UNIVERSIDADE
LUSÓFONA

Zero Queixa

Trabalho Final de curso

Relatório Intercalar 2º Semestre

Nsimba Tamaku, a22409693, LEI

Orientador: Prof. Pedro Alves

Entidade Externa: Associação Zero

Departamento de Engenharia Informática da Universidade Lusófona

Centro Universitário de Lisboa

08/04/2026

Direitos de cópia

ZeroQueixa, Copyright de Nsimba Tamaku, ULHT.

A Escola de Comunicação, Arquitetura, Artes e Tecnologias da Informação (ECATI) e a Universidade Lusófona de Humanidades e Tecnologias (ULHT) têm o direito, perpétuo e sem limites geográficos, de arquivar e publicar esta dissertação através de exemplares impressos reproduzidos em papel ou de forma digital, ou por qualquer outro meio conhecido ou que venha a ser inventado, e de a divulgar através de repositórios científicos e de admitir a sua cópia e distribuição com objetivos educacionais ou de investigação, não comerciais, desde que seja dado crédito ao autor e editor.

Resumo

O ZeroQueixa é uma aplicação móvel multiplataforma desenvolvida em parceria com a Associação ZERO – Associação Sistema Terrestre Sustentável, destinada ao registo e acompanhamento de queixas relacionadas com mobilidade urbana e proteção ambiental. Este Trabalho Final de Curso representa a continuidade do projeto iniciado no ano letivo 2024/2025, cujo objetivo é transformar o Produto Mínimo Viável (MVP) já funcional numa plataforma madura e pronta para utilização em larga escala.

O MVP, composto por uma aplicação móvel (iOS e Android) desenvolvida em Flutter e um sistema de BackOffice web para gestão pela ZERO, encontra-se atualmente em fase de testes piloto desde outubro de 2024. A fase atual de desenvolvimento concentra-se em três objetivos principais: preparar a aplicação para testes de utilizadores mais alargados e subsequente lançamento em produção, melhorar o BackOffice na vertente de processamento e encaminhamento de queixas para as entidades responsáveis, e planear a implementação de uma versão 2.0 com componente social (mapas interativos, feeds públicos, subscrição de queixas).

Este trabalho distingue-se por atingir uma fase raramente alcançada em projetos académicos – a preparação para produção efetiva – contribuindo simultaneamente para a missão da ZERO de promover cidadania ativa e sustentabilidade ambiental em Portugal.

Palavras-chave: Mobilidade Urbana, Proteção Ambiental, Aplicação Móvel, Participação Cívica, Flutter, Spring Boot, Gestão de Queixas.

Abstract

ZeroQueixa is a multiplatform mobile application developed in partnership with the ZERO Association – Sustainable Terrestrial System Association, designed to register and monitor complaints related to urban mobility and environmental protection. This Final Course Project represents the continuation of the project started in the 2024/2025 academic year, whose objective is to transform the already functional Minimum Viable Product (MVP) into a mature platform ready for large-scale use.

The MVP, consisting of a mobile application (iOS and Android) developed in Flutter and a web-based BackOffice system for management by ZERO, has been in the pilot testing phase since October 2024. The current development phase focuses on three main objectives: preparing the application for wider user testing and subsequent production launch, improving the BackOffice in terms of processing and forwarding complaints to the responsible entities, and planning the implementation of a version 2.0 with a social component (interactive maps, public feeds, complaint subscription).

This work stands out for reaching a stage rarely achieved in academic projects – preparation for actual production – while simultaneously contributing to ZERO's mission of promoting active citizenship and environmental sustainability in Portugal.

Keywords: Urban Mobility, Environmental Protection, Mobile Application, Civic Participation, Flutter, Spring Boot, Complaint Management.

Índice

Resumo ..... *iii*

Abstract ..... *iv*

Lista de figuras ..... *vii*

Lista de tabelas ..... *viii*

1 Introdução *1*

1.1 Contexto e Motivação **1**

1.2 Objetivos do Trabalho **2**

2 Pertinência e Viabilidade *3*

2.1 Pertinência **3**

2.1.1 Contexto da Continuidade 3

2.1.2 Necessidades Identificadas e Validadas 3

2.1.3 Alinhamento Estratégico com a Missão da ZERO 3

2.2 Viabilidade **3**

2.2.1 Viabilidade Técnica 3

2.2.2 Viabilidade Económica 4

2.2.3 Viabilidade Social 4

2.3 Análise Comparativa com Soluções Existentes **5**

2.3.1 Soluções existentes 5

2.3.2 Análise de benchmarking 5

3 Especificação e Modelação *7*

3.1 Análise de Requisitos **7**

3.1.1 Enumeração de Requisitos 7

3.1.2 Descrição detalhada dos requisitos principais 8

3.1.3 Casos de Uso/User Stories 9

3.2 Modelação **11**

3.2.1 Processamento das categorias 13

4 Solução Desenvolvida *17*

4.1 Apresentação **17**

4.2 Arquitetura **18**

4.3 Tecnologias e Ferramentas Utilizadas **19**

4.3.1 FrontEnd (Cliente) 19

4.3.2 BackOffice Web 19

4.3.3 Base de Dados 20

4.4 Ambientes de Teste e de Produção **20**

4.4.1 Ambiente de Desenvolvimento 20

4.4.2 Ambiente de Testes 20

4.4.3 Ambiente de Produção 20

4.5 Abrangência 21
4.6 Evolução face ao MVP 22
4.6.1 Backend e Backoffice 22
4.6.2 Aplicação Móvel 25
Método e Planeamento 29
4.7 Planeamento inicial 29
4.7.1 Desvios ao Planeamento e Factores Externos 29
4.8 Processo de Desenvolvimento e Gestão de Contribuições 31
5 — Testes e Validação 33
5.1 Coordenação com a Associação ZERO 33
5.2 Angariação de Testadores via Contexto Académico 33
5.3 Vídeo Promocional 33
5.4 Issues Identificados na Fase de Testes 33
5.4.1 Crash ao negar permissão de câmara (Android) 33
5.4.2 Crash ao pesquisar localização com widget desmontado (iOS) 33
5.4.3 Troca incorreta de foco entre campos de identificação (iOS) 34
5.5 Resultados Obtidos 34
Formulário de declaração de uso de ferramentas de Inteligência artificial 35
Glossário 39
Anexo A 40
6 Referencias 41

Lista de figuras

Figura 1 - Diagrama de Use Case Registo de uma queixa 9
Figura 2 - Diagrama Use Case Agrupamento de queixa 10
Figura 3 - Diagrama de relações 12
Figura 4 - Parte do Excel para processamento de categorias 13
Figura 5 - Interface do backoffice para carregamento do ficheiro CSV de categorias 15
Figura 6 - Visualizador de categorias com a árvore expandida 16
Figura 7 - Painel de detalhe de uma categoria com campos obrigatórios visíveis 16
Figura 8 - Arquitetura do ZeroQueixa 18
Figura 9 - Estado inicial da camada de controlo da API REST, evidenciando a ausência de separação de responsabilidades: endpoints de três domínios distintos (utilizadores, queixas e categorias) concentrados num único controlador, sem versionamento de API 22
Figura 10 - Ecrã inicial (sem elementos) 27
Figura 11 - Ecrã inicial (atualizado) 27
Figura 12 - Categoria de seleção de categoria (elementos em grelha) 27
Figura 13 - Categoria de seleção de categoria (elementos em coluna) 27
Figura 14 - Ecrã de resumo da queixa (antigo) 28
Figura 15 - Ecrã de resumo da queixa (atualizado) 28
Figura 16 - Diagrama de Gantt 29
Figura 17 - Exemplo de issue (#28) criado no repositório ZEROQueixa_Backend, com descrição estruturada dos problemas identificados e objetivos da refatoração. 32
Figura 18 - Pull Request #29 resultante do issue #28, evidenciando o branch de origem (efrahim20319:feature/Backend-28-new-rest-design), o branch de destino (DEISI-LABS-ULHT:new-rest-design) e a aprovação e merge efetuados pelo orientador. 32

Lista de tabelas

Tabela 1 - Tabela de benchmarking 5
Tabela 2 - Requisitos da aplicação móvel (APP) 7
Tabela 3 - Requisitos do BackOffice (BO) 8
Tabela 4 - Exemplo de estrutura do ficheiro CSV de categorias 14
Tabela 5 - Evolução dos endpoints no backend 23

1 Introdução

1.1 Contexto e Motivação

A ZERO – Associação Sistema Terrestre Sustentável é uma organização não governamental de ambiente sem fins lucrativos, com utilidade pública e de âmbito nacional. Exerce a sua atividade com total independência relativamente aos partidos políticos, empresas e entidades com fins lucrativos, de natureza religiosa e ao governo. Foi criada no final de 2015, fruto do interesse comum de cerca de uma centena de pessoas pela defesa dos valores da sustentabilidade (*ZERO - Associação Sistema Terrestre Sustentável*, 2025).

O projeto QueixaAqui (Ferreira & Alves, 2024) foi iniciado no ano 2024/2025 por Rui Ferreira, no âmbito do seu Trabalho Final de Curso, como resposta à necessidade identificada pela Associação ZERO de criar um canal digital eficaz para registo e acompanhamento de queixas relacionadas com mobilidade urbana e proteção ambiental. O trabalho desenvolvido por Rui Ferreira resultou na entrega de um MVP (Produto Mínimo Viável) funcional, composto por dois componentes principais: uma aplicação móvel multiplataforma (iOS e Android), desenvolvida em Flutter, que permite ao cidadão submeter queixas de forma simples e acompanhar o seu estado; e um sistema de BackOffice web, desenvolvido em HTML, CSS e Bootstrap com renderização no lado do servidor, destinado à equipa da ZERO para gestão e acompanhamento das queixas recebidas. O backend foi implementado em Spring Boot com Kotlin, seguindo uma arquitetura cliente-servidor, com comunicação via WebServices REST sobre HTTPS, e suportado por uma base de dados MySQL. A autenticação dos utilizadores foi assegurada via OAuth2, com suporte a login por Google e Apple.

No que respeita à aplicação móvel, foram implementadas funcionalidades como o registo e autenticação de utilizadores, a submissão de queixas com geolocalização, categorização e anexo de imagens, bem como a consulta de queixas submetidas. O BackOffice permitia à equipa da ZERO listar, consultar e alterar o estado das queixas registadas. No total, foram implementados e testados vinte e três endpoints REST. A solução foi colocada em operação piloto em outubro de 2024, com sete utilizadores ativos da equipa ZERO, que criaram diversas queixas de teste. O feedback recolhido foi positivo, destacando a simplicidade do processo de submissão e a clareza da interface.

Contudo, a análise da fase piloto, complementada por reuniões de acompanhamento com a ZERO, revelou lacunas que limitam o potencial do projeto, nomeadamente a necessidade de atualizar as categorias de queixas de forma mais simples, preparar a aplicação para testes mais alargados, e incorporar uma componente social que estará prevista para a versão 2.0.

O presente trabalho representa a continuação natural deste projeto, partindo da base sólida construída por Rui Ferreira para evoluir o QueixaAqui de um sistema básico de registo para uma plataforma completa e madura de gestão e acompanhamento de queixas cidadãs.

O trabalho adicional desenvolvido em relação ao MVP encontra-se detalhado na secção 4.6.

1.2 Objetivos do Trabalho

Este TFC tem como objetivo principal dar continuidade ao desenvolvimento do QueixaAqui, transformando-o de um sistema básico de registo numa plataforma completa e madura de gestão e acompanhamento de queixas cidadãs. A evolução proposta visa não apenas colmatar as lacunas identificadas, mas também posicionar o ZeroQueixa como referência nacional em participação cívica digital.

Objetivo Geral: Desenvolver e implementar um conjunto integrado de funcionalidades que complete o ciclo de vida das queixas, desde a submissão até à avaliação pós-resolução, promovendo transparência, eficiência e melhoria contínua.

Objetivos Específicos:

- Aperfeiçoar a solução atual de forma a se poderem efetuar testes de utilizador mais alargados, corrigir erros detetados durante essa fase e lançar a primeira versão em produção.
- Realizar uma manutenção corretiva à aplicação após estar em produção, acompanhando erros que irão necessariamente surgir com a sua utilização massiva.
- Melhorar o estado atual do backoffice, nomeadamente na forma como irão ser processadas e encaminhadas as queixas. Estas melhorias serão identificadas após a passagem a produção da aplicação móvel, para se perceber que tipo de queixas estão a ser introduzidas. As opções em análise são: enviar emails para entidades responsáveis, enviar para o Portal da Queixa, falar com as entidades responsáveis para criação de canais dedicados e as próprias entidades terem perfis de administradores no Zero Queixa.
- Implementar a versão 2.0, visando exercer uma componente mais social, que conta com diversos objetivos como: ter um mapa com todas as queixas detetadas, poder associar-se a uma queixa já existente ("também tenho esta queixa"), poder partilhar queixas nas redes sociais, ter um feed de queixas públicas, etc.

