Procura.Aí: Aplicação para auxiliar na recuperação de dispositivos móveis

Matheus César Silva Figuerêdo

CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA

João Pessoa, 2025

Matheus César Silva Figuerêdo

Procura.Aí: Aplicação para auxiliar na recuperação de dispositivos móveis

Relatório Técnico de Desenvolvimento de Software apresentado ao curso Ciência da computação
do Centro de Informática, da Universidade Federal da Paraíba,
como requisito para a obtenção do grau de Bacharel em titulo

Orientador: Raoni Kulesza

Maio de 2025

Catalogação na publicação
Seção de Catalogação e Classificação

F475p Figuerêdo, Matheus César Silva.
Procura.Aí: aplicação para auxiliar na recuperação de dispositivos móveis / Matheus César Silva Figuerêdo. - João Pessoa, 2025.
55 f. : il.

Orientação: Raoni Kulesza.
TCC (Graduação) - UFPB/CI.

1. Roubos de dispositivos móveis. 2. Monitoramento. 3. Recuração de dispositivos móveis. 4. Arquitetura de software. 5. Aplicação web. 6. Desenvolvimento de software. I. Kulesza, Raoni. II. Título.

UFPB/CI CDU 004.4

Elaborado por MICHELLE DE KASSIA FONSECA BARBOSA - CRB-738

CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA

Trabalho de Conclusão de Curso de Ciência da computaçãointitulado ***Procura.Aí: Aplicação para auxiliar na recuperação de dispositivos móveis*** de autoria de Matheus César Silva Figuerêdo, aprovada pela banca examinadora constituída pelos seguintes professores:

Prof. Dr. Raoni Kulesza
Instituicao do Professor A

Prof. Dr. Nome do Professor B
Instituicao do Professor B

Prof. Dr. Nome do Professor C
Instituicao do Professor C

Coordenador(a) do Departamento Departamento de informática
Nome do Coordenador
CI/UFPB

João Pessoa, 23 de maio de 2025

Centro de Informática, Universidade Federal da Paraíba
Rua dos Escoteiros, Mangabeira VII, João Pessoa, Paraíba, Brasil CEP: 58058-600
Fone: +55 (83) 3216 7093 / Fax: +55 (83) 3216 7117

RESUMO

O presente trabalho apresenta o desenvolvimento de uma aplicação web voltada à prevenção e monitoramento de roubos e furtos do estado da Paraíba, em resposta ao crescente aumento desse tipo de crime no estado. A aplicação permite que usuários registrem seus dispositivos, emitam alertas de furto, visualizem ocorrências em mapas interativos e mantenham contato com redes de confiança. Além disso, uma interface administrativa possibilita a gestão de dados de usuários e dispositivos, promovendo uma integração eficiente entre as informações de segurança. Foi utilizado ferramentas e tecnologias modernas para frontend como ReactJs, NextJs, Tailwind e Leaflet, bem como ferramentas de gerenciamento backend e armazenamento de dados com AppWrite, além de realizar testes de features da aplicação e documentação da arquitetura de software.

Palavras-chave: Roubos de dispositivos móveis, Monitoramento, Recuração de dispositivos móveis, Arquitetura de Software, Aplicação Web, Desenvolvimento de Software.

ABSTRACT

This work presents the development of a web application aimed at the prevention and monitoring of theft and robbery in the state of Paraíba, in response to the growing increase in this type of crime in the region. The application allows users to register their devices, issue theft alerts, view occurrences on interactive maps, and maintain contact with trusted networks. In addition, an administrative interface enables the management of user and device data, promoting efficient integration of security information. Modern tools and technologies were used for the frontend, such as React, Next.js, Tailwind, and Leaflet, as well as backend management and data storage solutions using AppWrite. The application features were tested, and the software architecture was thoroughly documented.

Key-words: Mobile device thefts, Monitoring, Mobile device recovery, Software Architecture, Web Application, Software Development.

Sumário

1 INTRODUÇÃO

A crescente incidência de roubos e furtos de celulares no Brasil, na última década e meia, junto às subtrações de dinheiro devido a manipulação de aplicativos bancários que neles estão instalados, representa um desafio significativo à segurança pública e à proteção de dados pessoais, visto que, transformou-se em um problema social de grande relevância, em consequência dos seus impactos negativos no que se refere a custos financeiros (quando trata-se da reposição dos celulares, bem como, os prejuízos que resultam da manipulação dos aplicativos bancários) e psicossociais que se associam a perturbações da mobilidade, comunicação e coordenação das rotinas diárias [15].

O uso de ferramentas eletrônicas e o compartilhamento de dados dentre os vários sistemas utilizados, podem ser importantes alternativas, auxiliando de forma relevante os órgãos responsáveis pela segurança pública, proporcionando a estes, melhores condições de fazer frente à evolução das táticas e do modus operandi dos delinquentes. Uma boa maneira de aproveitar o uso dos recursos tecnológicos é através da integração dos sistemas informatizados, visto que, estes aperfeiçoam o uso dos recursos disponíveis nas ações preventivas e repressivas, essenciais no combate à criminalidade. Esse tipo de integração atua como uma ferramenta de prevenção primária, sendo empregue na atividade fim por ser um tanto ostensivo e inibindo assim a ação delituosa, bem como, serve também para conduzir o policiamento investigativo através do mapeamento de áreas de aglomeração e prática de ilícitos, facilitando a identificação de criminosos, contribuindo para a captura dos autores, melhorando a eficiência e também a eficácia das ações de repressão imediata [6].

Dado o crescente aumento nos casos de roubos e furtos de aparelhos celulares no Brasil, é evidente que existe uma certa urgência em desenvolver soluções eficazes como uma forma de frear tais problemas que afetam todos os brasileiros. No ano de 2022, mais de um milhão de casos foram registrados, o que equivale a um aparelho a cada dois minutos que foram subtraídos, segundo o Anuário Brasileiro de Segurança Pública de 2023 [8]. Estados como a Paraíba, que apresentou um crescimento alarmante de 157,7% nos casos de furto, evidenciam que o problema atinge diversas regiões com intensidades variadas, demandando respostas articuladas e adaptadas à realidade local. A complexidade da situação vai além da simples subtração do bem material. Os smartphones, além de seu alto valor comercial, armazenam informações sensíveis como dados bancários, documentos pessoais e credenciais de acesso a redes sociais, o que amplia o impacto do crime e potencializa outros delitos, como fraudes e invasões de contas. Soma-se a isso a atuação de redes criminosas especializadas em receptação e comercialização ilegal dos aparelhos, o que dificulta a recuperação dos dispositivos e fortalece o ciclo do crime. Diante desse cenário, o projeto propõe uma abordagem integrada e multidisciplinar para enfrentar o

problema. A alternativa é fortalecer a cooperação entre os orgãos governamentais e a população com o desenvolvimento de um sistema responsável por notificar alertas sobre os roubos e furtos com atuação imediata após o reporte da ocorrência. Outra iniciativa envolve ações educativas e de conscientização, que buscam orientar a população sobre como proteger seus dados e a importância de registrar os crimes, fornecendo informações que ajudem na identificação de padrões e na atuação preventiva.

Por fim, o projeto também visa auxiliar e agilizar o processo de registro de ocorrências. e tornando as investigações mais eficientes e políticas públicas bem direcionadas, é possível reduzir significativamente os índices de criminalidade associados a esse tipo de crime. A proposta não se limita a medidas reativas, mas aposta em estratégias proativas e colaborativas para promover maior segurança digital e urbana à população.

1.1 Tema

O uso de tecnologias digitais voltadas à segurança pública tem se mostrado promissor para enfrentar problemas estruturais como a subnotificação de crimes, a descentralização de dados e a dificuldade de comunicação entre população e instituições estatais. Então a solução proposta nesse trabalho visa criar uma solução web que permita o registro fácil e seguro de ocorrências por parte da população, além de possibilitarem o cruzamento automatizado de dados com sistemas de segurança, favorecem uma resposta mais rápida e direcionada, além de fortalecerem a gestão estratégica e a transparência dos serviços públicos.

1.2 Problema

A falta de comunicação sobre crimes patrimoniais, como roubos e furtos de celulares, compromete a eficácia das políticas de segurança pública, dificultando a elaboração de estratégias preventivas e reativas adequadas. Fatores como a burocracia no registro de ocorrências, a falta de integração entre sistemas de informação e a ausência de canais acessíveis para a população contribuem para esse cenário. Nessa perspectiva, o desenvolvimento da plataforma “Procura.aí” representa uma resposta prática e inovadora a essas demandas. Ao oferecer uma interface intuitiva para o cidadão e recursos administrativos como dashboards e relatórios analíticos, a aplicação pode auxiliar tanto na recuperação de aparelhos quanto na formulação de políticas públicas mais eficazes. A integração com sistemas policiais e bases de dados institucionais ainda amplia o alcance da solução, favorecendo investigações e a integração entre as instituições, o que vem a ser um fator decisivo na resolução de casos desse gênero.

1.3 Objetivo geral

O trabalho tem por objetivo analisar e documentar uma aplicação web acessível via computadores ou dispositivos móveis, com a finalidade de auxiliar no cadastro e recuperação de celulares roubados, furtados ou perdidos, por meio de uma interface de usuário intuitiva, segura e responsiva. A implementação do sistema visa também fornecer um ambiente com ferramentas de monitoramento eficiente e geração de relatórios detalhados, de modo a auxiliar na gestão do serviço.

1.4 Objetivos específicos

- Identificar e descrever os requisitos funcionais e não funcionais da aplicação.
- Apresentar uma ferramenta de documentação da arquitetura geral do sistema.
- Realizar avaliação das funcionalidades principais da aplicação por meios da análise dos casos de testes.

1.5 Estrutura do relatório técnico

O presente trabalho está organizado em cinco seções. De modo que, na primeira seção é apresentada uma introdução ao tema, apontando o problema, além dos objetivos gerais e específicos do projeto. Na segunda seção, são abordados conceitos fundamentais, que configuram o embasamento teórico do trabalho. A terceira seção, se dá com base no levantamento de requisitos da aplicação Procura.aí, bem como na documentação de arquitetura de software. Na quarta seção, é apresentada a ferramenta utilizada para gerar os diagramas de arquitetura do sistema e os casos de teste. Por fim, na quinta seção, são apresentados as considerações finais sobre o trabalho.

2 CONCEITOS GERAIS

Nesta sessão, são descritos as principais tecnologias utilizadas no desenvolvimento de aplicações web modernas desempenham um papel fundamental na criação de sistemas eficientes, responsivos e escaláveis. Este referencial teórico aborda conceitos gerais relacionados a frameworks, bibliotecas e padrões de arquitetura que moldam a experiência do usuário e a performance das aplicações. A compreensão desses elementos é essencial para embasar decisões técnicas e justificar escolhas no desenvolvimento de soluções web atuais.

2.1 Appwrite

O *Appwrite* é uma plataforma de código aberto que disponibiliza serviços essenciais para o desenvolvimento de projetos variados como: aplicações móveis e web; plataformas SaaS que exigem gerenciamento de usuários e permissões; aplicativos que necessitam de armazenamento seguro e escalável. Possui uma interface intuitiva e uma instalação simplificada, além de dar suporte a múltiplos ambientes de desenvolvimento e linguagens de programações incluindo Javascript, Dart, Swift e Kotlin. Com uma arquitetura modular, permite que possa ser escalado de acordo com a necessidade e customização do projeto, possuindo como requisitos do sistemas apenas a instalação de Docker e Docker Compose para realizar a conteinerização do serviço e uso do Redis para gerencimaneto de filas e cache.

