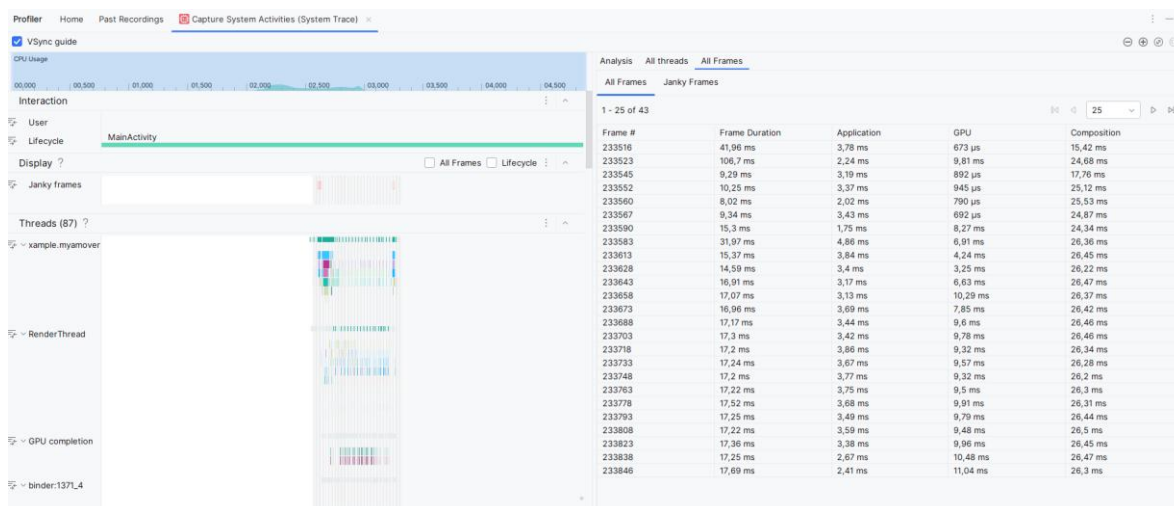


Figura 67 - Análise Capture System Activities - carregamento lista/rota



Figura 68 - Análise View Live Telemetry - carregamento lista/rota

TABELA 15 - CUSTO CARREGAMENTO DA LISTA DE TAREFAS

Componentes	Valores típicos
Application	2 - 4 ms
GPU	6 - 11 ms
Composition	24-26 ms

Durante a navegação para o ecrã de tarefas, conforme apresentado na **Tabela 15**, o evento de toque no botão apresentou uma duração média de aproximadamente 63ms, indicando uma boa resposta imediata da interface. No entanto, durante a criação inicial do

ecrã e Renderização da lista de tarefas, observaram-se alguns *frames* com duração próximos de 107ms. A análise demonstra que o tempo de execução da lógica da aplicação se manteve reduzido (entre 2ms e 4ms), sendo a composição gráfica do sistema Android com valores entre 24ms e 26ms por *frame*. O teste demonstrou que a aplicação estabiliza entre 16-17ms, garantindo uma navegação fluída.

G.3 Apresentação das rotas

Neste teste foi avaliado a capacidade de carregamento da rota na aplicação de navegação, quando o utilizador já se encontra na página da lista de tarefas/rota, aciona o botão “*Start Navigation*”, abrindo a *interface* de navegação. Para este teste foi utilizado as ferramentas Capture System Activities e View Live Telemetry, **Figura 69** e **Figura 70**.