2 Pertinência e Viabilidade

2.1 Pertinência

2.1.1 Contexto da Continuidade

O projeto Zero Queixa é socialmente relevante porque oferece aos cidadãos um canal dedicado à denúncia e acompanhamento de problemas relacionados com o ambiente e a mobilidade urbana. Ao contrário de plataformas genéricas de reclamação, como o Portal da Queixa, as denúncias aqui registadas inserem-se num contexto de preocupação cívica genuína, refletindo problemas reais que afetam o quotidiano dos cidadãos e das comunidades onde se inserem.

A análise desta fase piloto, enquanto o trabalho estava encarregue ao Rui Ferreira, complementada por cinco reuniões estruturadas com a ZERO e feedback direto dos utilizadores através de inquéritos e análise de métricas, revelou que embora o sistema base seja sólido e a usabilidade seja excelente, existem lacunas críticas que limitam significativamente o potencial do projeto.

A nível pessoal, este projeto representa também uma oportunidade de desenvolver competências em tecnologias com grande procura no mercado, nomeadamente Spring Boot e Flutter, contribuindo para a minha formação profissional. Para além disso, trabalhar num projeto com impacto real na sociedade, que responde a um problema concreto e contribui para uma cidadania mais ativa e informada, constitui uma motivação adicional de grande valor.

2.1.2 Necessidades Identificadas e Validadas

A ZERO, enquanto organização sem fins lucrativos com recursos humanos limitados, identificou sistematicamente ao longo das reuniões de acompanhamento necessidade de atualizar as categorias das queixas de modo mais simples e descomplicado, preparar a aplicação para testes e a incorporação de uma componente mais social que estará apenas disponível na versão 2.0.

2.1.3 Alinhamento Estratégico com a Missão da ZERO

A Associação ZERO, reconhecida como entidade de utilidade pública desde 2015, tem como missão contribuir para a sustentabilidade da sociedade portuguesa através da defesa do ambiente, dos recursos naturais e da promoção de uma cidadania ativa e informada. As funcionalidades propostas neste trabalho reforçam diretamente cada um destes pilares de forma mensurável e concreta.

2.2 Viabilidade

2.2.1 Viabilidade Técnica

A viabilidade técnica deste projeto está solidamente fundamentada na infraestrutura robusta estabelecida durante a fase MVP. O backend em Spring Boot e Kotlin está operacional em ambiente de produção nos servidores da Universidade Lusófona, com vinte e três endpoints REST já implementados e testados, base de dados MySQL bem estruturada e normalizada, e sistema de autenticação via OAuth2 funcional. A aplicação móvel em Flutter está compilada e disponível em ambientes de teste (TestFlight para iOS e Play Console Internal Testing para Android), com arquitetura baseada em Provider (Ashfaque Khokhar, 2025) para gestão de estado e camada de abstração de dados já estabelecida. Esta fundação sólida significa que as novas funcionalidades serão extensões e integrações, não reconstruções, reduzindo significativamente o risco técnico.

2.2.2 Viabilidade Económica

Os custos diretos de desenvolvimento são praticamente nulos. O servidor está disponibilizado pela Universidade Lusófona sem custos, a base de dados MySQL é open-source, o Firebase (Stevenson, 2018) no plano Spark gratuito suporta amplamente as necessidades do projeto piloto e fase inicial, as contas Apple Developer e Google Play Console já foram pagas no TFC anterior, e todas as bibliotecas utilizadas são open-source com licenças permissivas (Apache, MIT).

2.2.3 Viabilidade Social

A viabilidade social do projeto está demonstrada pelo envolvimento ativo dos stakeholders em todas as fases. Os sete utilizadores piloto do MVP reportaram satisfação elevada e todos indicaram que voltariam a usar a aplicação, condicionado às melhorias que este trabalho precisamente implementa. A ZERO comprometeu formalmente três membros da equipa para testes contínuos durante o desenvolvimento, garantindo validação iterativa com utilizadores reais. As reuniões regulares com a organização asseguram alinhamento constante entre o desenvolvimento técnico e as necessidades operacionais reais.

O sistema promove inclusão e participação cívica ao eliminar barreiras tradicionais de acesso aos mecanismos de queixa institucional. A interface móvel é significativamente mais acessível que formulários web complexos ou necessidade de deslocação presencial.

2.3 Análise Comparativa com Soluções Existentes

2.3.1 Soluções existentes

Embora já existam aplicações com o propósito de os cidadãos fazerem as suas denúncias no âmbito de mobilidade ou meio ambiente, poucas entregam funcionalidades robustas para a criação da queixa, gestão e acompanhamento das mesmas de forma colaborativa e prática. Exemplos de soluções:

- Aplicações João Pimentel Ferreira: As propostas de solução do problema desenvolvidas pelo Engenheiro João Ferreira, permitem aos cidadãos reportar as suas denúncias, no entanto perante o contexto da queixa, existe uma aplicação diferente para cada, o que não é algo eficiente nem centralizado. Apesar de ser eficiente no registo da queixa, as aplicações apresentam algumas limitações significativas: falta de feedback aos utilizadores sobre o estado da queixa e a denúncia é apenas gerada em texto para depois, eventualmente, ser submetida pelo utilizador na instituição responsável.
- Portal de Queixas: É uma plataforma que permite aos consumidores registar reclamações contra empresas ou serviços. Esta solução é focada na partilha de experiências e resolução do problema, através da mediação de conflitos entre utilizador e entidade. O sistema tem um alcance nacional significativo, no entanto não se destina as denúncias relacionadas com questões especificas de mobilidade ou meio ambiente e não existe como aplicação de telemóvel o que é uma grande lacuna comparado com o QueixaAqui.

2.3.2 Análise de benchmarking

Características/Funcionalidades	**QueixaAqui**	**Aplicações João Pimentel**	**Portal De Queixas**
Foco em Mobilidade e Ambiente	X	X	
Acompanhamento da Queixa	X		X
Anexar imagens/áudio	X		X
Registo da denuncia georreferenciada	X		
Mapa interativo com denúncias categorizadas*	X		
Sistema de feed para explorar outras queixas*	X		X
Subscrição de Queixas já registadas*	X		
BackOffice para gestão da queixa	X		X

Tabela 1 - Tabela de benchmarking

A nível institucional, a União Europeia dispõe de mecanismos como o **TERM** (*Transport and Environment Reporting Mechanism (TERM) Publications | Transport and Mobility | European Environment Agency (EEA)*, 2026), gerido pela Agência Europeia do Ambiente(*European Environment Agency (EEA)*, 2026), e o **EU Urban Mobility Observatory** (*EU Urban Mobility Observatory - Regional and Urban Policy*, 2024), orientados para a monitorização e partilha de informação entre Estados-Membros e profissionais do setor. Contudo, estas iniciativas não constituem canais de participação direta para os cidadãos, não permitindo o registo nem o acompanhamento de queixas individuais.

Constata-se, assim, a ausência de uma solução europeia unificada, de âmbito cidadão, dedicada especificamente às temáticas de mobilidade urbana e ambiente, com acompanhamento integrado das ocorrências por uma entidade especializada — lacuna que o Zero Queixa procura endereçar em contexto nacional, em parceria com a associação ZERO.

*: Previsto para versão 2.0

3 Especificação e Modelação

3.1 Análise de Requisitos

O levantamento de requisitos para as funcionalidades de continuidade do QueixaAqui seguiu uma abordagem iterativa e colaborativa, que consistiu em sessões estruturadas de elicitação com a equipa da Associação ZERO. Realizaram-se reuniões formais, envolvendo o coordenador técnico da ZERO, dois gestores operacionais, o orientador Prof. Pedro Alves e o desenvolvedor.

3.1.1 Enumeração de Requisitos

Foram identificadas melhorias e bugs por resolver antes que a aplicação entre para a fase de testes propiamente dita. Semanalmente é lançada uma nova versão em que são resolvidos iterativamente cada um desses requisitos. Os seguintes requisitos foram divididos em duas listas, contemplando os dois grandes componentes do projeto, a aplicação móvel e o backoffice administrativo.

Abaixo está a listagem detalhada, cada requisito é classificado com base em:

- Tipo: Obrigatório, Nice to Have
- Prioridade: Alta, média ou baixo
- Impacto: Elevado, médio ou reduzido

ID	Descrição	Tipo	Prioridade	Impacto
APP-1	Página de consulta dos dados do utilizador	Obrigatório	Média	Médio
APP-2	Melhorias na interface gráfica	Obrigatório	Alta	Alto
APP-3	Impedir que o utilizador submeta a queixa sem preencher os campos obrigatórios definidos no ficheiro excel de categorias (que será explicado em detalhe no capítulo 3)	Obrigatório	Alta	Alto
APP-4	Colocar ícone de abertura da aplicação.	Obrigatório	Baixo	Baixo
APP-5	Possibilidade de ver as queixas num mapa	Obrigatório	Alta	Médio

Tabela 2 - Requisitos da aplicação móvel (APP)

ID	Descrição	Tipo	Prioridade	Impacto
BO-1	Permitir o carregamento do ficheiro excel de categorias diretamente do backoffice.	Obrigatório	Alta	Alto
BO-2	Adaptar os webservices relacionados as categorias para incluírem novos campos.	Obrigatório	Alta	Médio
BO-3	Agrupar queixar relacionadas	Nice to have	Média	Alto

Tabela 3 - Requisitos do BackOffice (BO)

3.1.2 Descrição detalhada dos requisitos principais

APP-3: Impedir que o utilizador submeta a queixa sem preencher os campos obrigatórios definidos no ficheiro excel de categorias

- Objetivo: Garantir que o utilizador preencha sempre os campos obrigatórios para as categorias que assim exigirem.
- Dependências: Webservice das categorias indicando quais campos são obrigatórios para cada categoria.
- Critérios de Aceitação: A queixa nunca deve ser submetida com os campos obrigatórios em falta, em todos os casos.

BO-3: Agrupar queixar relacionadas

- Objetivo: O backoffice deve sugerir queixas que podem estar relacionadas ao mesmo assunto para posteriormente agrupá-las.
- Dependências: Algoritmo de similaridade de queixa.
- Critérios de Aceitação: Quando novas queixas são registadas no sistema, o algoritmo de similaridade é executado, então, o sistema sugere queixas com assuntos e categorias parecidos e apresenta a percentagem de similaridade.

APP-5: O utilizador poderá ter acesso a um mapa interativo integrado na aplicação em que possibilitará ver os pontos geográficos onde são reportadas as queixas.

- Objetivo: Permitir o utilizador acompanhar visualmente os locais mais afetados.
- Dependências: Integração com o google maps.
- Critérios de Aceitação: Quando novas queixas são registadas no sistema, é criado um ponto no mapa - que estará visível enquanto a queixa não estiver resolvida - em que o utilizador poderá verificar se de que queixa se trata.

3.1.3 Casos de Uso/User Stories

Para contextualizar de maneira mais clara os requisitos descritos anteriormente, estão elaborados dois cenários de uso que representam situações reais, cujo fluxo de interação é mais complexo, sendo um exemplo do aplicativo mobile e um do Dashboard.

Cenário 1: Registo de uma queixa por parte de um utilizador que abre a aplicação pela primeira vez.

- **Ator Principal**: Utilizador da aplicação móvel.
- **Descrição**: Um utilizador está insatisfeito com a manutenção de uma ciclovia na sua cidade e deseja apresentar uma queixa.
- **Fluxo de Iterações:**
    1. O utilizador abre a aplicação pela primeira vez e encontra uma página de registo, onde seleciona a opção de continuar com o Google.
    2. Após o registo preenche o pequeno formulário sobre dados adicionais relativos ao utilizador.
    3. Na página inicial seleciona na barra de navegação a opção de “Criar Queixa”.
    4. Seleciona a categoria de Mobilidade na primeira página que aparece.
    5. De seguida, seleciona uma tipologia de Ciclovia.
    6. Numa lista em outra página, com todas as queixas possíveis seleciona “Manutenção de Ciclovia”.
    7. Já numa página onde pode adicionar informação extra, escreve uma pequena descrição e seleciona a data e hora, confirmando também a localização.
    8. Na página seguinte, pode adicionar uma fotografia com anexo à queixa e confirma que aceita a divulgação dos seus dados para a entidade responsável.
    9. No resumo da queixa antes da submissão confirma que está tudo bem e submete a queixa.
    10. Recebe um pop-up que indica o sucesso da submissão da queixa.