2.2 Backend

O *backend* ou servidor, é a parte de um sistema que tem como principal responsabilidade lidar com as lógicas e regras de negócios gerais, de forma não diretamente visível pelos usuários finais. Normalmente tem um maior poder de processamento se comparado aos clientes, para realizar gerenciamentos e transformações de dados que são trocados entre os clientes e o servidor de forma padronizada, muitas vezes usando o formato JSON.

2.3 BaaS

BaaS(Backend as a Service) é um modelo de serviço baseado em nuvem que disponibiliza uma infraestrutura pré-configurada, permitindo aos desenvolvedores que se concentrem no desenvolvimento do frontend sem a necessidade de um grande foco em gerenciamento de servidores, banco de dados, autenticação, atualizações remotas e as demais ferramentas essenciais para que um sistema completo funcione de forma escalável. Com essas funcionalidades disponíveis, é possível fazer com que uma equipe de desenvolvimento foque na lógica de negócio e na experiência do usuário, sem perder tempo configurando

servidores, mesmo garantindo um alto desempenho e uma alta escalabilidade, além de oferecer uma maior produtividade, reduzir custos de manutenção e minimizar complexidade técnica [14].

2.4 Frontend

O *frontend* é a parte visual de uma aplicação ou sistema que permite a interação direta do usuário, ou seja, a interação com menus, telas, animações, elementos gráficos etc. Nesta etapa são utilizadas principalmente tecnologias como HTML(HyperText Markup Language), CSS(Cascading Style Sheets) e Javascript que juntas são utilizadas para aplicar conceitos como responsividade, acessibilidade e interatividade, que juntas são responsáveis por fazer a adaptação de layouts para diferentes tamanhos de tela, garantem que pessoas com diferentes necessidades possam utilizar a aplicação e tornam a navegação fluída e dinâmica por meio de eventos como cliques e preenchimento de formulários.

2.5 ReactJs

ReactJs é uma biblioteca JavaScript criada pelo Facebook para construir interfaces de usuário (UI), especialmente componentes reutilizáveis que permitem dividir a interface em partes independentes e isoladas. Cada componente representa um pedaço da UI (User Interface), como um botão, um cabeçalho ou um formulário inteiro, que podem ser reutilizados em contextos diferentes, sendo definidas propriedades distintas para cada situação.

Ele facilita a criação de interfaces dinâmicas com atualização eficiente e rápida dos dados na tela por meio de um conceito chamado DOM Virtual, que é uma estrutura que representa a interface visual de uma página web. Já que alterar o DOM diretamente pode ser lento e custoso, o ReactJs usa a DOM virtual sendo uma cópia mais leve que a original, dessa forma quando um componente muda, é criada uma nova versão da DOM e é feita uma comparação para detectar mudanças em relação a versão anterior, fazendo então a atualização somente dos trechos que sofreram alterações. Essa tratativa torna mais eficiente a manipulação do DOM, principalmente em aplicações com uma maior escala, além de otimizar o desenvolvimento, já que a UI é atualizada em tempo real.

Outras ferramentas presentes no ReactJs são os Hooks que permitem utilizar estado, efeitos colaterais, contexto e outras funcionalidades em componentes funcionais, como por exemplo: o useState para gerenciar estados locais; o useEffect para lidar com efeitos colaterais como requisições; useContext para consumir e lidar com contextos. Além disso, o framework dá suporte a renderizações condicionais que é uma prática muito usada para exibir diferentes componentes ou elementos visuais com base em condições lógicas e estados da aplicação.

2.6 NextJs

O *NextJs* é um framework para React.Js que se destaca por oferecer suporte nativo ao Server-Side Rendering (SSR), uma abordagem em que as páginas são renderizadas no servidor a cada requisição, como mostrado na Figura 1. Isso permite que o conteúdo chegue ao navegador já processado, favorecendo significativamente o SEO(Search Engine Optimization), já que os motores de busca conseguem indexar as páginas com facilidade. O SSR também contribui para uma melhor performance inicial, especialmente em conexões lentas, pois o usuário recebe rapidamente o HTML renderizado. Outro benefício é a atualização dinâmica de dados, o que torna NextJs ideal para páginas com conteúdo que muda com frequência. Com SSR, não é necessário aguardar o carregamento do JavaScript para exibir o conteúdo, o que melhora a experiência do usuário. Além disso, o NextJs disponibiliza uma camada intermediária para geração de rotas automáticamente, baseado na estrutura de pastas e arquivos que são criadas dentro do projeto. [5]

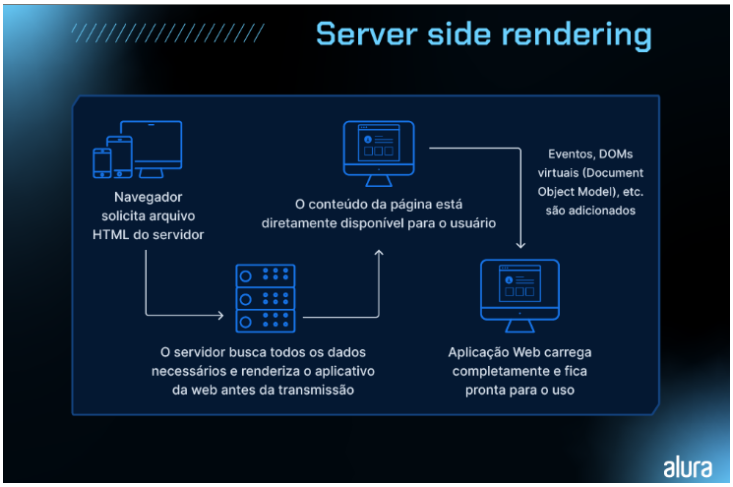

Figura 1: Fluxo de SSR.
Fonte: Disponível em https://www.alura.com.br/artigos/next-js

2.7 Single Page Application

Uma *SPA (Single Page Application)* é um tipo de aplicação web em que a navegação entre páginas ocorre sem recarregar toda a página no navegador. Em vez disso, apenas o conteúdo necessário é atualizado dinamicamente usando JavaScript, proporcionando uma experiência mais fluida e rápida para o usuário. Tecnologias como ReactJs, Vue.js e Angular são comumente utilizadas para desenvolver SPAs, pois permitem a criação de interfaces reativas que se atualizam conforme o estado da aplicação muda[13]. O fluxo tradicional dos sites e aplicações consistia em uma renderização página por página, onde muitas vezes o servidor retornava como resposta de uma requisição o conteúdo html, css e javascript, e com a implementação de SPAs o servidor passa a retornar apenas os dados

que serão exibidos em tela, normalmente em formato JSON, para que o lado do cliente fique com a responsabilidade de renderizar em tela os novos dados requisitados utilizando javascript, como mostrado na Figura 2.

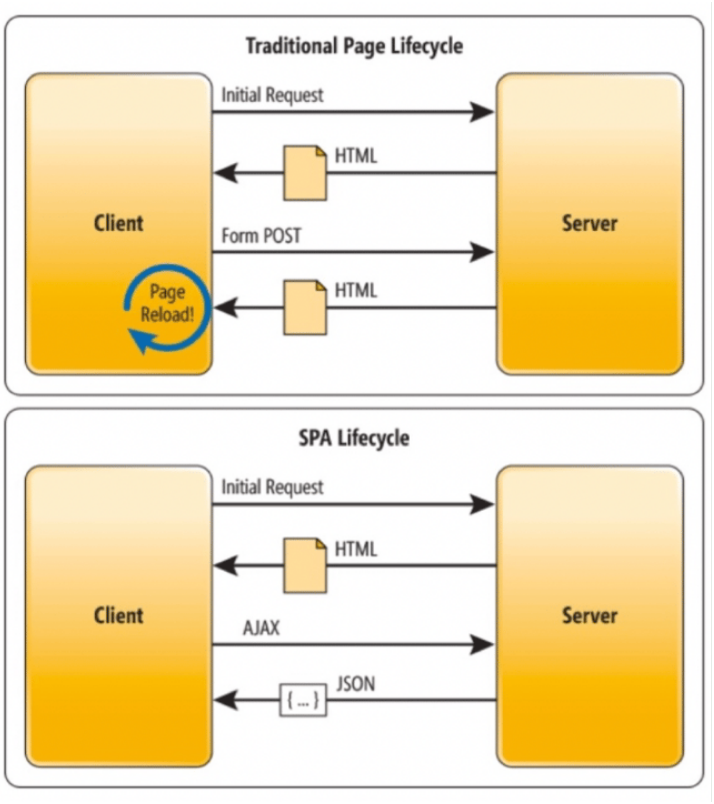

Figura 2: Fluxo de Single Page Application.
Fonte: Disponível em https://experienceleague.adobe.com/pt-br/docs/experience-platform/web-sdk/personalization/adobe-target/spa-implementation

2.8 Tailwind

Tailwind CSS é um framework que faz a utilização de CSS e permite a criação de interfaces modernas e responsivas de forma altamente customizável e eficiente. Ao contrário de frameworks tradicionais que oferecem componentes prontos, o Tailwind fornece classes úteis e pré estilizadas de baixo nível que são diretamente aplicáveis no HTML, o que torna o desenvolvimento mais ágil e flexível. Ele promove a prática de "utility-first", que incentiva a escrita de estilos diretamente nos elementos HTML ao invés de arquivos CSS separados, facilitando a manutenção e reduzindo a complexidade do código [19]. Além disso, no Tailwind é possível ter um sistema de design altamente configurável via o arquivo tailwind.config.js, o que possibilita alinhar o framework à identidade visual do projeto com facilidade. Sua popularidade cresceu rapidamente na comunidade front-end devido à sua integração com frameworks modernos, bem como seu suporte para temas escuros e variantes condicionais. O uso de ferramentas como JIT (Just-in-Time compiler) também otimizou a performance do framework, gerando CSS apenas quando necessário.

Esses fatos demonstram que Tailwind pode melhorar a produtividade dos desenvolvedores e a consistência visual das aplicações web modernas [11].

2.9 Leaflet

Leaflet é uma biblioteca JavaScript de código, amplamente utilizada para exibição de mapas interativos em aplicações web. Com uma API simples e leve, Leaflet permite a renderização de mapas utilizando dados de diferentes fontes, com suporte a camadas, marcadores, popups e eventos de interação. Por ser altamente extensível, a biblioteca conta com uma vasta gama de plugins que adicionam funcionalidades como a geolocalização. Leaflet é frequentemente escolhido em projetos por sua leveza em comparação a outras soluções como o Google Maps API, além de oferecer liberdade quanto ao uso de indicadores personalizados. Seu uso é comum em sistemas de monitoramento, aplicações de logística, turismo e planejamento urbano. Em tempos recentes, o Leaflet vem sendo utilizadoem conjunto com frameworks modernos como ReactJs e VueJs, através de bibliotecas como React-Leaflet e Vue2Leaflet, possibilitando a integração fluida em SPAs (Single Page Applications). Os vários casos de suporte da comunidade e atualizações asseguram sua relevância em projetos geoespaciais modernos [4].

2.10 Funções Serverless

A *computação serverless ou Functions as a Service (FaaS)*, permite que desenvolvedores executem trechos de código para responder eventos sem a necessidade de gerenciar servidores. Essa abordagem abstrai questões de infraestrutura, o que possibilita que as equipes se concentrem exclusivamente no desenvolvimento da das regras de negócios, sem manter o foco nos detalhes de como os eventos estão estruturados. Plataformas como AWS Lambda, Google Cloud Functions e Azure Functions oferecem suporte a esse modelo.