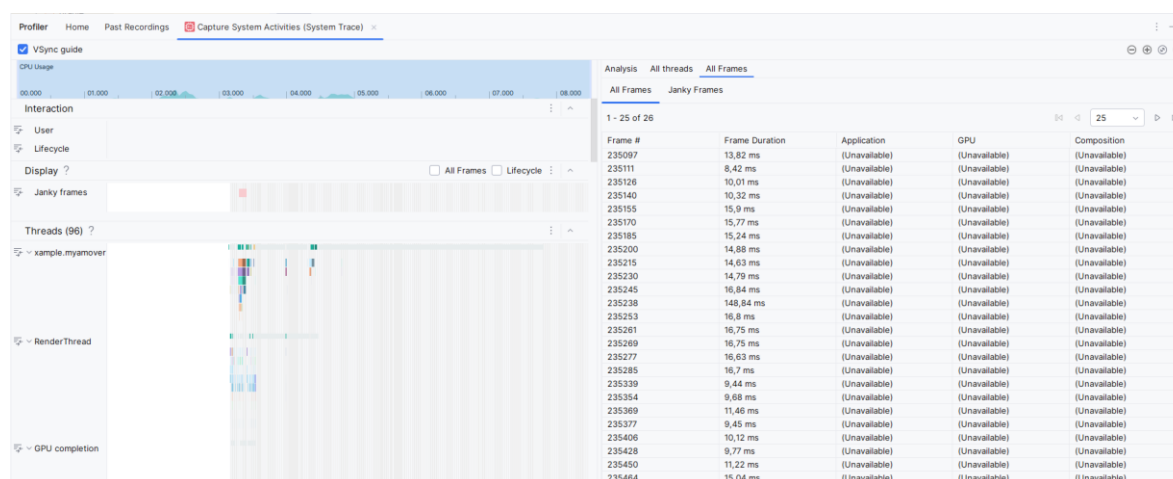


Figura 69 - Análise Capture System Activities - rotas

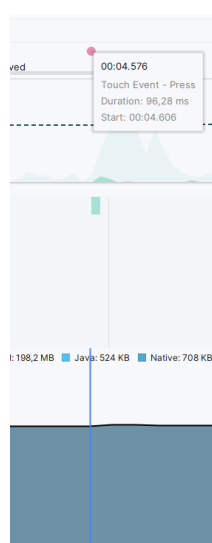


Figura 70 - Análise View Live Telemetry - rotas

Este teste demonstrou que durante a ativação da funcionalidade de navegação, através do botão “*Start Navigation*”, o evento de toque apresentou uma duração aproximada de 96ms, indicando uma resposta imediata da *interface*. Na transição para a aplicação externa Google maps, foi observado um *frame* isolado elevada (aprox. 149ms), associado à criação do *intente*, e provavelmente da criação do *polyline*. Este comportamento é esperado em sistemas Android quando ocorre a mudança de contexto entre aplicações. O consumo de memória manteve-se estável durante o processo, não sendo identificados indícios de degradação de desempenho ou fugas de memória.

G.4 Atualizações do estado da tarefa

Neste teste foi avaliado a capacidade de carregamento da alteração do estado da tarefa, quando o utilizador se encontra na página da lista de tarefas/rota, aciona o *card*, selecionando o botão “*complete task*” e alterando o estado da tarefa. Para este teste foi utilizado as ferramentas Capture System Activities e View Live Telemetry, **Figura 71** e **Figura 72**.