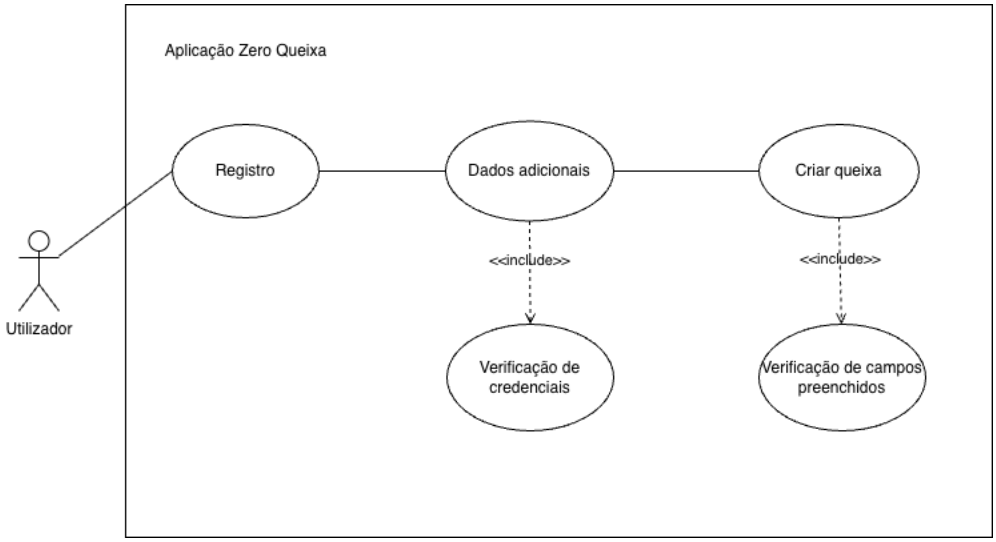

Figura 1 - Diagrama de Use Case Registo de uma queixa

Cenário 2: Agrupar queixar relacionadas.

- **Ator Principal**: Administração (BackOffice).
- **Descrição**: Sistema identifica queixas com temas semelhantes e o administrador agrupa a queixa.
- **Fluxo de Iterações:**
    1. O administrador acede ao Dashboard através da página do login, ao colocar as suas credenciais.
    2. Seleciona a opção de “Lista de Queixas”.
    3. O algoritmo de similaridade é executado.
    4. O sistema sugere queixas com assuntos e categorias parecidos apresenta a percentagem de similaridade.
    5. O administrador agrupa a queixa após validar a sugestão do sistema.

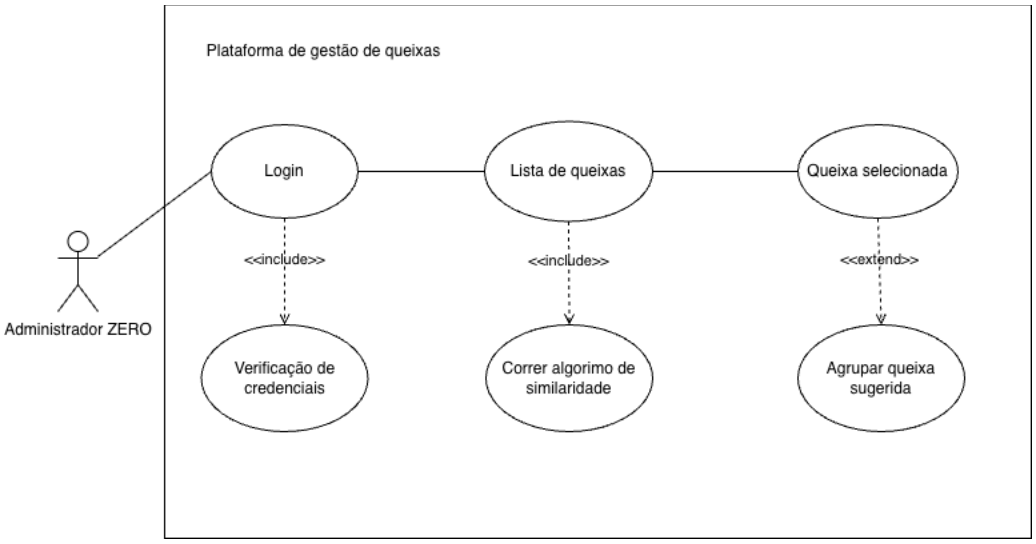

Figura 2 - Diagrama Use Case Agrupamento de queixa

3.2 Modelação

Para estruturar a base de dados do Zero Queixa e garantir a organização e integridade dos dados, foi elaborado um Diagrama Entidade-Relação (DER). Este diagrama reflete a estrutura de armazenamento de dados necessários para o funcionamento da aplicação e foi desenvolvido de acordo com os requisitos levantados durante as reuniões.

O modelo segue boas práticas de normalização, estando na 3ª Forma normal (Beeri et al., 1989), o que minimiza inconsistências, tendo todas as tabelas e relações modeladas para suportar as funcionalidades, tanto da aplicação móvel como do BackOffice.

Entre as principais tabelas destacam-se:

- **User**: Armazena as informações dos utilizadores, como o nome, email e número de identificação civil. É essencial para quando o utilizador fizer uma queixa, não ter de preencher novamente estes dados
- **Staff_Users**: Representa os utilizadores administrativos do backoffice. Estes utilizadores são então atribuídos a funções que correspondem às suas atribuições, garantindo que tenham acesso apenas aos recursos e ações necessários para o desempenho de suas responsabilidades.
- **Queixa**: Regista as queixas submetidas, com informações de localização, categoria e estado. É a tabela que considero como ponto central da minha modelação.
- **Categoria**: Organiza as categorias e subcategorias das queixas.
- **Estado_Queixa**: Permite o acompanhamento do histórico de estados de cada queixa. Vai facilitar o trabalho intermediário da ligação do BackOffice à aplicação mobile.
- **Entidade**: Representa as entidades responsáveis pelo tratamento das queixas por cada categoria.

As relações entre tabelas estão representadas através de chaves primárias e estrangeiras, assegurando a integridade referencial.

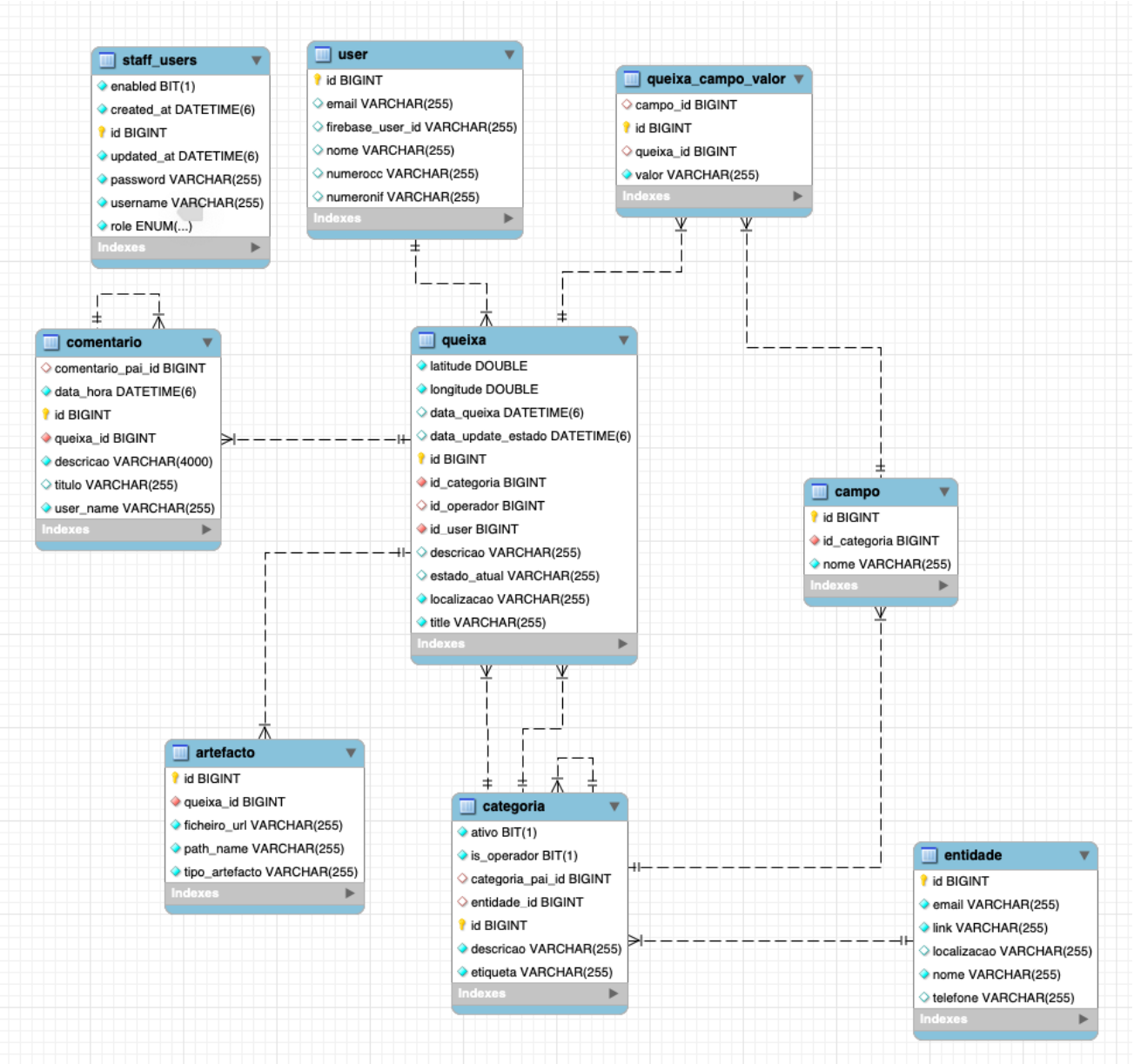

Figura 3 - Diagrama de relações

Todas as tabelas e relações foram concebidas de raiz para este projeto, não havendo inconsistências estruturais significativas, visto que a modelação procura equilibrar a simplicidade e eficiência, para suportar tanto o lado do utilizador quanto o lado administrativo.

3.2.1 Processamento das categorias

As categorias de queixa constituem uma função central no funcionamento da aplicação e têm a necessidade de permitir adaptações futuras sem intervenção direta no código.

Para responder a esse desafio, foi desenvolvido um sistema que permite alimentar a base de dados com categorias a partir de um ficheiro Excel estruturado. Esse ficheiro é preenchido por elementos da associação ZERO e serve como fonte principal para a criação e organização das categorias. A estrutura inclui campos como nome da categoria, hierarquia, estado de ativação e tipo de queixa associado.

A aplicação backend lê este ficheiro em CSV através de um script programado de forma recursiva, transforma-o num modelo de dados relacional e carrega automaticamente a informação para a base de dados. No âmbito deste trabalho, o sistema foi significativamente expandido em relação ao MVP: foram adicionados dois novos campos ao ficheiro, implementadas regras de importação mais robustas com validação e controlo de integridade, disponibilizada uma interface no backoffice para carregamento do ficheiro sem necessidade de intervenção técnica, e desenvolvida uma ferramenta auxiliar de visualização da hierarquia de categorias. Estas contribuições são descritas em detalhe nas subsecções seguintes.

3.2.1.1 Novos campos

Este ficheiro, nesta versão de testes, sofreu algumas reestruturações ligeiras, incluindo a adição de dois novos campos/colunas.

O primeiro é o **Estado**, que indica se uma categoria está ativa ou não. As categorias desativadas deixam simplesmente de aparecer no momento da criação de novas queixas, mas as queixas que foram criadas enquanto a categoria estava ativa continuam a existir normalmente.

O segundo campo é o dos **Campos Obrigatórios**. Aqui são listados todos os campos que devem ser preenchidos para que uma queixa seja considerada completa.

Tanto a aplicação móvel como o backoffice web foram desenvolvidos para se adaptarem automaticamente às categorias que existem na base de dados. Assim, qualquer nova categoria incluída e ativada via Excel passa imediatamente a estar visível na app e a ser usada pelos utilizadores na criação de queixas.