Há uma crescente demanda de serviços FaaS, com vários estudos recentes que têm analisado a evolução e os desafios da tecnologia citada [18]. Por exemplo, a pesquisa "Rise of the Planet of Serveless Computing: A Systematic Review"fornece uma revisão abrangente sobre a utilização e pesquisa em computação serveless, incluindo otimização de desempenho, frameworks de programação e migração de aplicações [20].

Além disso, questões de segurança na computação serverless têm sido objeto de investigação. O artigo "Serverless computing: a security perspective"discute as ameaças e desafios únicos introduzidos por essa arquitetura, destacando a necessidade de novas abordagens para garantir a segurança das aplicações serverless.

2.11 APIs e SDKs

APIs (Interfaces de Programação de Aplicações) são conjuntos de definições e tratativas que permitem realizar a integração entre diferentes sistemas e aplicações, tornando a comunicação facilitada e o compartilhamento de funcionalidades. Já os SDKs (Kits de Desenvolvimento de Software) são conjuntos de ferramentas, bibliotecas e documentação que auxiliam os desenvolvedores na criação de aplicações para plataformas específicas, fornecendo os requisitos necessários para a implementação de funcionalidades que interagem com APIs ou serviços subjacentes.

A combinação de APIs e SDKs é fundamental no desenvolvimento moderno de software, pois permite a criação de aplicações robustas e integradas. Enquanto as APIs oferecem os meios para acessar funcionalidades e dados, os SDKs fornecem as ferramentas que melhor implementam essas interações de forma eficiente e padronizada. Essas tecnologias estão sendo exploradas cada vez mais para o desenvolvimento e a utilização em diversos contextos, incluindo computação em nuvem, dispositivos móveis e Internet das Coisas (IoT), visando aprimorar a combinação de diferentes funcionalidades em um mesmo serviço e a eficiência no desenvolvimento de software.

3 METODOLOGIA

3.1 Visão Geral

A proposta de desenvolver a aplicação Procura.Aí é feita como meio de resposta ao cenário alarmante de criminalidade envolvendo dispositivos móveis. Portanto a ideia central do sistema é permitir que usuários registrem seus dispositivos, cadastrem ocorrências de roubo ou furto para que possam ter acesso simplicado e visualizem essas informações e possam acompanhar as ocorrências. Além disso, a aplicação facilita o envio de alertas, comunicação com contatos de confiança e oferece uma interface administrativa para órgãos responsáveis acompanharem os dados em tempo real por meio de gráficos e mapas, de forma coerente com a localização das ocorrências.

Do ponto de vista técnico, a aplicação foi construída utilizando tecnologias modernas como ReactJs, NextJs, Tailwind, AppWrite, Firebase e Leaflet, priorizando desempenho, usabilidade e escalabilidade. Com uma arquitetura baseada no modelo C4, o sistema é organizado em camadas bem definidas, com ênfase na modularidade e segurança. O objetivo final é não apenas oferecer uma ferramenta funcional, mas também promover a conscientização e o combate integrado ao mercado de venda/troca ilegal e ao ciclo de criminalidade envolvendo dispositivos móveis.

3.2 Tecnologias

- JavaScript
- TypeScript
- ReactJs
- NextJs
- Tailwind
- AppWrite
- Leaflet

3.2.1 JavaScript

JavaScript é a linguagem principal utilizada no projeto para construção de funcionalidades dinâmicas no front-end, permitindo interações diretas com os elementos da interface e manipulação de dados em tempo real.

3.2.2 TypeScript

TypeScript é uma linguagem baseada em JavaScript com tipagem estática, adotada no projeto para aumentar a robustez do código, facilitar a detecção de erros em tempo de desenvolvimento e melhorar a manutenção da aplicação.

3.2.3 ReactJs

ReactJs foi utilizado para a criação de componentes reutilizáveis na interface, possibilitando a construção de uma aplicação interativa e performática com atualização eficiente dos elementos da tela.

3.2.4 NextJs

NextJs é o framework escolhido para estruturar a aplicação, permitindo renderização do lado do servidor (SSR) e geração de páginas estáticas (SSG), melhorando o desempenho e o SEO da plataforma.

3.2.5 Tailwind

A estilização foi feita com *Tailwind*, uma biblioteca CSS utilitária que permite aplicar estilos diretamente nas classes HTML. Essa abordagem facilita a criação de layouts responsivos, com grande flexibilidade para personalização visual. Com o uso de utilitários pré-definidos, o desenvolvedor pode evitar arquivos CSS separados, reduzindo o tempo de desenvolvimento e melhorando a produtividade. Além disso, a integração com o ReactJs e NextJs é bastante fluida, permitindo a composição de interfaces modernas e bem estruturadas com menor esforço. A escolha do Tailwind também leva em consideração sua extensa documentação e comunidade ativa, que facilitam a manutenção e evolução da interface ao longo do tempo.

3.2.6 AppWrite

O *AppWrite* foi utilizado como uma plataforma backend como serviço (BaaS), centralizando diversas funcionalidades importantes como autenticação, banco de dados, permissões e gerenciamento de arquivos. Ele fornece uma API REST segura e personalizável, facilitando o desenvolvimento sem a necessidade de configurar um servidor backend completo do zero. No projeto, o AppWrite é responsável por armazenar dados de usuários, registros de celulares roubados, bem como garantir que apenas usuários autorizados tenham acesso a determinadas operações. Sua integração com o front-end permite rapidez

no desenvolvimento e escalabilidade da aplicação com menos complexidade.

3.2.7 Leaflet

O *Leaflet* foi empregado como biblioteca de mapeamento para representar visualmente a geolocalização dos roubos e furtos de celulares. Ele permite exibir mapas interativos leves, personalizáveis e compatíveis com diversos dispositivos. Com o Leaflet, é possível plotar marcadores nos locais das ocorrências, traçar padrões geográficos e analisar áreas de maior incidência criminal. Essa visualização espacial ajuda tanto usuários comuns quanto gestores públicos a entenderem melhor a distribuição dos crimes. A biblioteca também suporta camadas, popups e integração com serviços externos de mapas, como OpenStreetMap, tornando-se ideal para aplicações de monitoramento urbano.

3.3 Usuários

A aplicação apresentada foi desenvolvida para pessoas que possuem dispositivos com acesso a internet e navegador para acesso ao sistema, com faixa idade acima de 18 anos.

3.4 Requisitos funcionais

3.4.1 [RF01] Cadastrar usuário

O sistema deve permitir que usuários façam seu cadastro, utilizando nome, cpf, email e senha para que tenham acesso as funcionalidades da aplicação.

3.4.2 [RF02] Realizar autenticação

O sistema deve permitir login de usuários e verificar permissões distintas para usuários comuns e administradores.

3.4.3 [RF03] Cadastrar dispositivos

O sistema deve permitir que usuários registrem seus dispositivos informando marca, modelo do dispositivo, número de celular e IMEI(Identificador do aparelho).

3.4.4 [RF04] Editar dispositivos cadastrados

O sistema deve permitir que usuários editem seus dispositivos informando de forma semelhante a marca, modelo do dispositivo, número de celular e IMEI(Identificador do

aparelho).

3.4.5 [RF05] Visualizar dispositivos

O sistema deve permitir que usuários visualizem uma lista com os dispositivos cadastrados até o momento, além de incluir as opções de excluir, ver detalhes, editar e criar alerta de ocorrência na listagem.

3.4.6 [RF06] Deletar dispositivos

O sistema deve permitir que usuários deletem dispositivos previamente cadastrados, sendo aberto um popup de confirmação da ação.

3.4.7 [RF07] Adicionar alerta de ocorrência

O sistema deve permitir que usuários adicionem alertas de ocorrências onde é informado a data, descrição, tipo de ocorrência e o local no mapa onde ocorreu, permitindo a busca pelo cep.

3.4.8 [RF08] Cadastrar contato de confiança

O sistema deve permitir que usuários registrem seus contatos de confiança, para eventuais contatos de emergência, onde é informado o nome, telefone e email.

3.4.9 [RF09] Editar contato de confiança

O sistema deve permitir que usuários editem seus contatos de confiança, onde é informado o nome, telefone e email.

3.4.10 [RF10] Visualizar contatos de confiança

O sistema deve permitir que usuários visualizem uma lista com os contatos de confiança cadastrados até o momento, além de incluir as opções de excluir e editar na listagem.

3.4.11 [RF11] Deletar contato de confiança

O sistema deve permitir que usuários deletem contatos de confiança, sendo aberto um popup de confirmação da ação.

3.4.12 [RF12] Visualizar alertas de ocorrência

O sistema deve exibir, para usuários com perfil de administrador, uma lista de alertas com informações sobre os dispositivos como marca, modelo, status(perdido, furtado ou roubado) e seus proprietários, sendo filtrados por IMEI e nome do proprietário.

3.4.13 [RF13] Recuperar dispositivo

O sistema deve exibir, para usuários com perfil de administrador, a opção de recuperar dispositivo onde são informadas as informações gerais, orgãos responsáveis, setor, local de retirada, além da opção de anexar documentos.

3.4.14 [RF14] Visualizar Dashboard administrativo

O sistema deve exibir, para usuários com perfil de administrador, uma interface administrativa para visualização de dados consolidados, como alertas, usuários, dispositivos e ocorrências distribuídas por bairros da cidade.

3.4.15 [RF15] Recuperação e redefinição de senhas

Usuários devem poder redefinir sua senha por meio de formulários apropriados.

3.4.16 [RF16] Edição de perfil

Usuários devem poder atualizar suas informações pessoais e senha através da interface de perfil.

3.4.17 [RF17] Listagem e exportação de dados

O sistema deve permitir a exportação de dados de usuários e alertas, em formatos csv e xlsx.

3.5 Requisitos não funcionais

3.5.1 Usabilidade

3.5.1.1 [RNF01] Intuitividade

A aplicação deverá apresentar facilidade em seu uso, sem necessitar de conhecimento prévio do funcionamento do sistema.

3.5.1.2 [RNF02] Prevenção de erros

A aplicação deverá possuir mecanismos de confirmação e recuperação para prevenção de ações acidentais do usuário.

3.5.1.3 [RNF03] Feedback

O sistema deve apresentar toasts e modais de feedback em ações como login, cadastro, erro de validação ou sucesso.

3.5.1.4 [RNF04] Responsividade

A interface deve ser compatível com diferentes tamanhos de tela (*mobile*, *tablet*, *desktop*).

3.5.2 Portabilidade

3.5.2.1 [RNF05] Compatibilidade

A aplicação deve estar disponível para navegadores mais utilizados como *Chrome* versão 66+, *Edge* versão 16+, *Firefox* versão 57+ e *Opera* versão 53+.

3.5.3 Confiabilidade

3.5.3.1 [RNF06] Disponibilidade

É imprescindível a disponibilidade total da aplicação, tornando viável a criação de ocorrências a qualquer momento.

3.5.4 Aspecto Organizacional

3.5.4.1 [RNF07] Versionamento

A aplicação deve ser desenvolvida e gerenciada pela plataforma Gitlab, para o controle de versões, bem como utilizar CI/CD para implementar práticas de DevOps.

3.5.4.2 [RNF08] Ferramentas de implementação

A aplicação deve ser desenvolvida utilizando as tecnologias *ReactJs* versão 18.x.x, *NestJS* versão 15.x.x, *Tailwind* versão 3.x.x, *AppWrite* versão 14.x.x e *Leaflet* versão

1.9.x.

3.5.5 Interoperabilidade

3.5.5.1 [RNF09] Conexão à internet

A aplicação necessita de acesso à internet para o uso correto e eficaz.