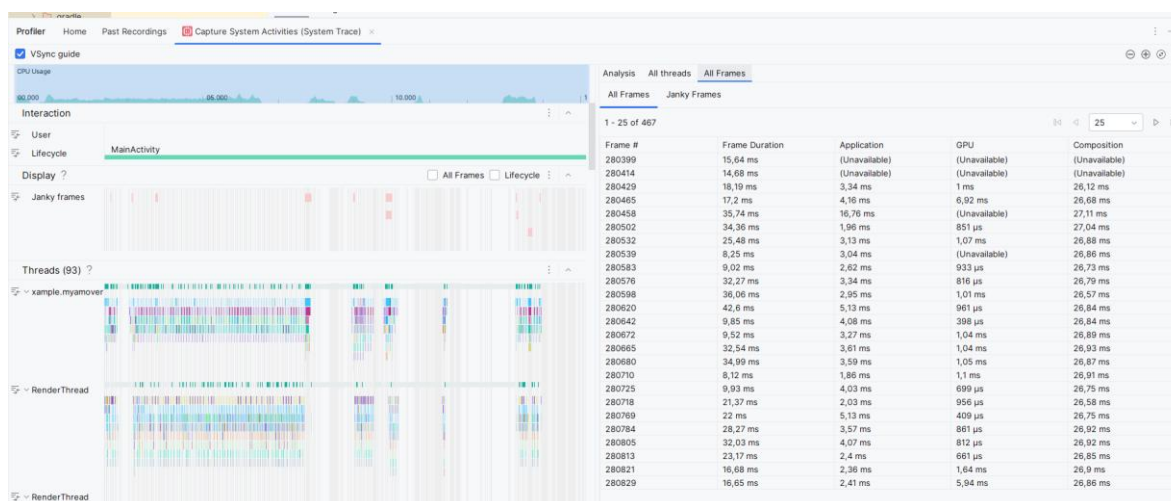


Figura 71 - Análise Capture System Activities - Atualização rotas



Figura 72 - Análise View Live Telemetry - atualização rotas

TABELA 16 - CUSTO CARREGAMENTO DA LISTA DE TAREFAS

Componentes	Valores típicos
Application	2 - 5 ms
GPU	~1 ms
Composition	26-27 ms

Neste teste verificou-se que durante a operação de conclusão de uma tarefa a partir do ecrã de lista/rota, foram registados eventos de interação com duração média entre 97 ms e 130 ms, indicando uma resposta adequada da interface. Contudo, observou-se um evento isolado com duração aproximada de 425ms, associado ao envio do pedido à API e à atualização subsequente da interface gráfica.

A análise dos *frames*, **Tabela 16**, revelou valores entre 25ms e 42ms durante este processo, evidenciando degradação temporária da fluidez visual. Verificou-se que o tempo de execução da lógica da aplicação permaneceu reduzido (2-5ms), sendo o principal fator limitativo a fase de composição gráfica do sistema Android, responsável pela recomposição da lista de tarefas e da rota. Após a estabilização da *interface*, os tempos de renderização regressam a valores próximo dos 16-17 ms por *frame*, garantindo uma navegação fluida.

G.5 Tempo de arranque da aplicação

Estes testes, realizados através de Android Macrobenchmark, mediu os tempos de arranque e latência de navegação entre ecrãs críticos. Estes testes foram executados em dispositivo físico. O foco principal foi medir o tempo de arranque em *cold start*, isto é, arranque da aplicação sem processos previamente em memória, **Figura 73**.

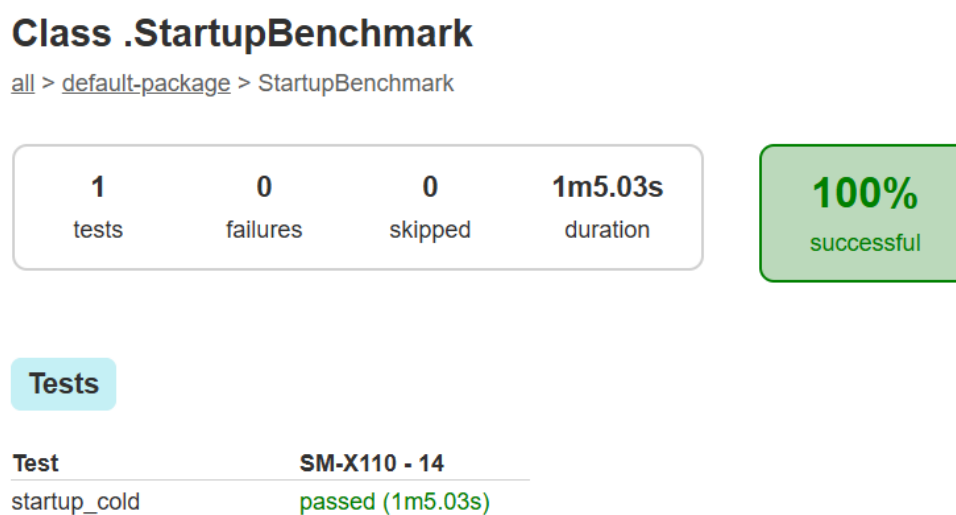


Figura 73 - Resultados teste desempenho arranque da aplicação

O ensaio foi executado num dispositivo Samsung SM-X110 com Android 14, garantindo ambiente estável e evitando distorções associadas a emulação. O teste foi concluído com sucesso, sem falhas ou interrupções, apresentando tempo de arranque aproximado de 1 minuto e 5 segundos. Embora o valor observado na Figura 67, seja superior ao esperado para *builds* de produção, importa contextualizar que o teste foi realizado em ambiente de desenvolvimento, com instrumentação ativa de *benchmark*, registo de *logs* e verificações adicionais que introduzem sobrecarga artificial. Adicionalmente, o *hardware* do dispositivo utilizado enquadra-se na gama média, o que influencia diretamente o desempenho medido.

Apesar disso, o resultado confirma que a aplicação inicializa todos os seus componentes sem falhas críticas, carrega corretamente o fluxo de autenticação, renderiza a interface sem bloqueios fatais e mantém estabilidade durante todo o processo de arranque. Estes indicadores demonstram que, mesmo sob instrumentação de *benchmark* e em ambiente de desenvolvimento, o sistema apresenta comportamento consistente e previsível, assegurando integridade funcional desde a fase inicial de execução.