	A	B	C	D	E	F	G	H	I
1	**Nível 1**	**Nível 2**	**Nível 3**	**Nível 4**	**Nível 5**	**Nível 6**	**Nível 7**	**Estado**	**Campos obrigatórios**
2	1						Transporte Público	Ativo	
3	1	1					Autocarro / Elétrico	Ativo	
4	1	1	1				Operadores	Ativo	
5	1	1	1	1			Carris	Ativo	
6	1	1	1	2			Carris Metropolitana	Ativo	
7	1	1	1	3			MobiCascais	Ativo	
8	1	1	1	4			Transportes Coletivos do Barreiro	Ativo	
9	1	1	2				Categoria de queixas	Ativo	
10	1	1	2	1			Carreira	Ativo	
11	1	1	2	1	1		Atraso	Ativo	Carreira; Sentido; dia/hora
12	1	1	2	1	2		Serviço cancelado sem aviso	Ativo	Carreira; Sentido; dia/hora

Figura 4 - Parte do Excel para processamento de categorias

3.2.1.2 Estrutura do ficheiro CSV

O ficheiro é composto por nove colunas. As primeiras seis colunas — designadas Nível 1 a Nível 6 — definem a posição do nó na hierarquia da árvore de categorias através de valores numéricos. A coluna Nível 7 contém sempre o nome ou descrição da categoria, independentemente da profundidade real do nó. As duas colunas restantes definem o estado de ativação da categoria e os campos obrigatórios a preencher pelo utilizador ao submeter uma queixa nessa categoria, separados por ponto e vírgula.

A profundidade real de cada nó na árvore é determinada pelo último valor numérico não vazio entre os níveis 1 e 6. O sistema gera automaticamente uma etiqueta interna para cada categoria, concatenando os valores dos níveis com um ponto — por exemplo, `1.1.2.1.3.1`. Esta etiqueta é utilizada para evitar duplicados durante a importação: se uma categoria com a mesma etiqueta já existir na base de dados, não é criada novamente.

A distinção entre sub-árvores de operadores e sub-árvores de categorias de queixas é determinada pelo valor do Nível 3. Quando este vale `1`, a sub-árvore correspondente representa operadores, como por exemplo Carris, CP ou Transtejo. Quando vale `2`, representa categorias de queixas propriamente ditas. Esta convenção permite que a aplicação móvel e o backoffice tratem os dois tipos de nó de forma diferenciada.

O trecho seguinte ilustra a estrutura de uma sub-árvore de Autocarro / Elétrico:

Estrutura CSV:

Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7	Estado	Campos obrigatórios
1						Transporte Público	Ativo	
1	1					Autocarro / Elétrico	Ativo	
1	1	1				Operadores	Ativo	
1	1	1	1			Carris	Ativo	
1	1	2				Categoria de queixas	Ativo	
1	1	2	1			Carreira	Ativo	
1	1	2	1	1		Atraso	Ativo	Carreira; Sentido; dia/hora

Tabela 4 - Exemplo de estrutura do ficheiro CSV de categorias

Que gera a seguinte árvore:

```
Transporte Público
 └── Autocarro / Elétrico
      ├── Operadores
      │    └── Carris
      └── Categoria de queixas
           └── Carreira
                └── Atraso   [campos: Carreira, Sentido, dia/hora]
```

3.2.1.3 Regras de importação

É importante referir que, após o ficheiro ser carregado, nem a descrição da categoria nem os campos obrigatórios podem ser alterados. Esta restrição existe para garantir a integridade das queixas já submetidas: uma queixa fica associada a uma categoria no momento da sua criação, e alterar retroativamente a descrição ou os campos obrigatórios dessa categoria comprometeria a consistência dos dados históricos. Caso seja necessário atualizar estas informações, recomenda-se marcar a categoria anterior como inativa através da coluna Estado e criar uma nova categoria com os dados corretos.

Durante a importação, o sistema verifica se cada categoria já existe na base de dados com base na sua etiqueta interna. Caso não exista, é inserida. Caso já exista, o sistema verifica se os dados são idênticos — e não faz nada — ou se o estado foi alterado, caso em que efetua uma atualização. Se qualquer outro campo tiver sido modificado, o sistema rejeita a importação e devolve um erro, forçando o utilizador a corrigir o ficheiro. O sistema verifica também se o próprio ficheiro contém etiquetas duplicadas, rejeitando a importação com rollback caso sejam detetadas.

3.2.1.4 Carregamento pelo backoffice

O carregamento do ficheiro CSV é feito diretamente pelo backoffice, através de uma interface dedicada que permite à equipa da ZERO fazer o upload do ficheiro sem necessidade de qualquer conhecimento técnico. Após o carregamento, o backoffice apresenta as categorias importadas para verificação.

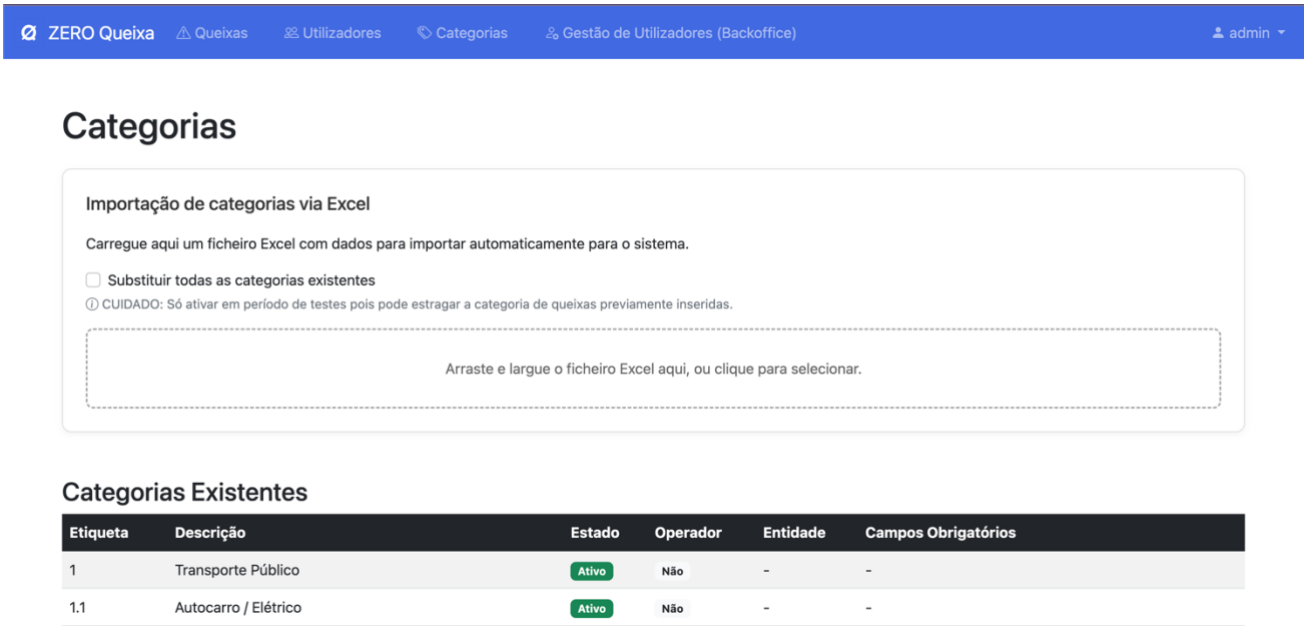

Figura 5 - Interface do backoffice para carregamento do ficheiro CSV de categorias

3.2.1.5 Visualizador de categorias

Dada a limitação de visualizar a hierarquia de categorias diretamente no ficheiro CSV — onde a estrutura em árvore não é imediatamente percetível —, foi desenvolvida no âmbito deste trabalho uma ferramenta auxiliar de visualização. Trata-se de uma aplicação web que permite carregar o ficheiro CSV e navegar pela árvore de categorias de forma interativa, consultando para cada nó a sua etiqueta interna, o estado de ativação, o tipo (operador ou categoria), os campos obrigatórios e os filhos diretos. A ferramenta suporta ainda pesquisa por nome de categoria e expansão ou colapso de toda a árvore.

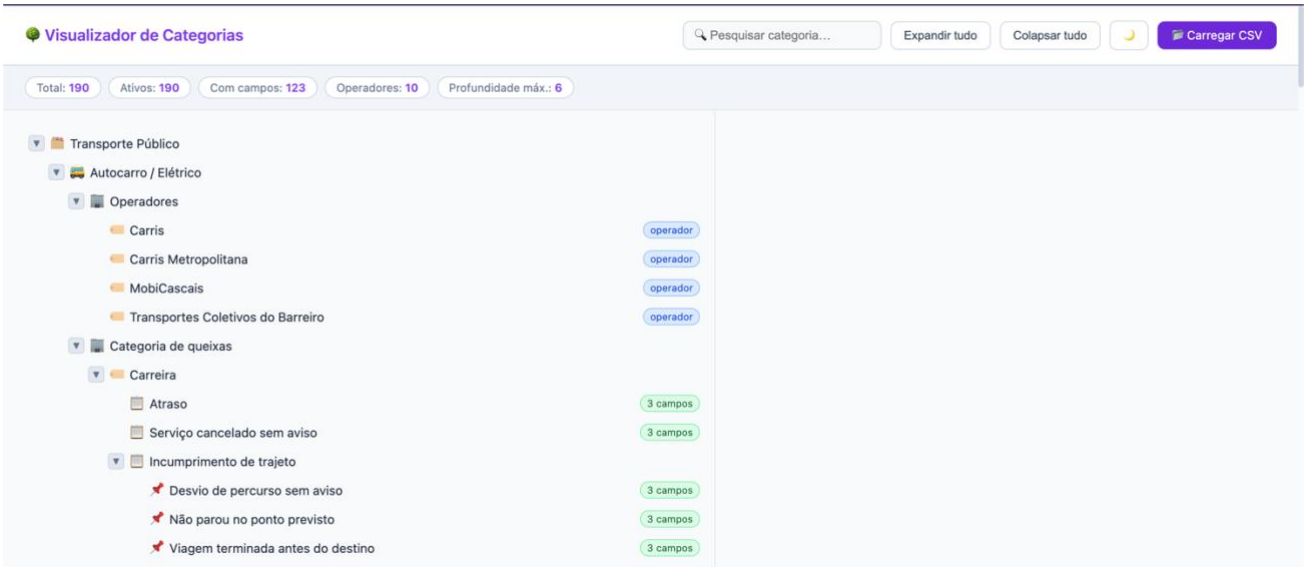

Figura 6 - Visualizador de categorias com a árvore expandida

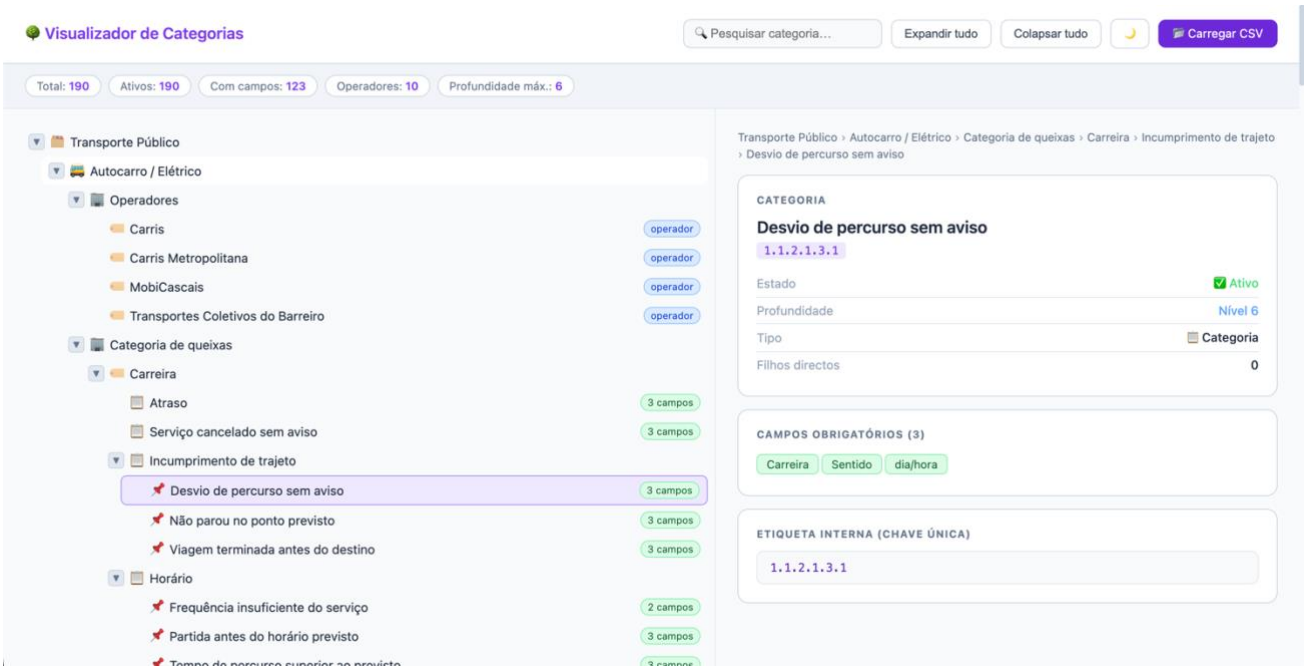

Figura 7 - Painel de detalhe de uma categoria com campos obrigatórios visíveis

Esta ferramenta foi utilizada ao longo do desenvolvimento para validar a estrutura do ficheiro CSV antes de cada importação, evitando erros de hierarquia que de outra forma só seriam detetados após o carregamento na base de dados. De momento, a ferramenta existe como utilitário independente e não se encontra integrada no backoffice, estando prevista a sua inclusão numa atualização futura.