3.5.5.2 [RNF10] Acesso ao Firabase e AppWrite

Serviços como Firebase e AppWrite devem ser utilizados para autenticação, armazenamento para garantir maior segurança e escalabilidade.

3.5.6 Privacidade

3.5.6.1 [RNF11] Proteção de dados

A aplicação deve assegurar que os dados dos usuários não sejam expostos a outros usuários sem permissão.

3.5.7 Segurança

3.5.7.1 [RNF12] Validação de dados

Todos os formulários devem validar os dados de entrada antes do envio para a api.

3.5.7.2 [RNF13] Permissionamento

A aplicação deve assegurar que apenas usuários com devida permissão acessem determinadas telas.

3.5.7.3 [RNF14] Validação do Código IMEI

A aplicação deve assegurar a validade do código de identificação do dispositivo IMEI.

3.6 Arquitetura de Software

A aplicação Procura.Aí segue uma arquitetura baseada no modelo c4, que é uma abordagem moderna e estruturada para descrever a arquitetura de software de forma clara

e visual, o que torna possível uma visão mais abrangente sobre o projeto e não apenas técnica. Essa abordagem é chamada de "C4"por causa dessas quatro camadas: Context, Containers, Components e Code. É uma forma eficiente de documentar e tomar decisões arquiteturais de forma progressiva, adaptada a diferentes grupos de pessoas envolvidas e diferentes fases do projeto.

3.6.1 Diagrama de Contexto

O diagrama de contexto é responsável por passar uma visão generalista sobre o sistema, seus usuários (atores) e os sistemas externos com os quais interage. Sua característica principal é possibilitar o entendimento de onde o sistema se encaixa no ambiente maior, como apresentado na Figura 3.

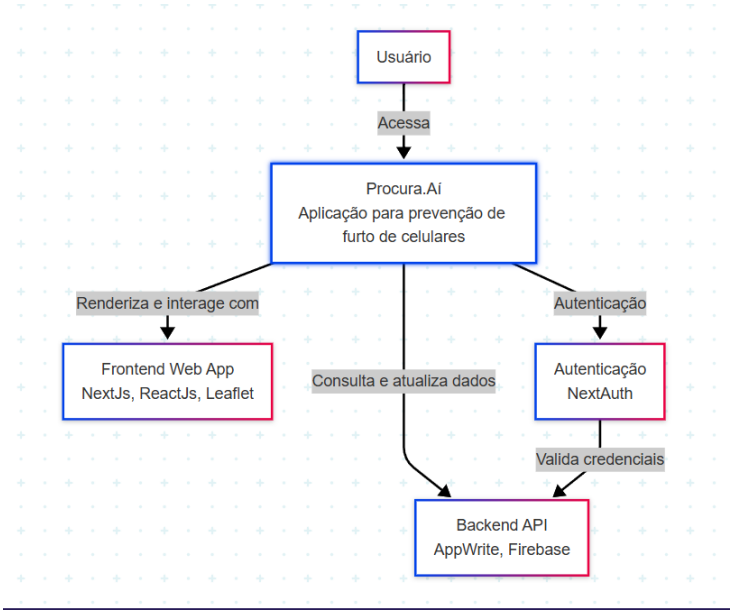

Figura 3: Diagrama de Contexto
Fonte: Elaborado pelo Autor

3.6.2 Diagrama de Container

O diagrama de container detalha as principais partes do sistema, como aplicações web, APIs, bancos de dados e serviços de backend. Cada container representa um processo ou aplicativo executável de forma conjunta ou não, como representado na Figura 4.

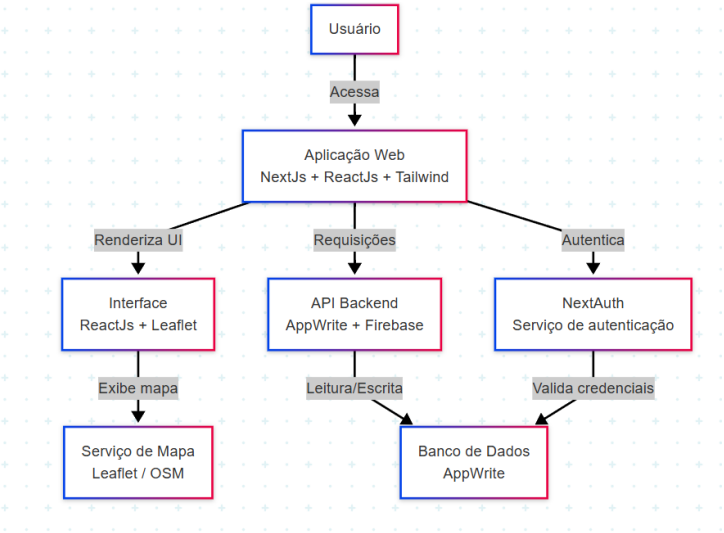

Figura 4: Diagrama de Container
Fonte: Elaborado pelo Autor

3.6.3 Diagrama de Componente

O objetivo do diagrama de componente é focar de forma mais específica dentro de um container, detalhando os componentes internos, como classes, módulos, serviços e bibliotecas que executam funcionalidades específicas, de forma exemplificada na Figura 5.

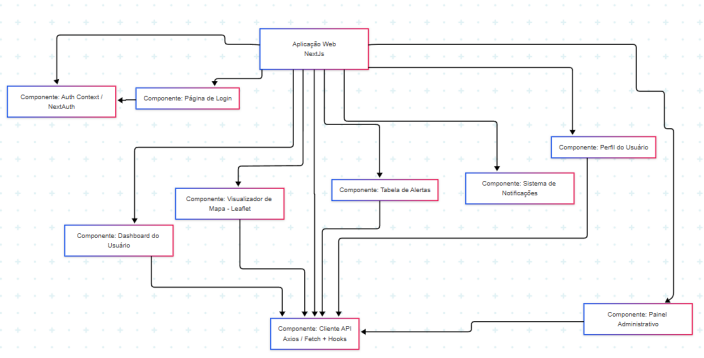

Figura 5: Diagrama de Componentes
Fonte: Elaborado pelo Autor

3.6.4 Diagrama de Código

O diagrama de código, ou diagrama de classe, tem como objetivo mostrar a estrutura de código-fonte em nível de classes ou arquivos, sendo um diagrama em formato UML simplificado, facilitando ao time de desenvolvimento o entendimento sobre as classes do sistema, como apresentado na Figura 6.

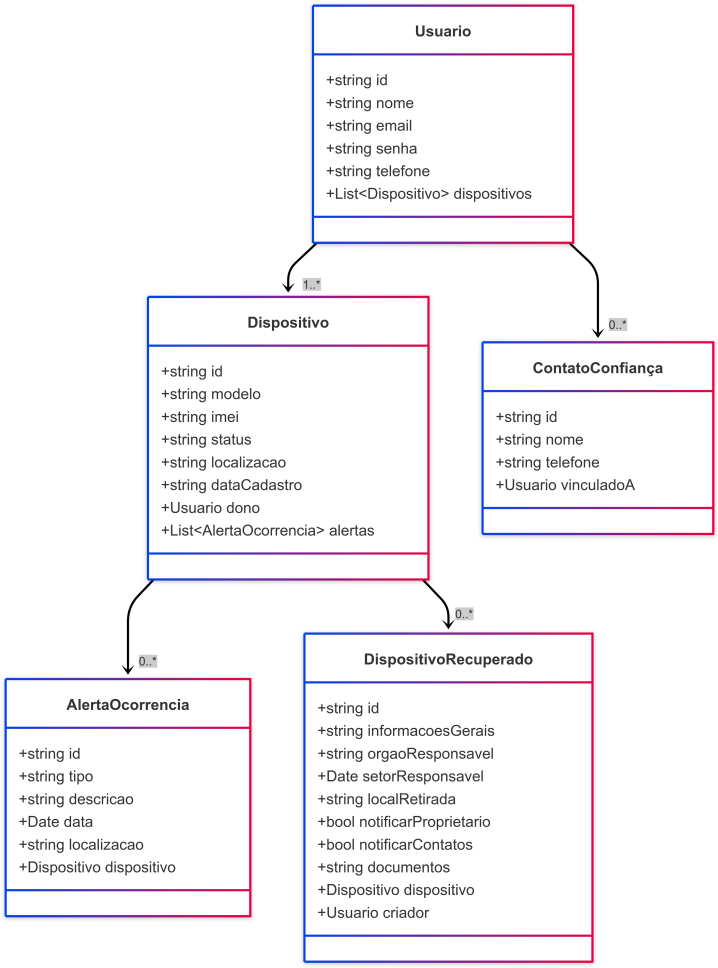

Figura 6: Diagrama de Código
Fonte: Elaborado pelo Autor

3.7 Telas da aplicação

As telas da aplicação se dividem em dois fluxos distintos: o fluxo do administrador, onde é possível consultar e analisar os dados gerais sobre o sistema por meio de dashboards interativos, além de consultar e gerenciar alertas e usuários cadastrados; no fluxo do usuário comum, é possível gerenciar dispositivos pessoais, gerenciar contatos, gerenciar perfis e criar alertas de ocorrências.

3.7.1 Telas sem autenticação

3.7.1.1 Tela inicial

A tela inicial da aplicação Procura.Aí, quando o usuário ainda não está autenticado, é apresentada pela Figura 7.

Figura 7: Tela inicial

3.7.1.2 Tela de login

A Figura 8 apresenta a tela de login, onde são feitas as regras de autenticação e salvamento do token de acesso.

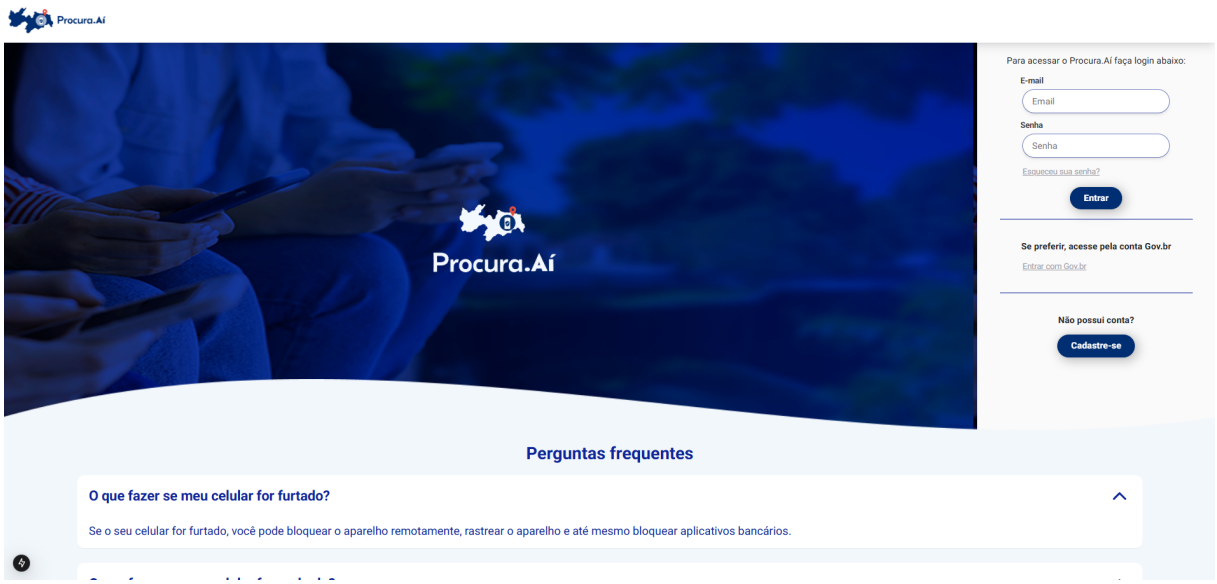

Figura 8: Tela de Login

3.7.1.3 Tela de cadastro de usuário

A Figura 9 apresenta a tela de cadastro de novos usuários, onde é requerido suas principais informações para registro.