Apêndice H – Testes unitários

H.1 LoginViewModel

Os testes unitários executados sobre a classe **LoginViewModel** tiveram como objetivo avaliar a lógica interna de forma isolada, assegurando previsibilidade, integridade do estado e correção dos fluxos de autenticação. Executados em ambiente JVM, sem dependência de componentes Android ou serviços externos, permitiram validar o comportamento da ViewModel de forma determinística, isolando a camada de apresentação.

Através da simulação de cenários e falha na autenticação, foi possível validar transições de estado, tratamento de exceções e consistência dos dados expostos via *state holders*. Esta abordagem contribuiu para a deteção precoce de erros lógicos, aumento da confiabilidade do sistema e melhoria da manutenção do código, assegurando que alterações futuras na ViewModel não introduzam regressões funcionais, **Figura 74**.

Class com.example.myamover.model.LoginViewModelTest

all > com.example.myamover.model > LoginViewModelTest

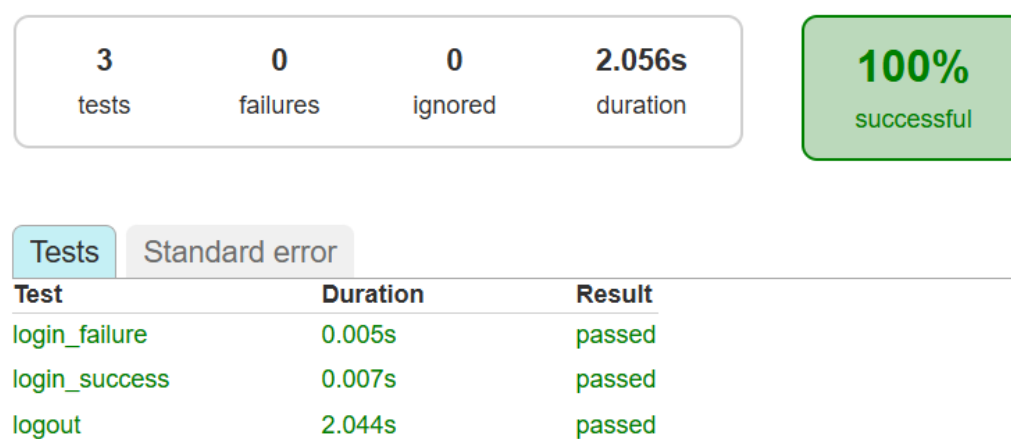


Figura 74 - Resultados teste unitário LoginViewModel

Os testes executados, **Figura 74**, sobre a classe apresentaram resultados integralmente positivos, sem falhas ou testes ignorados. Foram realizados testes distintos, com duração total de execução de aproximadamente 2,056 segundos. Este conjunto de testes valida os principais

cenários associados ao processo de autenticação, login bem – sucedido, falha de autenticação e encerramento da sessão.

Os resultados demonstram que o ViewModel responde de forma determinística aos diferentes fluxos de execução, assegurando a realização do estado interno e consistência do comportamento da aplicação. Observa-se que dois testes apresentam tempos de execução muito reduzidos (inferiores a 0,01 segundos), o que indica execução praticamente instantâneas da lógica interna. O teste de logout apresenta duração superior, o que está associado à simulação dos fluxos assíncronos ou espera controlada de *coroutines*, comportamento esperado em cenários de teste de estado reativo. Ainda assim, o tempo total permanece baixo, confirmando que os testes são adequados para execução frequente em pipelines de integração contínua.

H.2 TaskRemoteViewModel

Nos testes de integração associados à classe TaskRemoteViewModel o objetivo foi validar o comportamento coordenado entre ViewModel e a camada de repositório responsável pela gestão remota de rotas. Estes testes verificam a integração entre componentes internos, simulando fluxos completos de carregamento, atualização e conclusão de tarefas.