4 Solução Desenvolvida

4.1 Apresentação

A solução desenvolvida para o projeto *ZeroQueixa* consiste numa aplicação móvel multiplataforma, acompanhada por uma aplicação web que permite à equipa de BackOffice da ZERO gerir e acompanhar as queixas submetidas pelos cidadãos. O principal objetivo é proporcionar aos utilizadores um meio simples, acessível e centralizado para registar, consultar e acompanhar queixas relacionadas com mobilidade e proteção ambiental, enquanto a associação ZERO dispõe de ferramentas eficazes para tratamento e análise dessas mesmas queixas.

Tal como previsto inicialmente, o desenvolvimento é dividido em duas fases. A primeira fase corresponde ao **MVP**, que valida o conceito e estabelece a base tecnológica e funcional do sistema. A versão **2.0** introduz melhorias estruturais, incorporação das funcionalidades sociais e mecanismos inteligentes, incluindo análise de similaridade, notificações, exportação de registos e outras funcionalidades evolutivas.

Nesta fase, o objetivo é manter a solução original e reforçá-la com práticas e componentes modernos que aumentam qualidade, organização interna e escalabilidade. Esta melhoria reflete-se sobretudo na adoção de padrões arquiteturais mais robustos e na introdução de mecanismos inteligentes, como o sistema de recomendação baseado em análise textual.

Neste capítulo apresentam-se toda a arquitetura detalhada dos componentes, as tecnologias adotadas, os padrões aplicados e os ambientes considerados mais adequados para validação e evolução da solução.

4.2 Arquitetura

A solução segue uma arquitetura **cliente-servidor**, conforme representado na Figura 5. A aplicação móvel comunica com um backend que centraliza toda a lógica de negócio, acede à base de dados e expõe WebServices REST via HTTPS. O BackOffice, através da aplicação web, interage diretamente com o servidor, onde são executadas queries e operações sobre os dados para apoio à gestão das queixas.

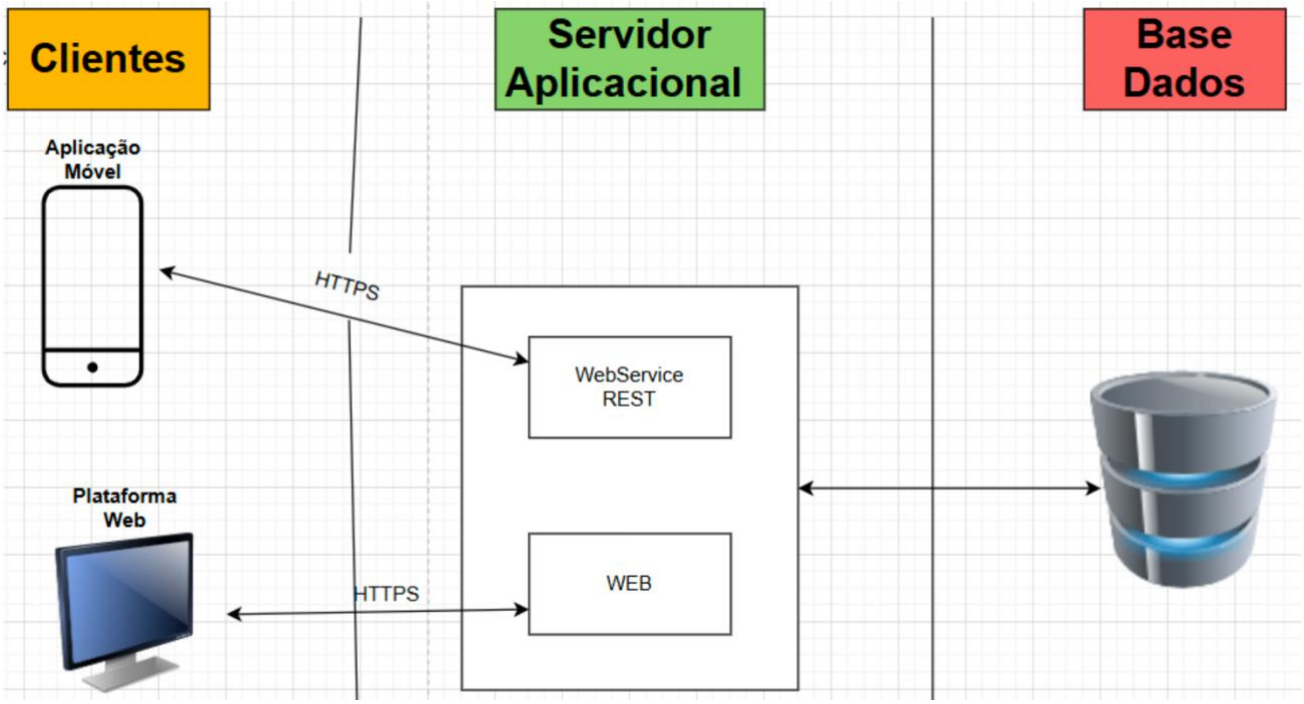

Figura 8 - Arquitetura do ZeroQueixa

A componente móvel utiliza comunicação encriptada via HTTPS. Durante o desenvolvimento foi identificada e corrigida uma questão relacionada com a definição do **endereço canónico (canonical address)**, que causava falhas na validação SSL em ambiente de testes. Este problema foi resolvido mediante a correta configuração do domínio, ajuste dos registos DNS e alinhamento entre o certificado SSL e o domínio utilizado pela aplicação.

4.3 Tecnologias e Ferramentas Utilizadas

O sistema mantém a divisão clara em FrontEnd, BackEnd e Base de Dados, com tecnologias adaptadas a cada propósito, agora complementadas com componentes adicionais que suportam novas funcionalidades avançadas.

4.3.1 FrontEnd (Cliente)

A aplicação móvel foi desenvolvida em **Flutter** (Aung et al., 2024), utilizando a linguagem **Dart** e seguindo boas práticas de modularização e separação de responsabilidades.

Principais práticas aplicadas:

- **Provider:** Usado para gestão de estado e injeção de dependências e para a gestão do estado da aplicação
- **Repositórios:** Toda a lógica de comunicação com as fontes de dados (API e base de dados locais) foi centralizada em classes de tipo Repository, promovendo uma camada de abstração entre a interface do utilizador e os dados.
- **Separação UI / Lógica:** A interface gráfica foi organizada em Widgets reutilizáveis, enquanto a lógica e validação de formulários, pedidos HTTP e transformação de dados está encapsulada noutras camadas.
- **Componentização:** Criação de componentes reutilizáveis (botões, cartões, campos), promovendo consistência visual e produtividade.

4.3.2 BackOffice Web

A aplicação web mantém a abordagem simples e focada:

- HTML, CSS, Bootstrap
- Renderização no lado do servidor
- Interfaces claras para consulta e gestão de queixasBackEnd (Servidor)

O backend, originalmente implementado em **Spring Boot com Kotlin**, mantém a sua estrutura modular:

- **controller**: Contém os endpoints REST e as rotas da aplicação.
- **service**: Responsável pela lógica de negócio da aplicação.
- **repository**: Interfaces que fazem a ligação entre o serviço e a base de dados (utilizando Spring Data JPA).
- **entity**: Classes que representam as entidades da base de dados.
- **dto**: Objetos de transferência de dados usados para transportar dados entre camadas sem expor diretamente as entidades.

4.3.3 Base de Dados

A base de dados continua a ser **MySQL**, modelada de forma relacional com suporte para:

- entidades principais (Utilizador, Queixa, Categoria),
- novas tabelas introduzidas pelas funcionalidades adicionadas,
- relações hierárquicas nas categorias.

A estrutura foi projetada para permitir escalabilidade e futura integração com mecanismos analíticos.

4.4 Ambientes de Teste e de Produção

4.4.1 Ambiente de Desenvolvimento

O ambiente de desenvolvimento encontra-se distribuído por dois equipamentos com funções distintas. O backend, desenvolvido em Spring Boot com base de dados MySQL, é executado num computador HP 15 com Ubuntu 24.04 LTS, 16 GB de RAM e 1 TB SSD. A aplicação móvel Flutter é desenvolvida num MacBook Air M1, com macOS Sonoma, utilizando Android Studio e Xcode, o que permite testar simultaneamente em ambiente iOS e Android. Ambos os equipamentos comunicam numa sub-rede Wi-Fi dedicada a 5 GHz.

4.4.2 Ambiente de Testes

O ambiente de testes mantém a estrutura conceptual definida inicialmente, com pequenas adaptações:

O backend está alojado num servidor da **Universidade Lusófona**, permitindo validar:

- acessibilidade externa;
- configuração de DNS;
- integração SSL sem erros.

A aplicação móvel está validada nas:

- **Google Play – Internal Testing** (Android)
- **TestFlight** (iOS) possibilitando distribuição controlada para testes reais em dispositivos físicos.

4.4.3 Ambiente de Produção

O ambiente de produção encontra-se alojado num servidor da Universidade Lusófona, numa máquina virtual com 4 vCPU, 4 GB de RAM e 50 GB de armazenamento em disco, a correr Ubuntu Server 20.04.2 LTS x64. O backend e o BackOffice estão instalados nesta infraestrutura, garantindo disponibilidade externa e separação lógica do ambiente de desenvolvimento.

A aplicação móvel encontra-se disponível publicamente na Google Play Store e na Apple App Store, ambas associadas à conta da associação ZERO.

4.5 Abrangência

Neste projeto serão utilizados conhecimentos das Unidades Curriculares Computação Móvel e Linguagens de Programação II, ligado ao desenvolvimento móvel e programação, também fundamentos adquiridos em Computação Distribuída para desenho e implementação de Web Services. Com a ajuda da Engenharia de Software e Interação Humano-Máquina existe uma melhor interação entre os utilizadores e a aplicação. Por fim, serão também utilizados conhecimentos de Base de Dados para desenho e implementação de bases de dados relacionais.

4.6 Evolução face ao MVP

O presente capítulo descreve o conjunto de contribuições desenvolvidas ao longo deste trabalho, tendo como ponto de partida o MVP entregue por Rui Ferreira. Enquanto a secção 1.1 contextualiza o estado inicial do projeto, esta secção detalha as alterações efetivamente realizadas, abrangendo a refactorização técnica do backend, o redesign da API REST, as melhorias funcionais e de interface da aplicação móvel, e as adaptações decorrentes do processo de validação com utilizadores reais e com a associação ZERO.

4.6.1 Backend e Backoffice

O desenvolvimento do backend incidiu sobre quatro áreas principais: refactorização da arquitectura, redesign da API REST, melhorias de infraestrutura e implementação de funcionalidades de privacidade.