Figura 9: Tela de cadastro de usuário

3.7.2 Fluxo do usuário Admin

3.7.2.1 Dashboard

A Figura 10 e a Figura 11 apresentam os dados disponibilizados para consulta do usuário admin sobre o sistema no geral, representado por gráficos e mapas interativos.

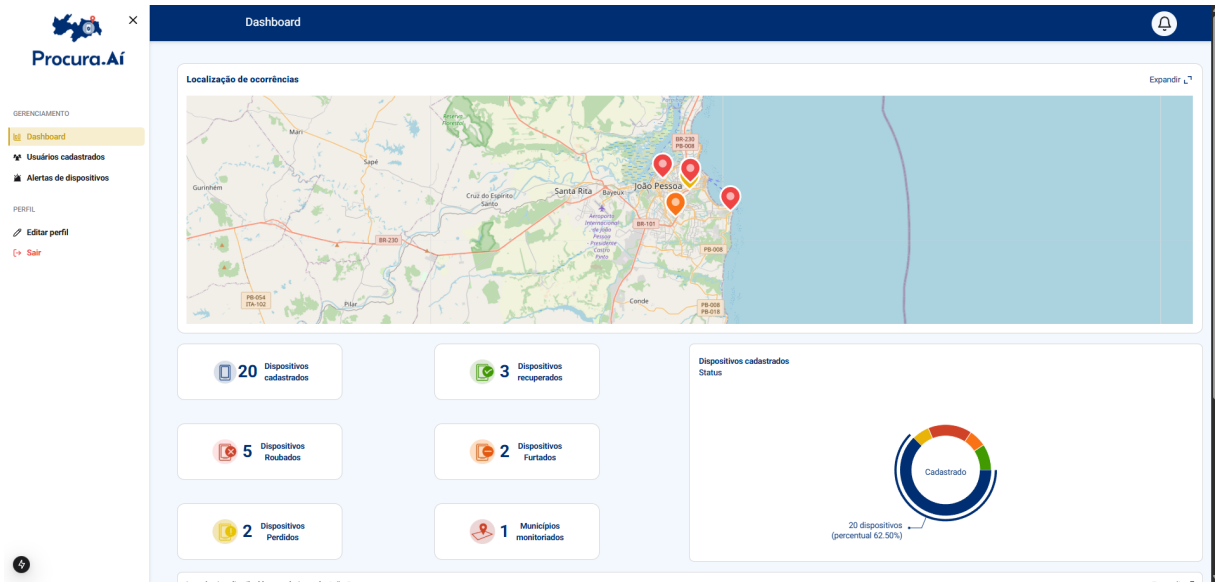

Figura 10: Tela de Dashboard

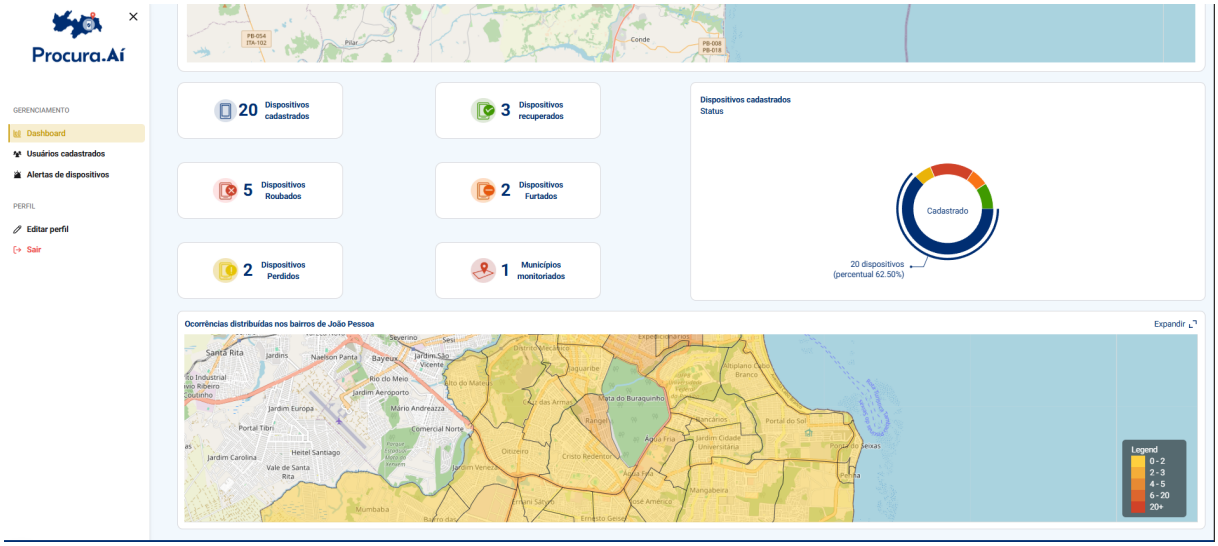

Figura 11: Tela de Dashboard

3.7.2.2 Listagem de usuários

A Figura 12 apresenta a listagem de todos os usuários cadastrados no sistema, com suas principais informações e seus respectivos perfis. Já a Figura 13 apresenta os modais de confirmação das ações que existem nessa listagem, que são: detalhes; desativação do usuários e exclusão.

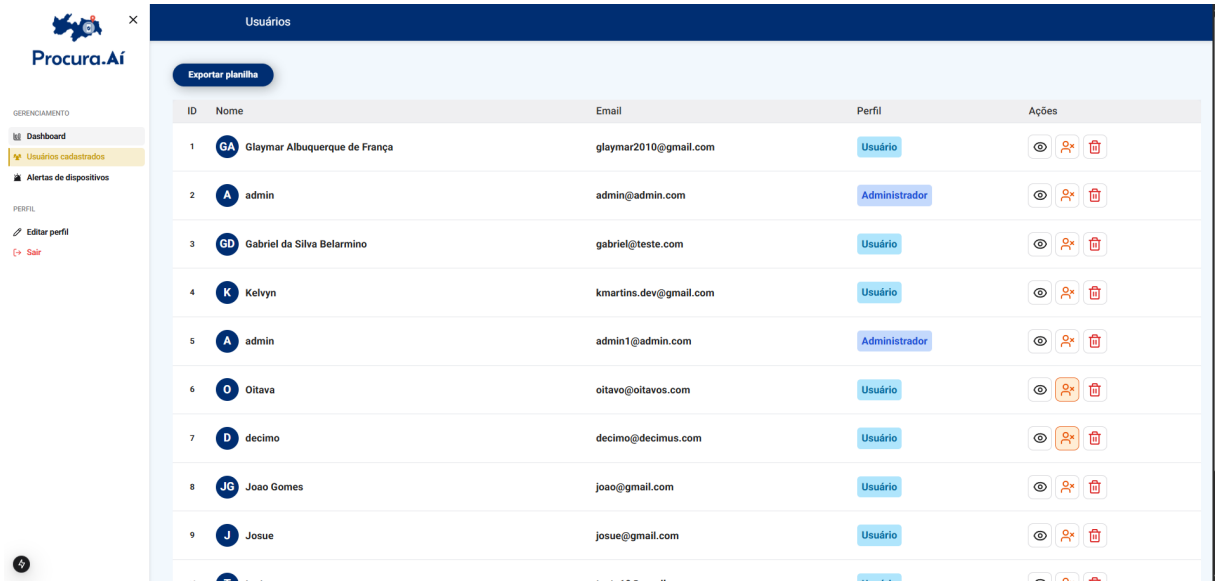

Figura 12: Tela de listagem de usuários

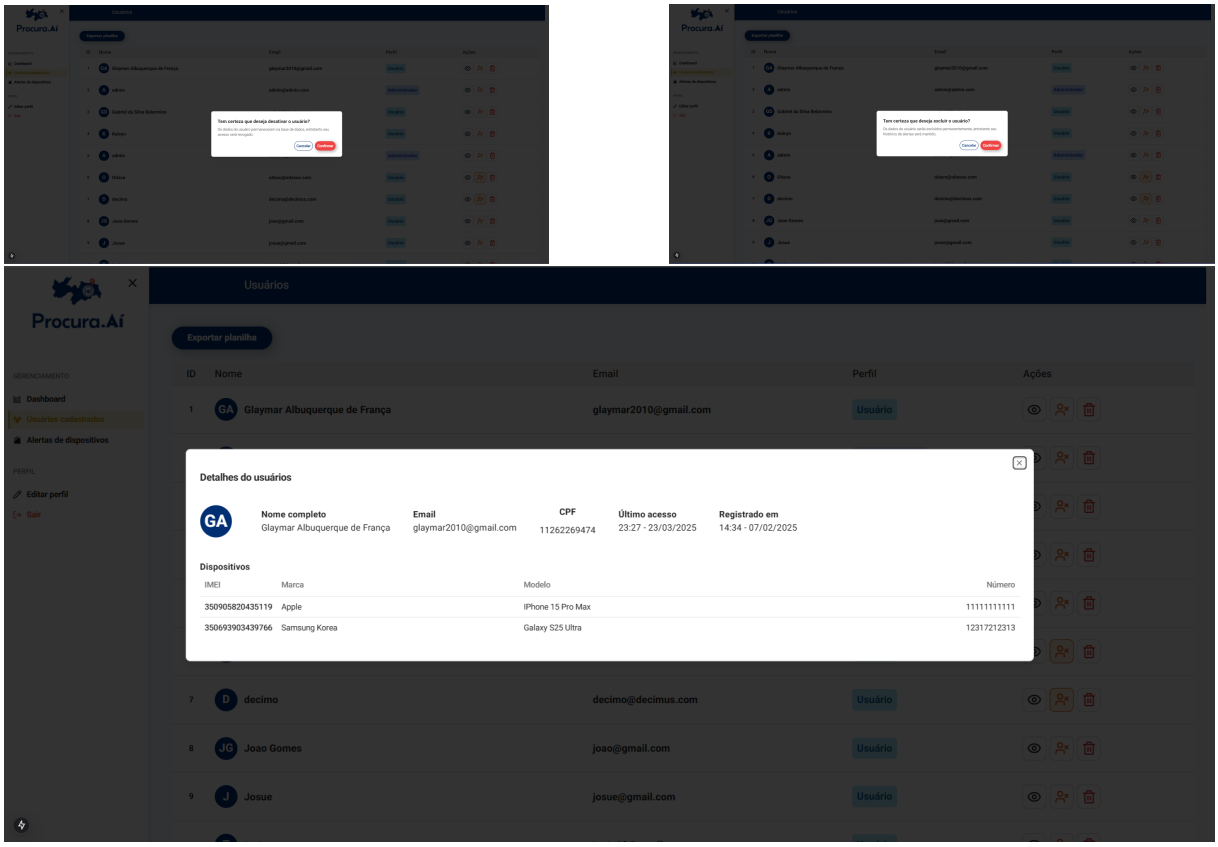

Figura 13: Modais de ações na listagem de usuários

3.7.2.3 Listagem de alertas

A Figura 14 apresenta a listagem dos alertas cadastrados no sistema, onde são listadas as informações sobre os alertas e o status atual de cada alerta. Já a Figura 15 apresenta as informações detalhadas sobre cada alerta.