Class com.example.myamover.model.TaskRemoteViewModelTest

all > [com.example.myamover.model](#) > TaskRemoteViewModelTest

6	0	0	2.097s	100% successful
tests	failures	ignored	duration	

Tests	Standard error
Test	Duration
completeTaskAndRefresh_failure_callsOnError	0.050s
completeTaskAndRefresh_success_updatesRoute_callsOnSuccess	0.035s
init_callsLoadTasksAndLoadTodayRoute	0.016s
loadTasks_failure_setsError	1.956s
loadTasks_success_updatesState	0.020s
loadTodayRoute_success_setsRoute	0.020s

Figura 75 - Resultados teste unitário TaskRemoteViewModel

Pelos resultados obtidos na **Figura 75** podemos observar que foi validado o comportamento de inicialização automática do ViewModel. O teste confirma que, no momento da criação do objeto, o sistema desencadeia o carregamento da lista de tarefas e da rota diária. Os testes de carregamento de tarefas verificam tanto cenários de sucesso como de falha. Nos casos de sucesso, confirma-se que os dados recebidos do repositório são refletidos corretamente no estado da interface. Nos cenários de erro, o sistema converte exceções em estados controlados, preservando previsibilidade e evitando falhas não tratadas.

Nos fluxos de conclusão de tarefas, que representam operações transacionais críticas, os testes garantem que, quando uma tarefa é concluída com sucesso, a rota diária é atualizada, os *callbacks* de sucesso são acionados e o estado global é sincronizado e, em caso de falha, o sistema propaga corretamente o erro e executa os *callbacks* apropriados, mantendo integridade do fluxo. O teste de falha no carregamento de tarefas apresenta duração superior aos restantes, o que sugere simulação de latência assíncrona controlada. Este comportamento é esperado em cenários de validação de *coroutines* e confirma que o sistema lida adequadamente com operações não bloqueadoras.

No conjunto, os testes de integração demonstram que o módulo de gestão de tarefas apresenta comportamento consistente, tratamento resiliente de erros e sincronização correta entre componentes, assegurando estabilidade operacional em cenários reais de utilização.

Apêndice I – Testes algoritmos

Cada cenário foi modelado com base numa história realista e testado com um algoritmo específico, escolhido pela sua adequação às características do problema. Os resultados obtidos permitiram identificar as vantagens e limitações de cada abordagem, como apresentados a seguir:

Cenário 1 - Rede Local de Distribuição Alimentar (DelivFresh)

- *Descrição:* A empresa DelivFresh inicia as entregas de produtos frescos logo ao amanhecer, com o objetivo de manter a qualidade e frescura. As encomendas incluem pão quente, bolos e queijos regionais. O serviço tem de ser concluído antes das 08:00, respeitando restrições de tempo e autonomia.

- *Configurações:*

- › Partida: Armazém DelivFresh (nó 0) às 06:30;
- › Clientes: Seis estabelecimentos (nós 1 a 6);
- › Capacidade da mota: 25Kg;
- › Autonomia: 10 kWh;
- › Janelas de tempo: 06:30 - 08:00.

- *Algoritmo aplicado:* Tabu Search

- *Justificação:* Adapta-se bem a problemas com múltiplas restrições e alta complexidade combinatória.

- *Resultados:*

- › Rota: [0, 1, 2, 3, 4, 5, 6, 0];
- › Distância total: 45.38 km;
- › Energia estimada: 9.08 kWh;
- › Janelas de tempo: Cumpridas (06:30-08:00).

A rota cumpriu todos os requisitos logísticos, cumpriu integral das restrições, boa eficiência na redução de distância e garantiu a entrega dentro da janela de tempo.

Cenário 2 - Recolha de Análises Clínicas (MobileSaúde)

- *Descrição:* A unidade móvel do laboratório MobileSaúde recolhe amostras clínicas em várias instituições. As recolhas devem seguir uma ordem específica, respeitando janelas de tempo e mantendo o material em condições adequadas.

- *Configurações:*

- › Partida: Laboratório Central (nó 0) às 08:00;
- › Clientes: 5 pontos com janelas de tempo (nós 1 a 5);
- › Capacidade da mota: 18Kg;
- › Autonomia: 10 kWh;

- *Algoritmo aplicado:* GRASP

- *Justificação:* Geração de múltiplas soluções com melhoria local, ideal para janelas de tempo.

- *Resultados:*

- › Rota: [0, 1, 5, 2, 4, 3, 0];
- › Distância total: 28.88 km;
- › Janelas de Tempo: Cumpridas para todos os locais.

O algoritmo GRASP gerou rota compatível com todas as restrições temporais e de carga. Alta flexibilidade para lidar com múltiplas combinações de visitas. Resultados consistentes com eficiência aceitável.

Cenário 3 – Assistência Técnica Rápida (TecFast)

– *Contexto:* Uma avaria urgente num painel solar ativa uma missão de assistência técnica com partida imediata. O objetivo é chegar ao local no menor tempo e distância possíveis, com bateria limitada.