No que respeita à arquitetura, a camada de controlo encontrava-se organizada em controladores monolíticos, onde um único controlador acumulava responsabilidades sobre utilizadores, queixas e categorias em simultâneo.

```
29  open class UserAPIController(...) {                                                    3 4 135
37
38      @ExceptionHandler( ...value = Exception::class)  Pedro Alves
39      open fun handleRuntimeException(ex: Exception): ResponseEntity<String> {...}
42
43      @GetMapping( ...value = "/test")  Pedro Alves
44      open fun test(): String {...}
47
48      @GetMapping( ...value = "/user/info")  Pedro Alves
        open fun getUserInfo(principal: Principal): User {...}
57
58      @PostMapping( ...value = "/user/register")  Pedro Alves +1
59      open fun registerUser(@RequestBody registerUserRequest: RegisterUserRequest,
                              principal: Principal): ResponseEntity<String> {...}
74
75
76      //Lista de queixas do Utilizador
77      @GetMapping( ...value = "/user/queixas")  RuiFerreira +2
        open fun listMyQueixas(principal: Principal): List<Map<String, Any?>> {...}
125
126     //Informação da Queixa Selecionada
127     @GetMapping( ...value = "/queixa/info/{id}")  RuiFerreira +2
128     open fun infoQueixa(@PathVariable id: Long): Map<String, Any?> {...}
171
172
173     //Todas as queixas da tabela
174     @GetMapping( ...value = "/queixas")  RuiFerreira +1
175     open fun infoQueixa(principal: Principal): List<Map<String, Any?>> {...}
218
219
220     //Criar Queixa
221     @PostMapping( ...value = "/queixa/register")  RuiFerreira +2
222     open fun createQueixa(@RequestBody registerQueixaRequest: RegisterQueixaRequest, principal: Principal): ResponseEntity<Map<String,*>
277
278
279     @GetMapping( ...value = "/categorias")  RuiFerreira +3
280     open fun listCategorias(): List<Map<String, Any?>> {...}
294
295     @GetMapping( ...value = "/queixa/artefato/{queixaId}/{fileName}")  RuiFerreira +1
296     open fun queixaArtifact(@PathVariable queixaId: Long, @PathVariable fileName: String, ): ResponseEntity<ByteArray> {...}
321
322     open fun processarImagem(base64Image: String, tipoMime: String, queixa: Queixa): Artefacto {  1 Usage  RuiFerreira
```

Figura 9 - Estado inicial da camada de controlo da API REST, evidenciando a ausência de separação de responsabilidades: endpoints de três domínios distintos (utilizadores, queixas e categorias) concentrados num único controlador, sem versionamento de API

Esta organização dificultava a manutenção e a escrita de testes isolados por funcionalidade. Para resolver esta situação, a camada de controlo foi reorganizada numa estrutura modular, com separação clara entre os controladores da API e os controladores do backoffice web. Foram criados três controladores principais sob o prefixo `/api/v1`: `CategoriasController`, `QueixasController` e `UsersController`, cada um responsável pelo seu domínio específico.

O redesign da API REST surgiu da necessidade de corrigir várias inconsistências de design presentes na versão herdada do MVP. A nomenclatura das rotas era irregular, alternando entre singular e plural, e os paths incluíam verbos que deveriam ser expressos pelo método HTTP. Por exemplo, a criação de uma queixa era feita através de `POST /api/queixa/register`, quando a semântica REST indica que `POST /api/queixas` é suficiente para expressar a mesma intenção. O acesso à informação de um utilizador era feito via `/api/user/info`, sem qualquer distinção entre o utilizador autenticado e outros recursos. Adicionalmente, algumas rotas não seguiam o prefixo comum `/api`, como era o caso de `POST /categorias/upload`. O redesign introduziu versionamento explícito com o prefixo `/api/v1`, adoptou substantivos no plural nos caminhos dos recursos, eliminou verbos dos paths, introduziu o sub-recurso `/me` para o utilizador autenticado e reorganizou os sub-recursos de forma hierárquica. A tabela seguinte resume as principais alterações:

Antes	**Depois**	**Motivo**
`GET /api/queixa/info/{id}`	`GET /api/v1/queixas/{id}`	Singular → plural; removido `/info` desnecessário
`POST /api/queixa/register`	`POST /api/v1/queixas`	`POST` já implica criação; removido verbo do path
`GET /api/user/info`	`GET /api/v1/users/me`	`/me` é a convenção para o utilizador autenticado
`GET /api/user/queixas`	`GET /api/v1/users/me/queixas`	Sub-recurso do utilizador autenticado
`POST /categorias/upload`	`POST /api/v1/categorias/upload`	Unificado sob o prefixo `/api/v1`
`GET /api/queixa/artefato/{queixaId}/{fileName}`	`GET /api/v1/queixas/{id}/artefatos/{fileName}`	Sub-recurso hierárquico correcto

Tabela 5 - Evolução dos endpoints no backend

Na área de infraestrutura, foi configurado um pipeline de CI/CD com GitHub Actions, com execução automática de testes em pull requests e integração da ferramenta Qodana para análise estática de qualidade de código. A imagem base do Docker foi actualizada e foram adicionadas excepções personalizadas (`NotFoundException` e `UnsupportedFileTypeException`) para um tratamento de erros mais robusto.

Por fim, foi implementada no backoffice a visualização do consentimento do utilizador no ecrã de detalhe de queixa. Esta funcionalidade tornou-se necessária para que os operadores da ZERO consigam, no momento de encaminhar uma queixa para a entidade responsável, saber se o queixoso autorizou ou não a divulgação da sua identidade. Foram adicionados indicadores visuais que distinguem claramente os dois casos.

4.6.1.1 Testes

A cobertura de testes do backend foi alargada e reestruturada em paralelo com a reorganização da camada de controlo. Na versão herdada do MVP existiam apenas testes de integração monolíticos (`IntegrationTests.kt`) e testes unitários à lógica de serviço das categorias (`CategoriaServiceTest.kt`), sem qualquer separação por controlador. Com a introdução dos novos controladores da API v1, foram criados ficheiros de teste dedicados a cada um deles: `CategoriasControllerTest.kt`, `QueixasControllerTest.kt` e `UsersControllerTest.kt`.

Para isolar os testes da base de dados MySQL de produção, foi configurada uma base de dados H2 em memória com um ficheiro de propriedades dedicado. Foi também necessário desativar o Firebase e substituí-lo por tokens de desenvolvimento estáticos, uma vez que o ambiente de testes não tem acesso à autenticação externa.

Os testes unitários ao serviço de categorias cobrem os seguintes cenários: importação correta de uma hierarquia com dois níveis de folhas, deteção de etiquetas duplicadas no ficheiro CSV com lançamento de exceção, mapeamento do estado `Apagado` para `ativo = false`, idempotência na reimportação do mesmo ficheiro e actualização incremental quando um segundo ficheiro introduz novas categorias.

4.6.2 Aplicação Móvel

O desenvolvimento da aplicação móvel abrangeu diversas áreas, desde a configuração inicial até melhorias progressivas de funcionalidade, interface e experiência do utilizador.

Na fase inicial, foram adicionados os ícones e o ecrã de arranque, implementado o suporte a campos dinâmicos na queixa incluindo o campo de localização, e atualizada a URL base para o ambiente de produção.

Ao nível da interface, foram realizadas melhorias gráficas no ecrã final de registo de queixa. As categorias do primeiro passo, que anteriormente eram apresentadas em grelha, passaram a ser exibidas numa única coluna, tornando a leitura mais clara em dispositivos móveis.

Os itens da dropdown de queixas passaram igualmente a ocupar uma única linha com separador visual entre eles. Foi também corrigida a apresentação de categorias inativas, que estavam a ser mostradas ao utilizador indevidamente, e efetuadas diversas correções textuais e de ícones, algumas das quais identificadas diretamente pela representante da associação ZERO durante o período de testes internos.

No que respeita à funcionalidade, tornou-se possível editar os dados do utilizador e introduzir manualmente a localização da ocorrência, para os casos em que o auto-complete não devolvia o resultado pretendido.

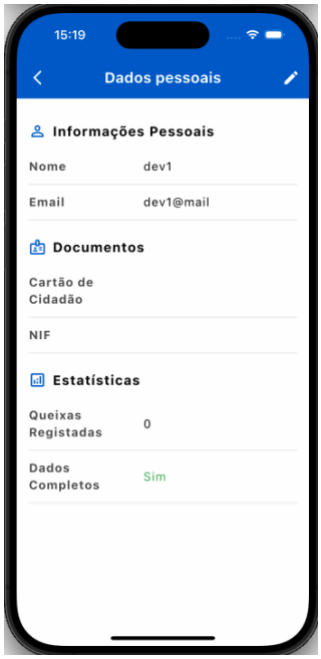

Figura 10 - Novo ecrã com informação pessoal do utilizador, com possibilidade de alterá-los

O auto-complete de moradas passou também a restringir-se à Área Metropolitana de Lisboa, dado ser a zona geográfica de atuação da aplicação. Complementarmente, foi adicionado um aviso para queixas registadas fora desta área, informando o utilizador de que a aplicação ainda não está disponível nessa zona, embora permita registar a queixa para fins estatísticos. Foi ainda implementada uma verificação no arranque que alerta o utilizador caso não tenha a versão mais recente instalada, garantindo que todos os utilizadores beneficiam das correções mais recentes.

Relativamente à privacidade e conformidade, a política de privacidade da aplicação foi elaborada em conjunto com a associação ZERO e validada por uma consultora de RGPD. Na sequência dessa validação, foi adicionado à página de login um campo de consentimento obrigatório, conforme recomendação da consultora, que exige que o utilizador declare expressamente ter lido e aceite a política de privacidade antes de poder criar conta. A foto de perfil foi igualmente removida da aplicação, uma vez que a conta Apple não permite acesso à

fotografia do utilizador e a funcionalidade não acrescentava valor nesta fase. Foi ainda corrigido um problema em que a indicação de consentimento do utilizador não estava a ser enviada juntamente com a queixa, o que comprometia a capacidade da ZERO de saber se poderia ou não divulgar a identidade do queixoso ao encaminhar a reclamação.

Por último, foram corrigidos problemas de acessibilidade relacionados com o aumento de tamanho de fonte, a perda de foco no campo de morada e a validação dos campos obrigatórios na submissão da queixa.

4.6.2.1 Evolução dos ecrãs

Encontra-se em seguida os ecrãs que sofreram alterações nesta fase do projeto:

Figura 11 - Ecrã inicial (sem elementos)

Figura 12 - Ecrã inicial (atualizado)

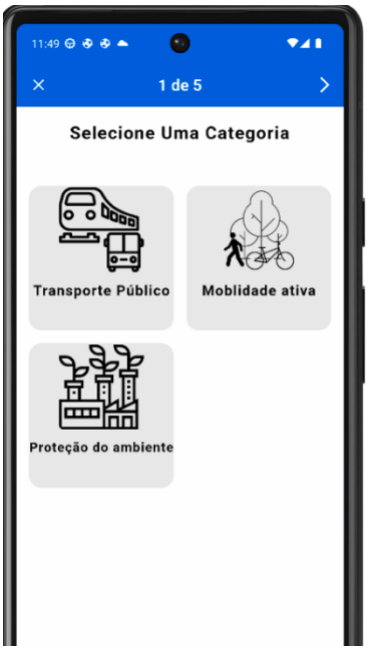

Figura 13 - Categoria de seleção de categoria (elementos em grelha)

Figura 14 - Categoria de seleção de categoria (elementos em coluna)

Figura 15 - Ecrã de resumo da queixa (antigo)

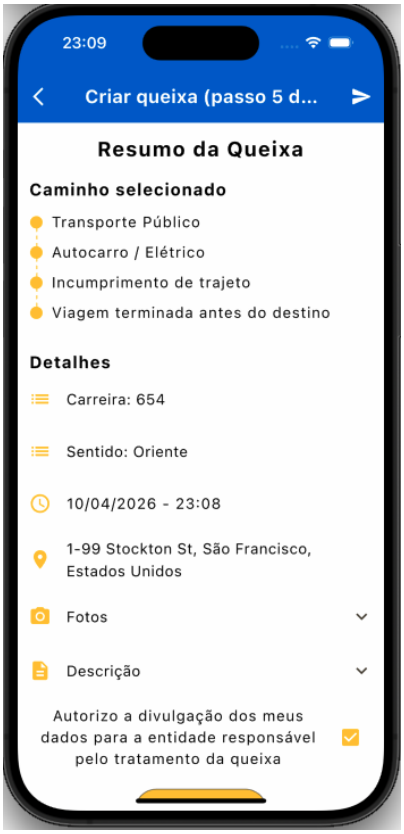

Figura 16 - Ecrã de resumo da queixa (atualizado)

Método e Planeamento

4.7 Planeamento inicial

O desenvolvimento do projeto ZeroQueixa segue uma abordagem Agile para garantir flexibilidade e alinhamento às metas estabelecidas ao longo do período letivo de desenvolvimento. O plano inicial foi elaborado com base nos requisitos técnicos descritos no capítulo 3 e dividiu as tarefas entregáveis com prioridades definidas.

O calendário elaborado guiará o desenvolvimento da aplicação, para que todas as entregas sejam realizadas de acordo com o regulamento da Unidade Curricular Trabalho Final De Curso. O calendário criado foi dividido em 6 fases:

Fase 1: Correção de Bugs e Aperfeiçoamento das funcionalidades

Fase 2: Atualizações na UI (Fase atual).