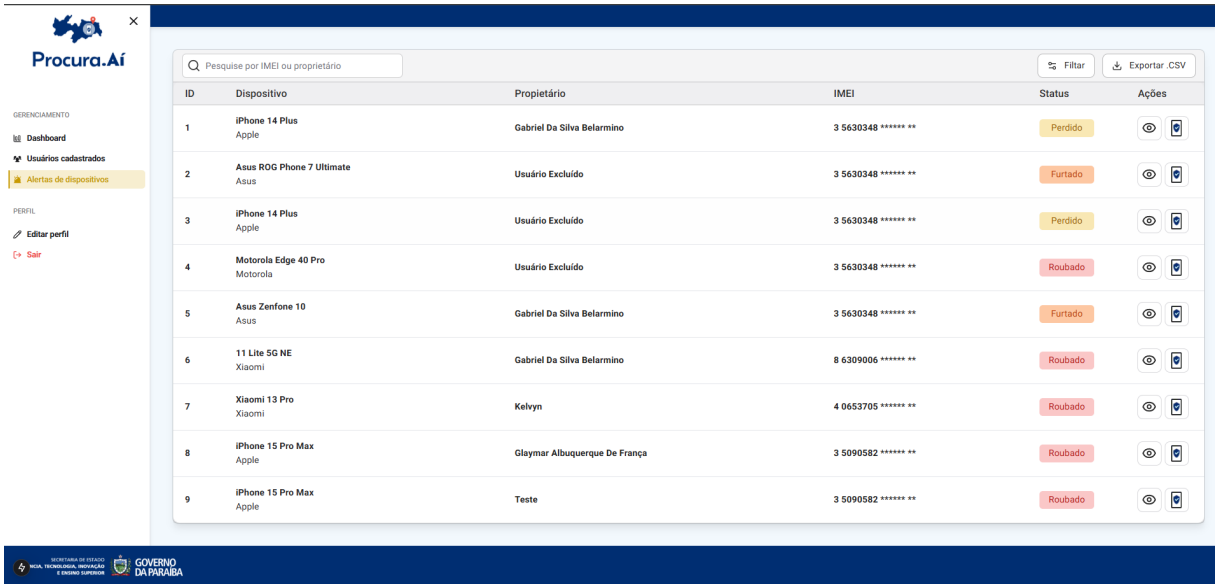

Figura 14: Tela de Listagem de Alertas

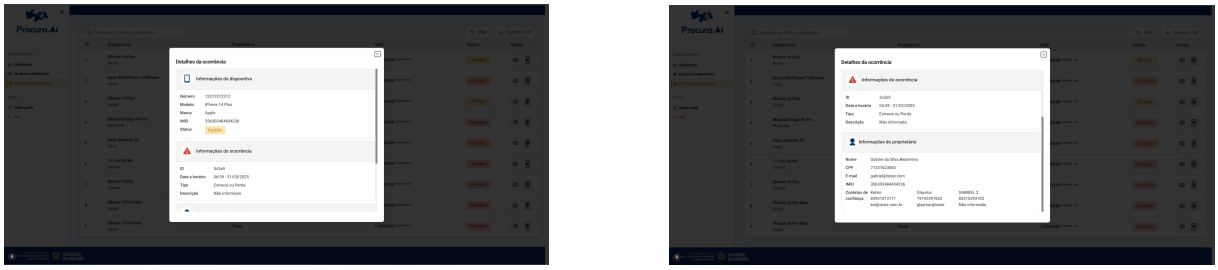

Figura 15: Detalhes do alerta

3.7.2.4 Cadastro de recuperação de dispositivo

A Figura 16 apresenta, em forma de modal(popup), o formulário para cadastro de dispositivos recuperados.

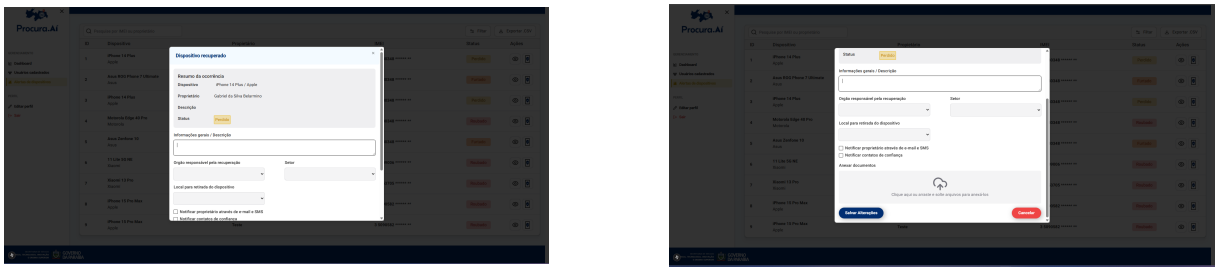

Figura 16: Cadastro de recureção de dispositivo

3.7.2.5 Edição do perfil

A Figura 25 apresenta o formulário para edição dos dados do perfil do usuário logado, que foram cadastrados previamente.

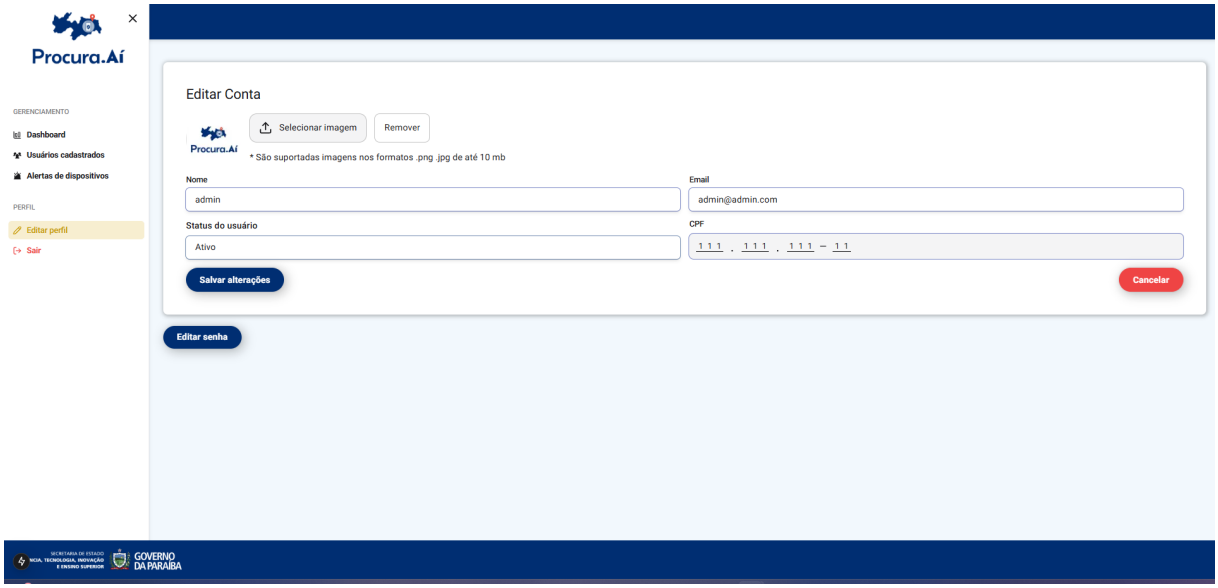

Figura 17: Tela de edição do perfil

3.7.3 Fluxo do usuário comum

3.7.3.1 Meus dispositivos

A Figura 18 apresenta a lista de dispositivos cadastrados por um determinado usuário, que neste caso seria o usuário logado. Além disso, na Figura 19 é apresentado os detalhes do dispositivo e o popup de confirmação de deleção do item.

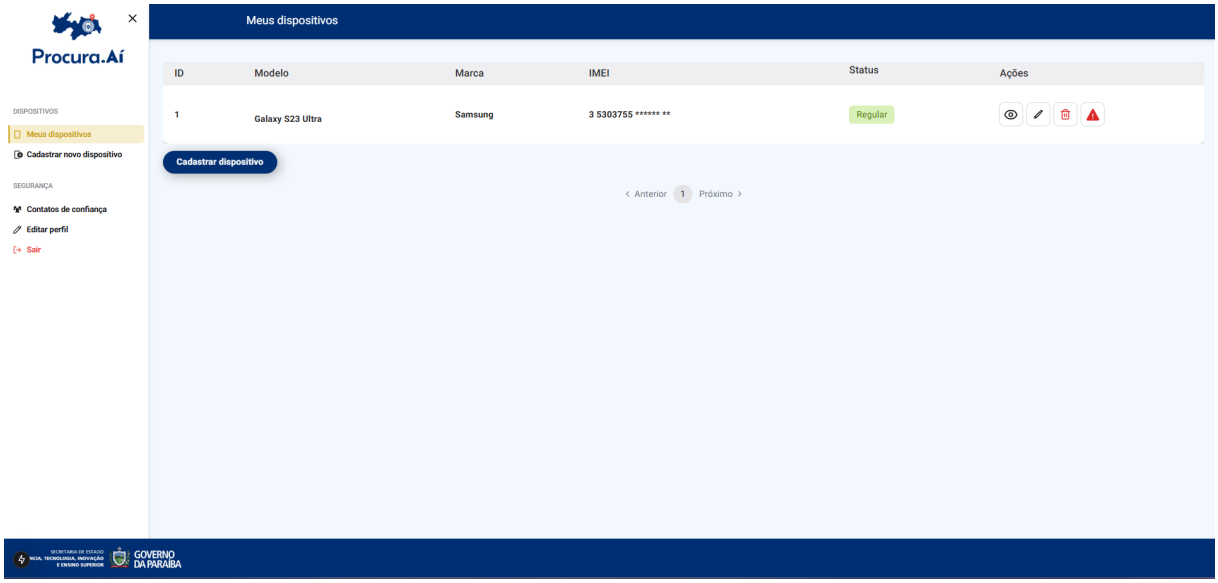

Figura 18: Tela de meus dispositivos

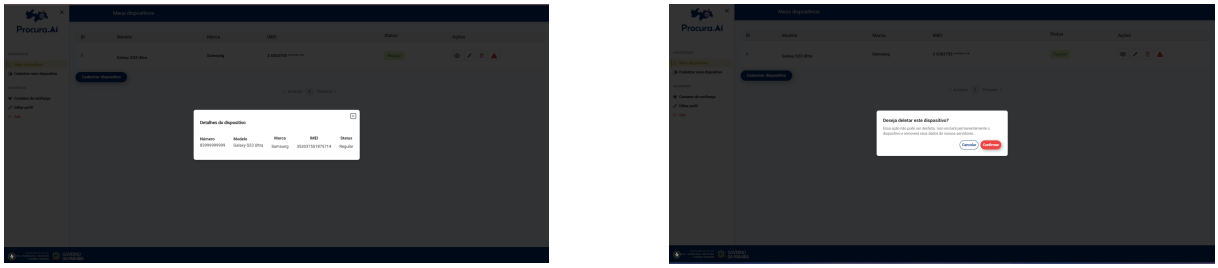

Figura 19: Cadastro de recureção de dispositivo

3.7.3.2 Cadastro de dispositivos

Na Figura 20 é apresentado a tela de cadastro de dispositivos, com as principais informações do mesmo.