– *Configuração:*

- › Partida: Garagem central (nó 0) às 15:12;
- › Chegada: Escola Técnica (nó 3);
- › Capacidade da mota: irrelevante;
- › Autonomia: 5 kWh.
- › Restrições: Rota direta e mínima.

– *Algoritmo aplicado:* Dijkstra

– *Justificação:* Cálculo do caminho mais curto entre dois pontos.

– *Resultados:*

- › Rota ótima: [0, 3];
- › Distância total: 8.6 km;
- › Energia estimada: 1.72 kWh.

O algoritmo de Dijkstra foi ideal para encontrar o percurso mais curto até ao destino. Adequado para rotas ponto-a-ponto em situações urgentes. Tempo de execução muito baixo.

Cenário 4 – Logística com Múltiplos Depósitos (EcoPost)

– *Contexto*: A empresa EcoPost distribui materiais reciclados a partir de dois depósitos. Cada mota parte do seu depósito e realiza entregas específicas com foco na eficiência energética.

– *Configuração*:

- › Partida: Mota Norte (nó 0): Escola 1, Papelaria Laranja;
- › Partida: Mota Sul (nó 5): Escola 2, Papelaria Azul;
- › Capacidade da mota: 30 Kg/mota;
- › Autonomia: 8 kWh.
- › Objetivo: máxima eficiência e equilíbrio logístico.

– *Algoritmo aplicado*: Savings (Clarke & Wright)

– *Justificação*: Ideal para agrupamento local e economia de percurso em múltiplos depósitos.

– *Resultados*:

- › Rota Norte: [0, 1, 2, 0] → 25.62 km;
- › Rota Sul: [5, 3, 4, 5] → 21.04 km.

O algoritmo Savings foi eficaz na divisão das rotas com base nos depósitos. Boa economia de percurso e balanceamento entre as motas. Resultado final com consumo energético equilibrado.

Cenário 5 – Visita Técnica a Microempresas (ProLocal)

– *Contexto*: Serviço técnico de manutenção preventiva a pequenos comércio. O número de clientes é reduzido, permitindo uso de algoritmo exato.

– *Configuração*:

- › Partida: Oficina (nó 0) às 09:00;
- › Clientes: 6 locais até às 12:00;
- › Capacidade da mota: irrelevante;
- › Autonomia: 10 kWh;
- › Restrições: Sem janelas de tempo, com foco em distância mínima.

– *Algoritmo aplicado*: Branch and Bound

– *Justificação*: Ideal para instâncias pequenas, permitindo encontrar a solução ótima.

– *Resultados*:

- › Rota: [0, 1, 3, 5, 4, 6, 2, 0];
- › Distância total: 21.4 km.

O algoritmo exato encontrou a solução ótima. Tempo de execução aceitável, dado o número reduzido de nós. Ideal para casos com pequena escala e elevado rigor.

Cenário 6 – Entregas Expresso em Zonas Próximas (FastDrop)

– *Contexto*: Serviço de entregas expresso em zonas densas, onde a decisão rápida é prioritária. O foco é encontrar rotas aceitáveis rapidamente.

– *Configuração*:

- › Partida: Centro logístico (nó 0) às 10:00;
- › Clientes: 6 clientes até às 12:00;
- › Capacidade da mota: 20 Kg;
- › Autonomia: 10 kWh;

– *Algoritmo aplicado*: Nearest Neighbor (com 2-opt)

– *Justificação*: Heurística rápida, melhorada localmente para solução eficiente.

– *Resultados*:

- › Rota: [0, 1, 6, 4, 5, 2, 3, 0];
- › Distância total: 17.63 km;
- › Energia estimada: 3.53 kWh

Solução gerada rapidamente, com distância aceitável. Ideal para contextos com baixa complexidades. A aplicação do 2-opt melhorou a solução inicial.

Os testes realizados ao longo deste trabalho demonstraram que a aplicação de algoritmos de otimização de rotas em cenários urbanos reais é não apenas viável, mas também altamente eficaz no apoio à operação profissional de motas elétricas. A estratégia de associar cada algoritmo a um cenário específico — considerando as respetivas restrições, objetivos e características operacionais — revelou-se adequada e produtiva, permitindo obter soluções sólidas, eficientes e ajustadas à realidade de cada contexto.