Fase 3: Início dos testes em massa

Fase 4: Correção dos erros identificados durante os testes

Fase 5: Desenvolvimento da versão 2.0

Fase 6: Revisão e lançamento

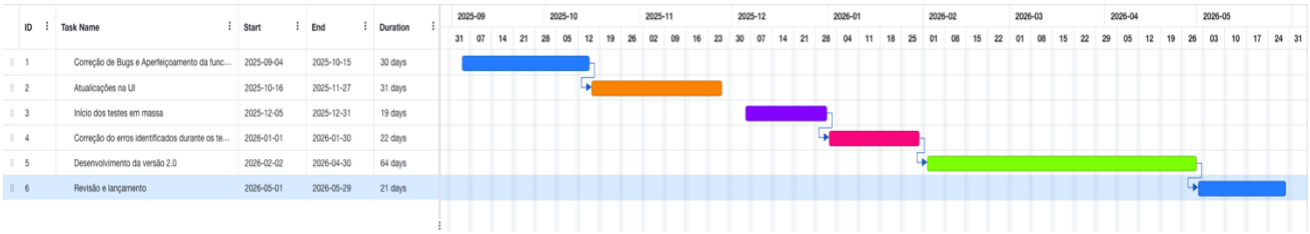

Figura 17 - Diagrama de Gantt

4.7.1 Desvios ao Planeamento e Factores Externos

O plano inicial previa o início da Fase 3 — testes em massa — em dezembro de 2025. Contudo, o arranque efetivo dos testes apenas ocorreu em fevereiro de 2026, com um atraso de aproximadamente dois meses. Este desvio resultou exclusivamente de fatores externos ao desenvolvimento, que se descrevem de seguida.

O primeiro fator foi a necessidade de publicar a política de privacidade num endereço web acessível publicamente, requisito obrigatório imposto pela Google Play Store e pela Apple App Store para a transição para o modo de testes abertos. A elaboração deste documento exigiu a consulta de uma especialista em RGPD, cujo parecer final foi recebido a 15 de dezembro de 2025. A publicação do documento no site da associação ZERO ocorreu apenas a 23 de janeiro de 2026, após coordenação interna da associação para disponibilizar o conteúdo no seu website.

O segundo fator foi o próprio processo de aprovação das lojas. Após a submissão para open testing, tanto a Google Play Store como a Apple App Store sujeitam as aplicações a um processo de revisão que pode demorar vários dias. Os links públicos de instalação ficaram disponíveis a 30 de janeiro de 2026.

Com a aplicação disponível publicamente, a divulgação foi iniciada pela equipa da ZERO no seu canal de Discord a 3 de fevereiro de 2026. Contudo, a adesão foi muito reduzida, tendo sido registada apenas uma

queixa após essa divulgação. Face a esta situação, foi necessário delinear uma estratégia alternativa, que passou por apresentações no contexto académico realizadas em março de 2026, conforme descrito em detalhe no capítulo 5.

É importante sublinhar que os atrasos descritos não decorreram de falhas no desenvolvimento técnico, que avançou de acordo com o previsto, mas de dependências externas sobre as quais não havia controlo direto: a disponibilidade da associação ZERO para tratar da política de privacidade e da sua publicação, e os processos de aprovação das plataformas de distribuição.

4.8 Processo de Desenvolvimento e Gestão de Contribuições

O desenvolvimento do projeto ZeroQueixa seguiu uma abordagem predominantemente sequencial, próxima do modelo *waterfall (What Is the Waterfall Methodology? | Atlassian, 2026)*, em que as fases de análise, implementação e validação foram executadas de forma ordenada e progressiva. Apesar de o planeamento geral contemplar iterações faseadas, a gestão do trabalho no dia a dia assentou numa disciplina de execução linear de tarefas, priorizadas por criticidade e dependência técnica.

A plataforma central de suporte ao desenvolvimento foi o **GitHub (Bell & Beer, 2015)**, utilizado não apenas para controlo de versões, mas também como principal ferramenta de gestão de tarefas. Para o efeito, foi criada a organização **DEISI-LABS-ULHT**, sob a qual estão alojados dois repositórios distintos: **ZEROQueixa_Backend**, correspondente ao servidor e ao backoffice da aplicação, e **ZEROQueixa_Mobile**, onde reside o código da aplicação Flutter.

De forma a preservar a integridade dos repositórios principais e a promover boas práticas de desenvolvimento, foi efetuado um *fork* de cada repositório para a conta pessoal do autor. Todo o trabalho de implementação é realizado sobre estes repositórios derivados, sendo as contribuições posteriormente integradas nos repositórios da organização através de um processo formal de revisão.

A gestão de tarefas é feita através do sistema de *Issues* do GitHub. Cada *issue* representa uma unidade de trabalho — seja a correção de um *bug*, a implementação de uma funcionalidade ou uma melhoria identificada. Os *issues* são criados pelo orientador do projeto ou pelo próprio autor, sendo em ambos os casos atribuídos (*assigned*) ao autor, sinalizando que se encontra ativamente a trabalhar nessa tarefa em exclusividade.

Após a atribuição de um *issue*, o processo de desenvolvimento segue os seguintes passos:

1. **Sincronização** do repositório pessoal (*fork*) com o repositório de origem, garantindo que o ponto de partida reflita o estado mais recente da *main branch*;
2. **Criação de um branch dedicado** para o *issue* em questão, seguindo uma convenção de nomenclatura estruturada: para o backend e backoffice, os branches seguem o padrão `feature-backend-{número-do-issue}-{descrição}`; para a aplicação móvel, o padrão adotado é `feature-mob-{número-do-issue}-{descrição}`;
3. **Implementação** da solução no branch criado, de forma isolada e rastreável;
4. **Auto-revisão** (*self-review*) do código produzido, antes de qualquer submissão formal;
5. **Criação de um *Pull Request*** para o repositório da organização, submetendo as alterações à revisão do orientador;
6. **Revisão e aprovação** por parte do orientador; caso o *Pull Request* seja aprovado, é efetuado o *merge* para a *main branch*.

Esta metodologia assegura um fluxo de trabalho controlado e auditável, em que cada alteração ao código está associada a um *issue* concreto, a um *branch* identificado e a um processo de revisão documentado. A adoção de convenções de nomenclatura consistentes facilita ainda a rastreabilidade entre as tarefas planeadas e as contribuições efetivamente realizadas.

Figura 18 - Exemplo de issue (#28) criado no repositório ZEROQueixa_Backend, com descrição estruturada dos problemas identificados e objetivos da refatoração.

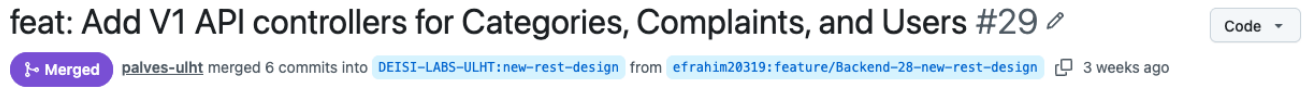

Figura 19 - Pull Request #29 resultante do issue #28, evidenciando o branch de origem (efrahim20319:feature/Backend-28-new-rest-design), o branch de destino (DEISI-LABS-ULHT:new-rest-design) e a aprovação e merge efetuados pelo orientador.

5 — Testes e Validação

A fase de testes revelou-se uma das mais exigentes e morosas de todo o projeto, tendo envolvido considerável esforço por parte da equipa de desenvolvimento com resultados aquém do esperado.

5.1 Coordenação com a Associação ZERO

Ao longo desta fase, realizaram-se várias reuniões com a equipa da Associação ZERO, que inicialmente assumiu a responsabilidade de coordenar e mobilizar testadores através do canal de Discord da associação. Contudo, a adesão nesse canal foi muito reduzida: após a divulgação inicial, em fevereiro de 2026, registou-se apenas uma queixa submetida. Face a esta estagnação, agendou-se uma nova reunião em março de 2026 para delinear uma estratégia alternativa, tendo a equipa da ZERO reconhecido a necessidade de reforçar os esforços de divulgação.

5.2 Angariação de Testadores via Contexto Académico

Perante a falta de resposta no canal da ZERO, solicitou-se ao professor orientador autorização para realizar apresentações no contexto das aulas, com o objetivo de angariar testadores junto da comunidade académica. Esta abordagem revelou-se mais produtiva: surgiram mais utilizadores registados e o volume de queixas submetidas aumentou progressivamente. Os problemas identificados durante esta fase foram registados no GitHub com uma *tag* específica criada para o efeito, permitindo rastrear os *issues* (*About Issues - GitHub Docs*, 2026) decorrentes dos testes.

5.3 Vídeo Promocional

Paralelamente, foi produzido um vídeo convidativo para incentivar a participação nos testes, divulgado nas redes sociais. Nas primeiras 24 horas após a publicação, no Instagram, o vídeo alcançou mais de 2 300 visualizações, no Linkedin, 662 impressões e 380 utilizadores alcançados — um resultado notável, que demonstrou o alcance potencial da campanha de divulgação.

5.4 Issues Identificados na Fase de Testes

A fase de testes contribuiu diretamente para a identificação de novos problemas na aplicação. Para facilitar o seu rastreamento, foi criada no GitHub a *label* **"Discovered in test phase"**, aplicada a todos os *issues* reportados durante este período. De seguida descrevem-se os três problemas identificados.

5.4.1 Crash ao negar permissão de câmara (Android)

Em dispositivos Android, foi detetado um erro fatal do tipo `CameraException` quando o utilizador negava a permissão de acesso à câmara. O problema tinha origem na função `_initializeCamera()`, que iniciava o `CameraController` sem verificar previamente se a permissão havia sido concedida. Quando o utilizador negava ou não respondia ao pedido de permissão, era lançada uma exceção `CameraAccessDenied` não tratada, resultando num crash fatal da aplicação. O evento foi registado pelo Sentry em ambiente de produção a 23 de março de 2026, na versão `1.0.0+24`, sobre Android 15.

5.4.2 Crash ao pesquisar localização com widget desmontado (iOS)

Em dispositivos iOS, foi identificado um erro fatal do tipo `TypeError` provocado pelo uso do operador de verificação nula (`!`) sobre um valor nulo dentro do pacote `google_places_flutter`. O crash ocorria quando o utilizador iniciava uma pesquisa de localização e navegava para fora do ecrã antes de a resposta assíncrona

ser recebida — momento em que o widget já havia sido desmontado e o `BuildContext` retornava nulo. O evento foi registado pelo Sentry a 20 de março de 2026, igualmente na versão `1.0.0+24`, sobre iOS.

5.4.3 Troca incorreta de foco entre campos de identificação (iOS)

Foi reportado um problema de usabilidade em iOS em que, ao preencher o campo do Cartão de Cidadão e confirmar com a tecla *Enter*, o valor introduzido era indevidamente transferido para o campo do NIF, que recebia também o foco. Este comportamento causava confusão ao utilizador e comprometia a integridade dos dados introduzidos.

5.5 Resultados Obtidos

No total, foram submetidas **25 queixas** por **17 utilizadores distintos**, no período compreendido entre 25 de novembro de 2025 e abril de 2026. As queixas abrangeram diversas categorias — desde atrasos e serviços cancelados sem aviso, a problemas de ruído, pavimento degradado e falhas em sistemas de pagamento —, e envolveram múltiplos operadores, entre os quais a Carris, a CP, o Metropolitano de Lisboa, a Carris Metropolitana, a Transtejo e a MobiCascais.

Não obstante o expressivo alcance das iniciativas de divulgação, o número de queixas ficou abaixo das expectativas. Acredita-se que a principal causa desta fraca adesão reside nas instruções fornecidas aos participantes: foi explicitamente pedido que as queixas reportadas correspondessem a situações reais e concretas, vivenciadas pelos próprios utilizadores. Este requisito revelou-se um obstáculo significativo, uma vez que nem todos os participantes se encontravam, no momento dos testes, perante uma situação que os afetasse efetivamente no domínio da mobilidade urbana ou do ambiente. Para não submeterem queixas fictícias, a maioria optou simplesmente por não participar. Os dados são elucidativos: o alcance publicitário foi considerável, mas a conversão em participação ativa ficou muito aquém do desejado.

Formulário de declaração de uso de ferramentas de Inteligência artificial

Formulário de declaração de uso de ferramentas de Inteligência Artificial a anexar a relatório

Todos os relatórios deverão incluir anexo com cópia, devidamente preenchida, do formulário abaixo.

Assinalar as opções aplicáveis e completar os campos solicitados.

1. Utilização de IA

[ ] Não foram utilizadas ferramentas de IA na realização deste trabalho.
[x] Foram utilizadas ferramentas de IA na realização deste trabalho.