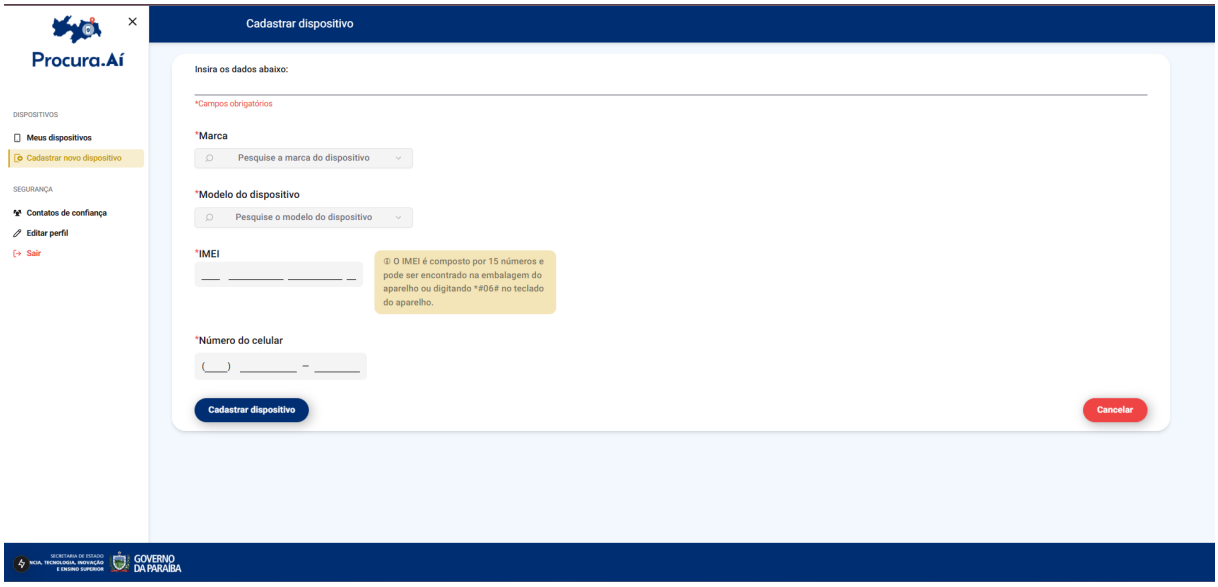

Figura 20: Tela de cadastro de dispositivos

3.7.3.3 Edição de dispositivos

Na Figura 21 é apresentado a tela de edição de dispositivos, de forma semelhante ao cadastro, possui as principais informações do dispositivo previamente preenchidas.

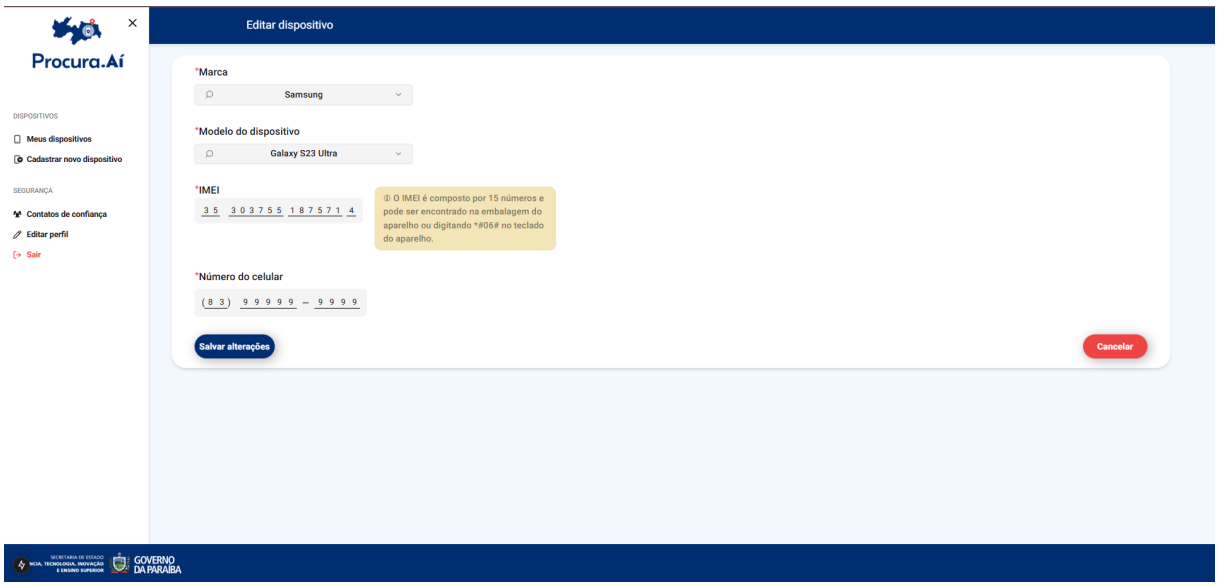

Figura 21: Tela de cadastro de dispositivos

3.7.3.4 Adição de alertas

Na Figura 22 é apresentada a tela de adição de alertas, com as informações necessárias para se criar um alerta, incluindo escolha de localização em mapa.

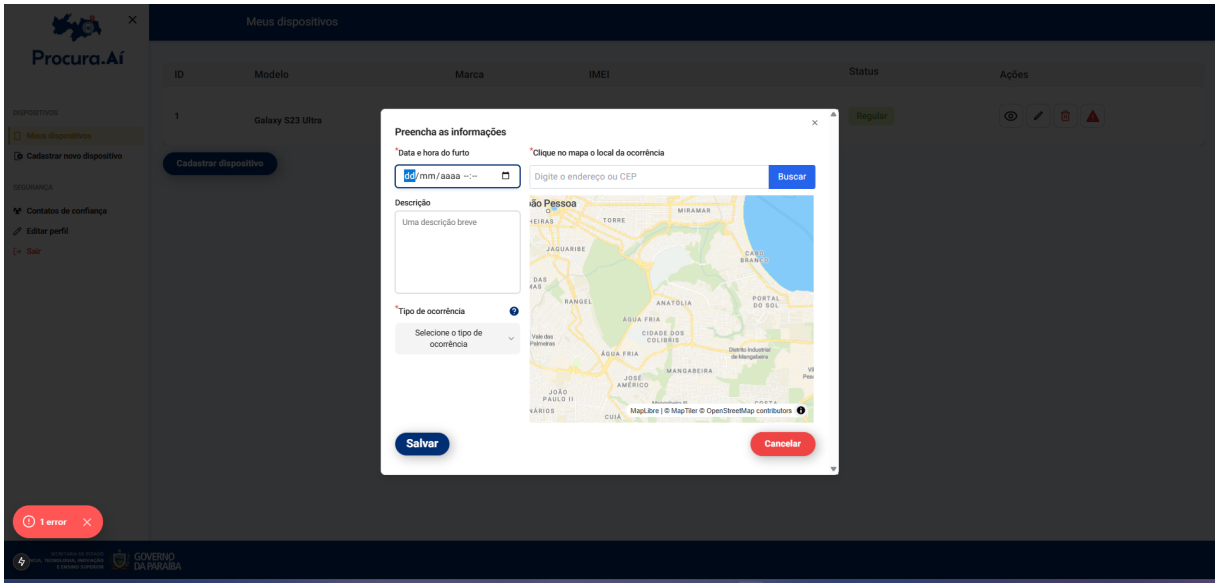

Figura 22: Tela de cadastro de dispositivos

3.7.3.5 Contatos de confiança

Na Figura 23 é apresentado a lista de contatos de confiança, cadastrada pelo usuário logado. Além disso, na Figura 24 é apresentado os modais de cadastro e edição de um contato.

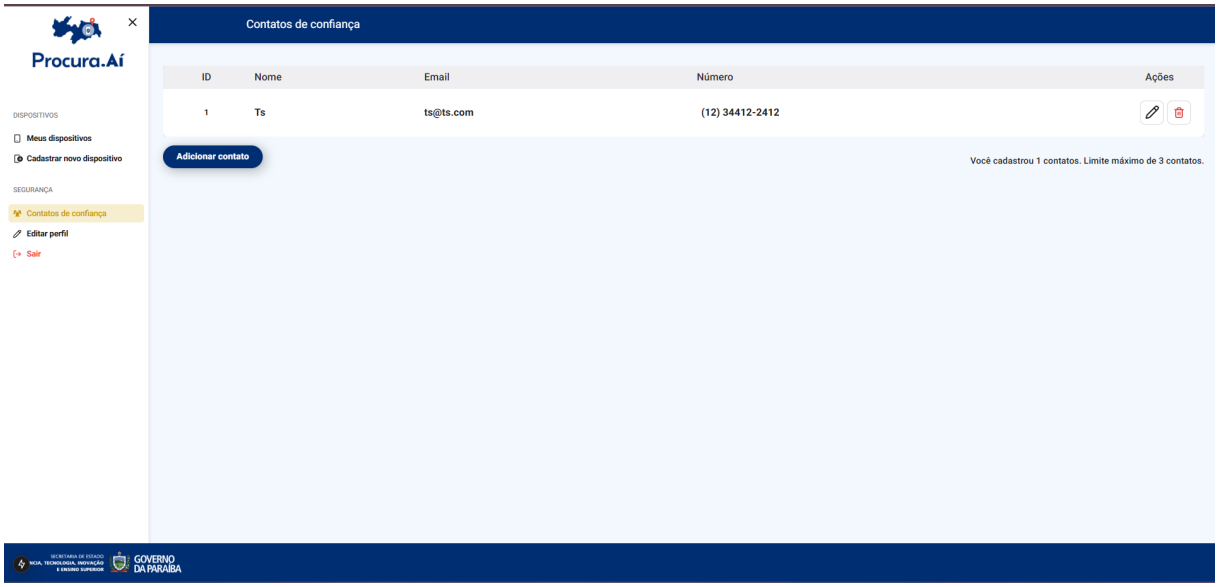

Figura 23: Cadastro de recureção de dispositivo

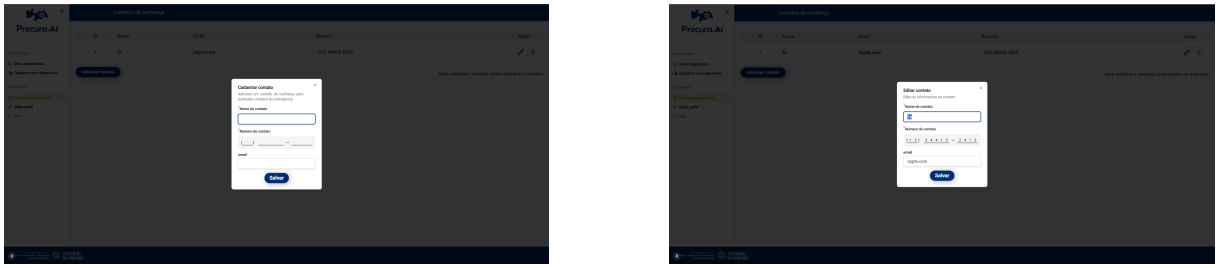

Figura 24: Cadastro de recuperação de dispositivo

3.7.3.6 Edição do perfil

A Figura 25 apresenta o formulário para edição dos dados do perfil do usuário logado, que foram cadastrados previamente, funcionando da mesma forma que o fluxo do usuário Admin.