---

2. Ferramentas utilizadas

Assinalar todas as que se aplicam.

Assistência geral à escrita, análise ou ideação

[x] ChatGPT
[ ] Microsoft Copilot
[ ] Gemini
[x] Claude
[ ] Perplexity
[ ] Outras. Quais? DeepSeek

Assistência à programação / desenvolvimento

[ ] GitHub Copilot
[x] Claude
[ ] OpenAI Codex
[ ] Cursor
[ ] Tabnine
[ ] Amazon CodeWhisperer / Amazon Q
[ ] Outras. Quais? Gemini

Geração de imagem / design / multimédia

[ ] DALL·E
[ ] Midjourney
[ ] Stable Diffusion
[ ] Canva AI / Magic Design
[ ] Outras. Quais? _____

Outros usos

[ ] Contexto: Ferramentas? _____

3. Fases do trabalho em que foi utilizada IA

[ ] Planeamento do trabalho
[ ] Pesquisa exploratória / levantamento inicial de informação
[ ] Documentação técnica
[x] Redação do relatório
[ ] Desenho / modelação / arquitetura
[ ] Design / prototipagem / interface
[x] Geração de código
[ ] Revisão / refatoração / debugging de código
[x] Criação de testes / casos de teste
[ ] Análise de resultados
[x] Preparação de apresentação ou materiais auxiliares
[ ] Outros. Quais? _____

4. Tipo de utilização

Descrever sucintamente como a IA foi utilizada.

Exemplos: brainstorming, estruturação de secções, revisão linguística, sugestão de arquitetura, geração de exemplos, explicação de conceitos, geração parcial de código, correção de erros, criação de casos de teste, apoio ao design.

Correção de erros, análise e revisão linguística no relatório, geração parcial de código e casos de testes.

5. Partes do trabalho afetadas

Indicar as secções, componentes, módulos, ficheiros, entregáveis ou atividades que foram influenciados pelo uso de IA.

Controllers, Views, repositório, relatório, etc.

6. Exemplos de *prompt*

Inserir exemplos de *prompt*, diferenciando por âmbito (enquadrado na questão 2) e fase (enquadrado na questão 4)

"O meu TFC está em fase de testes, crie um roteiro sobre um vídeo convidativo para que as pessoas possam testar a aplicação que estou a desenvolver, vai ser um walk trough completo, desde a aplicação móvel até o backoffice, fale sobre a Zero e a sua iniciativa, estou a considerar pelo menos 1:30 minutos de vídeo"

7. Validação, revisão e intervenção dos autores

Descrever que verificação, revisão, correção, adaptação ou reescrita foi realizada pelos autores.

Nota: se a IA tiver sido usada em código, testes, scripts, modelos, consultas, configurações ou outros artefactos técnicos, deve ser indicado de que forma os autores validaram o funcionamento e confirmaram a sua compreensão.

Testes unitários escritos por outro modelo LLM.

8. Grau de utilização

[ ] Residual
[ ] Moderado
[X] Extensivo

[ ] Utilização homogénea
[ ] Grau de uso diferenciado por fase ou componente de trabalho

Descrever sucintamente os diferentes usos.

9. Trabalhos em parceria

Protecção de dados confidenciais e recursos proprietários de parceiros

[ ] O trabalho foi realizado em parceria com entidade externa ao DEISI

No caso da resposta anterior ser verdadeira, responder às seguintes questões:

[ ] O parceiro tem regras para restringir submissão de dados
[ ] As submissões validam aplicação de regras de tratamento de dados
[ ] Foram implementados mecanismos para restringir a partilha de recursos proprietários

---

10. Declaração de responsabilidade

Ao assinarem a presente declaração, os autores declaram que:

- a informação acima é verdadeira e reflete o uso efetivo de ferramentas de IA na realização do trabalho;
- compreendem que a IA não substitui autoria nem responsabilidade académica;
- verificaram a validaram e veracidade das referências bibliográficas incluídas no relatório
- assumem integralmente a responsabilidade técnica, científica, ética e académica por todo o conteúdo submetido, incluindo texto, código, modelos, testes, imagens, diagramas e restantes artefactos entregues.

---

11. Identificação dos autores

Nome(s): Nzimba António Diandanga Tamaku

Número(s): 22409693

Data: 8 / 4 / 2026

Assinatura(s): Nzimba A. D. Tamaku

Glossário

LEI	Licenciatura em Engenharia Informática
LIG	Licenciatura em Informática de Gestão
TFC	Trabalho Final de Curso

Anexo A

Link do vídeo de convite de testadores para aplicação: https://youtu.be/e0HJ98fn6Pk

Link do vídeo de configuração da aplicação: https://youtu.be/X39oHdsN08U

6 Referencias

About issues - GitHub Docs. (2026). https://docs.github.com/en/issues/tracking-your-work-with-issues/learning-about-issues/about-issues

Ashfaque Khokhar. (2025, July 14). *Flutter Provider: The Easy Way to Manage State | by Ashfaque Khokhar | Medium*. https://medium.com/@ashfaque-khokhar/creating-a-provider-in-flutter-88a786078969#id_token=eyJhbGciOiJSUzI1NiIsImtpZCI6ImNjZTRlMDI0YTUxYWEwYzFjNDFjMWE0NTE1YTQxZGQ3ZTk2MTkzNmIiLCJ0eXAiOiJKV1QifQ.eyJpc3MiOiJodHRwczovL2FjY291bnRzLmdvb2dsZS5jb20iLCJhenAiOiIyMTYyOTYwMzU4MzQtazFrNnFlMDYwczJ0cDJhMmphbTRsamRjbXMwMHN0dGcuYXBwcy5nb29nbGV1c2VyY29udGVudC5jb20iLCJhdWQiOiIyMTYyOTYwMzU4MzQtazFrNnFlMDYwczJ0cDJhMmphbTRsamRjbXMwMHN0dGcuYXBwcy5nb29nbGV1c2VyY29udGVudC5jb20iLCJzdWIiOiIxMDc0NjMzMjE3NDgyNTUyNjU4MzQiLCJlbWFpbCI6ImVmcmFoaW10a3NAZ21haWwuY29tIiwiZW1haWxfdmVyaWZpZWQiOnRydWUsIm5vbmNlIjoibm90X3Byb3ZpZGVkIiwibmJmIjoxNzc1OTA0ODczLCJuYW1lIjoiTnNpbWJhIFRhbWFrdSIsInBpY3R1cmUiOiJodHRwczovL2xoMy5nb29nbGV1c2VyY29udGVudC5jb20vYS9BQ2c4b2NKczd1VGdGNnFqRlJDR19mUjF3Z1RoNFhFSUdQVU00WEI5aW16cE0zZmlrMUw2WDAyZWNBPXM5Ni1jIiwiZ2l2ZW5fbmFtZSI6Ik5zaW1iYSIsImZhbWlseV9uYW1lIjoiVGFtYWt1IiwiaWF0IjoxNzc1OTA1MTczLCJleHAiOjE3NzU5MDg3NzMsImp0aSI6IjgyZjMyZDQ0ODk4YTJkYWM4MmM2MDBjMzM2ZWZmOGE1ZWUwNmVlYzQifQ.P8CVaI7wEPxTTL96tBFDVg5i3V12JvrYdXYBdPIvvn8a-a2u-YmZ73-p78o2D3O3IFiFf4cVR15fCno4n5uL3_oZJp3Oa8MGt3Mu5wlExDrvOEKzAUsmsUKqWOCpGec5gEEIdyWt_jEyQNVpeesAQcZLseXUfkw7fve-vflxP6q7G8-6xTyDp2Y4fDvMm21ca3RWqYICBYIOc5OwZt3271_qh-2VNM0bOslsoJHtSkoYYFsjQKDleTwJJRafOW7dRJl42Lh3_qjC33BmqIc6cnXb_chVLxiogPOT70w7k2g5LNQvRBapS0uN6Ar3U6W0ZdoSQyIZgpv6f-YcBL5OAA

Aung, S. T., Funabiki, N., Aung, L. H., Kinari, S. A., Mentari, M., & Wai, K. H. (2024). A Study of Learning Environment for Initiating Flutter App Development Using Docker. *Information 2024, Vol. 15, Page 191*, *15*(4), 191. https://doi.org/10.3390/INFO15040191

Beeri, C., Bernstein, P. A., & Goodman, N. (1989). A SOPHISTICATE'S INTRODUCTION TO DATABASE NORMALIZATION THEORY. *Readings in Artificial Intelligence and Databases*, 468–479. https://doi.org/10.1016/B978-0-934613-53-8.50035-2

Bell, Peter., & Beer, Brent. (2015). *Introducing GitHub : a non-technical guide*.

EU Urban Mobility Observatory - Regional and Urban policy. (2024). https://regions-and-cities.ec.europa.eu/cities-portal/eu-initiatives-cities/eu-urban-mobility-observatory_en

European Environment Agency (EEA). (2026). https://www.eea.europa.eu/en

Ferreira, R., & Alves, P. (2024). *Trabalho Final de curso Relatório Intercalar 1º Semestre*.

Stevenson, D. (2018, September 24). *What is Firebase? The complete story, abridged. | by Doug Stevenson | Firebase Developers | Medium*. https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0#id_token=eyJhbGciOiJSUzI1NiIsImtpZCI6ImNjZTRlMDI0YTUxYWEwYzFjNDFjMWE0NTE1YTQxZGQ3ZTk2MTkzNmIiLCJ0eXAiOiJKV1QifQ.eyJpc3MiOiJodHRwczovL2FjY291bnRzLmdvb2dsZS5jb20iLCJhenAiOiIyMTYyOTYwMzU4MzQtazFrNnFlMDYwczJ0cDJhMmphbTRsamRjbXMwMHN0dGcuYXBwcy5nb29nbGV1c2VyY29udGVudC5jb20iLCJhdWQiOiIyMTYyOTYwMzU4MzQtazFrNnFlMDYwczJ0cDJhMmphbTRsamRjbXMwMHN0dGcuYXBwcy5nb29nbGV1c2VyY29udGVudC5jb20iLCJzdWIiOiIxMDc0NjMzMjE3NDgyNTUyNjU4MzQiLCJlbWFpbCI6ImVmcmFoaW10a3NAZ21haWwuY29tIiwiZW1haWxfdmVyaWZpZWQi

OnRydWUsIm5vbmNlIjoibm90X3Byb3ZpZGVkIiwibmJmIjoxNzc1OTA1MDYyLCJuYW1lIjoiTnNpbWJhIFRhbWFrdSIsInBpY3R1cmUiOiJodHRwczovL2xoMy5nb29nbGV1c2VyY29udGVudC5jb20vYS9BQ2c4b2NKczd1VGdGNnFqRlJDR19mUjF3Z1RoNFhFSUdQVU00WEI5aW16cE0zZmlrMUw2WDAyZWNBPXM5Ni1jIiwiZ2l2ZW5fbmFtZSI6Ik5zaW1iYSIsImZhbWlseV9uYW1lIjoiVGFtYWt1IiwiaWF0IjoxNzc1OTA1MzYyLCJleHAiOjE3NzU5MDg5NjIsImp0aSI6IjI4MjhkNTNmNzY0ZjEzOWM0MWNhYzlkOWViOTc2ZWFlYjc5ZWE5NGIifQ.Pr4yirFR5w0lVwY4umsEEsUgQt7TA8uzE28R_lLPYLDri_KQPNyigwD0CLiXfX8wgJ54_EnkOx-8N5XPc5AJbktceGA0yvMifIqmtdVRYpOxpwi1hwPW2YqjcmAbHeoBUi0rwVE6MK7kVj7Zjrs6FmYvWXKK6SRpMjkw-lpmSRVTgR871LCTCDpNZmhqo8PrOON5pdCMjsvgWV3-H1470NDubww6nCAu0gymRu3WewM_SDOCQw__kalgVWZK8XF0kyzgwwtxfhTS1xWS6jbH5n1tBRHdk8ffmPzGpOz-KjphJTP85Mb1SaGCBJMUsTsyXmGxe-wr3JtDNkA5UyqhCw

Transport and Environment Reporting Mechanism (TERM) publications | Transport and mobility | European Environment Agency (EEA). (2026). https://www.eea.europa.eu/en/topics/in-depth/transport-and-mobility/transport-and-environment-reporting-mechanism-term

What is the Waterfall Methodology? | Atlassian. (2026). https://www.atlassian.com/agile/project-management/waterfall-methodology

ZERO - Associação Sistema Terrestre Sustentável. (2025). https://zero.ong/quem-somos/