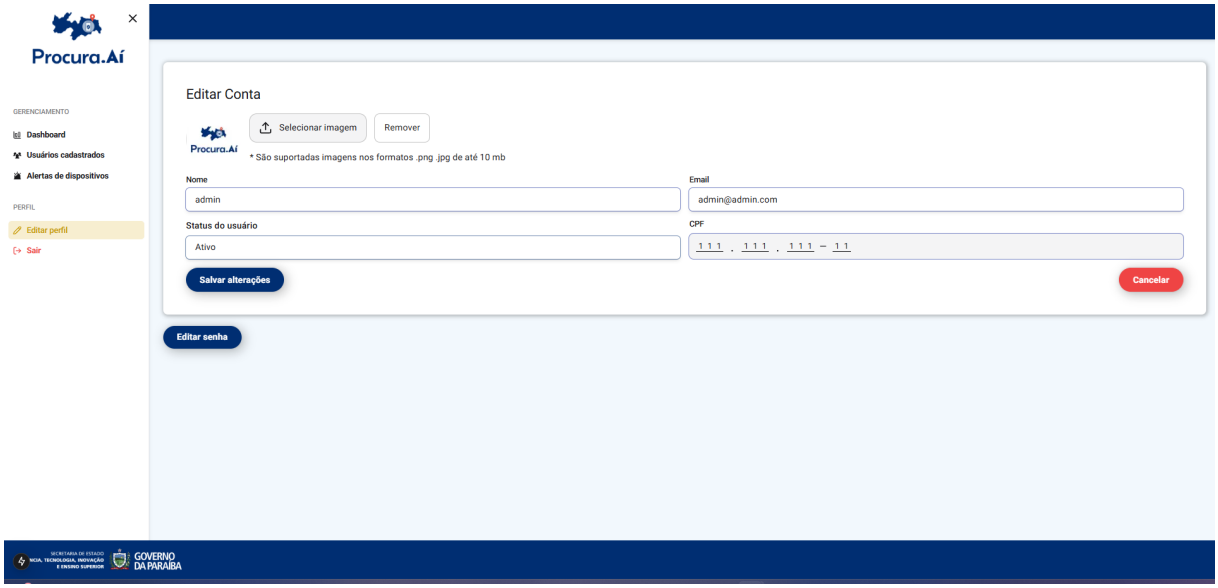

Figura 25: Tela de edição do perfil

4 ANÁLISE DOS RESULTADOS

Nesta seção será introduzida a ferramenta MermaidChart, utilizada para documentar a arquitetura do sistema por meio de diagramas visuais baseados no modelo C4. Essa ferramenta permite a criação de diagramas de forma simples e integrada à documentação, contribuindo para a melhor compreensão da estrutura e do funcionamento da aplicação. Além disso, serão apresentados os resultados obtidos na execução dos casos de teste definidos pelas principais funcionalidades da aplicação, incluindo a análise de sucesso ou falha de cada funcionalidade analisada.

4.1 Análise de Documentação

A criação da documentação da arquitetura foi realizada de maneira prática e ágil, graças à simplicidade encontrada em utilizar a ferramenta *MermaidChart*, que transforma os textos estruturados em diagramas compreensíveis e de melhor visualização. Para que isso fosse possível, se fez necessário o uso de comandos específicos da sintaxe do *Mermaid* como *graph TD* para definir fluxos direcionais e *classDiagram* para representar a estrutura de classes.

Durante a elaboração, foi possível criar diferentes níveis de abstração do sistema, como os diagramas de contexto, containers, componentes e classes, seguindo o modelo C4. A principal vantagem encontrada foi a capacidade de acrescentar ajustes rapidamente sobre os desenhos, permitindo ajustes dinâmicos sem necessidade de ferramentas gráficas complexas, como no exemplo da Figura 26. Além disso, a compatibilidade do Mermaid com diversos editores (como VSCode e Mermaid Live Editor) facilitou a validação visual dos diagramas ao longo do processo de documentação. A experiência com a ferramenta se mostrou eficaz e o uso do Mermaid é uma excelente escolha para documentações técnicas que exigem clareza, padronização e agilidade na atualização dos artefatos arquiteturais.

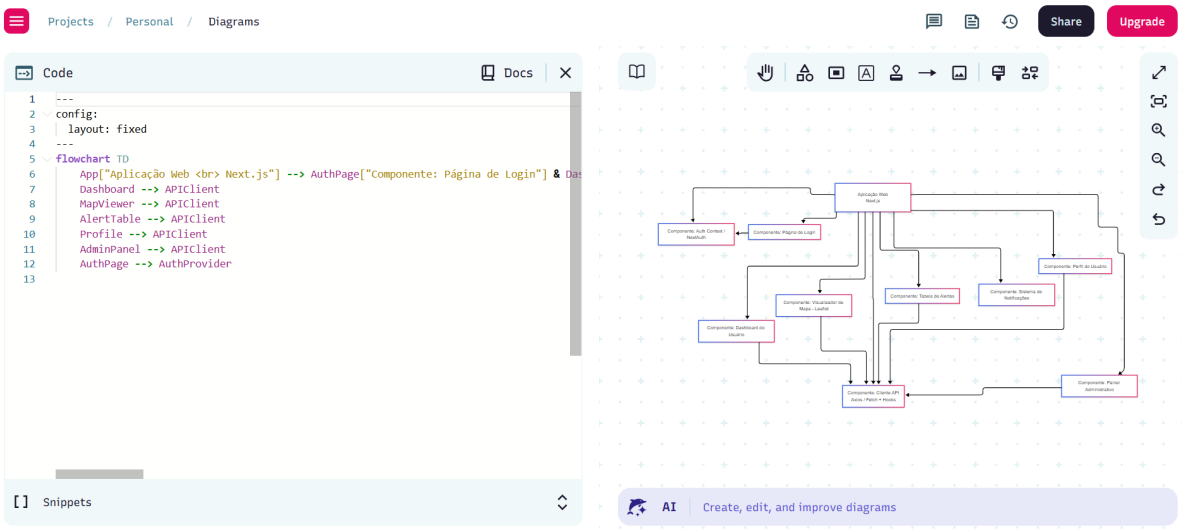

Figura 26: Tela de edição de diagramas no Mermaid Chart

4.2 Análise de Testes

A seguir serão descritos os casos de teste elaborados para validar as principais funcionalidades da aplicação de prevenção ao roubo e furto de celulares. Cada caso de teste tem como objetivo assegurar que os fluxos de cadastro, autenticação, gerenciamento de dispositivos, emissão de alertas, visualização de dados no mapa, além de funcionalidades administrativas e de suporte ao usuário, estejam funcionando conforme o esperado. Os testes abrangem tanto operações básicas realizadas pelo usuário comum (como registrar um dispositivo, emitir um alerta, ou editar o perfil), quanto funcionalidades específicas para administradores (como acesso ao painel de gestão e exportação de relatórios).

Cada caso de teste será apresentado com uma estrutura, contendo sua descrição, entradas, resultado esperado, resultado obtido, análise do resultado e observações relevantes. O conjunto de testes foi pensado para cobrir os principais cenários de uso da aplicação, garantindo que o sistema seja confiável, seguro e eficiente.

Caso de Teste T01 - Cadastro de Novo Usuário	
Descrição	Avaliar o processo de criação de um novo usuário na aplicação, preenchendo informações obrigatórias.
Entradas	Usuário informa nome, cpf, e-mail, confimação de e-email, senha e confirmação de senha no formulário de registro e clica em 'Criar Conta'.
Resultado Esperado	Usuário registrado com sucesso, exibindo mensagem de confirmação e redirecionamento para a tela de login.
Resultado Obtido	Bem-sucedido
Análise do Resultado	Todas as validações foram realizadas corretamente e o fluxo foi concluído sem erros.
Observações	Campos obrigatórios marcados corretamente; senha exigindo no mínimo 8 caracteres.

Tabela 1: Caso de teste T01 - Cadastro de Novo Usuário

Caso de Teste T02 - Login com Credenciais Válidas	
Descrição	Avaliar se o usuário consegue se autenticar com e-mail e senha válidos.
Entradas	Usuário preenche e-mail e senha corretamente e clica em 'Entrar'.
Resultado Esperado	Usuário redirecionado ao dashboard com sessão ativa, caso tenha perfil admin. Caso tenha perfil de usuário é redirecinado para a tela de dispositivos
Resultado Obtido	Bem-sucedido
Análise do Resultado	Login realizado com sucesso e tokens armazenados corretamente.
Observações	Tempo de resposta satisfatório.

Tabela 2: Caso de teste T02 - Login com Credenciais Válidas

Caso de Teste T03 - Registro de Dispositivo	
Descrição	Avaliar se o usuário consegue adicionar um dispositivo associado à sua conta.
Entradas	Usuário acessa a seção 'Adicionar Dispositivo', preenche marca, modelo, IMEI e número do celular, e confirma.
Resultado Esperado	Novo dispositivo listado com status 'Ativo'.
Resultado Obtido	Bem-sucedido
Análise do Resultado	Dispositivo salvo corretamente na base de dados.
Observações	Validação de formato de IMEI funcionando adequadamente.

Tabela 3: Caso de teste T03 - Registro de Dispositivo

Caso de Teste T04 - Emissão de Alerta de Furto	
Descrição	Avaliar se o usuário consegue emitir um alerta em caso de roubo de celular.
Entradas	Usuário seleciona o dispositivo, preenche descrição, tipo do alerta, data e hora do evento, marca no mapa a localização e envia o alerta.
Resultado Esperado	Alerta criado e exibido no mapa e na tabela de alertas.
Resultado Obtido	Bem-sucedido
Análise do Resultado	Alerta corretamente associado à localização fornecida.
Observações	Mensagem de confirmação exibida corretamente.

Tabela 4: Caso de teste T04 - Emissão de Alerta de Furto

Caso de Teste T05 - Visualização de Alertas no Mapa	
Descrição	Verificar se os alertas emitidos aparecem corretamente no mapa interativo para usuários com perfil de administrador.
Entradas	Usuário acessa a seção de dashboard.
Resultado Esperado	Marcadores de alertas visíveis nas localizações corretas.
Resultado Obtido	Bem-sucedido
Análise do Resultado	Dados exibidos conforme esperado.
Observações	Carregamento se torna menos eficiente com grandes volumes de dados.

Tabela 5: Caso de teste T05 - Visualização de Alertas no Mapa

Caso de Teste T06 - Editar Perfil do Usuário	
Descrição	Avaliar se o usuário consegue atualizar seus dados pessoais.
Entradas	Usuário acessa perfil, altera os dados e clica em 'Salvar'.
Resultado Esperado	Dados atualizados e refletidos na tela de perfil.
Resultado Obtido	Bem-sucedido
Análise do Resultado	Validações aplicadas corretamente e operação concluída.
Observações	Nenhuma.

Tabela 6: Caso de teste T06 - Editar Perfil do Usuário

Caso de Teste T07 - Logout do Sistema	
Descrição	Verificar se o usuário consegue encerrar sua sessão de forma segura.
Entradas	Usuário clica no botão de logout, localizado no menu lateral.
Resultado Esperado	Redirecionamento para tela de login e remoção de tokens locais.
Resultado Obtido	Bem-sucedido
Análise do Resultado	Sessão encerrada corretamente.
Observações	Nenhuma.

Tabela 7: Caso de teste T07 - Logout do Sistema

Caso de Teste T08 - Adicionar Contato de Confiança	
Descrição	Verificar se é possível adicionar contatos para suporte em caso de roubo.
Entradas	Usuário acessa 'Contatos de Confiança' no menu lateral, preenche nome, telefone e email, e confirma.
Resultado Esperado	Contato adicionado à lista.
Resultado Obtido	Bem-sucedido
Análise do Resultado	Contato salvo com sucesso.
Observações	Validação de campos obrigatórios realizada.

Tabela 8: Caso de teste T08 - Adicionar Contato de Confiança

Caso de Teste T09 - Recuperação de Senha	
Descrição	Avaliar o processo de recuperação de senha via e-mail.
Entradas	Usuário clica em 'Esqueci minha senha', informa o e-mail e envia.
Resultado Esperado	Link de recuperação enviado para o e-mail informado.
Resultado Obtido	Mal-sucedido
Análise do Resultado	Redirecionamento para tela de recuperação de senha não concluído.
Observações	Necessário criar tela de recuperação de senha e integrar as funcionalidades.

Tabela 9: Caso de teste T09 - Recuperação de Senha

Caso de Teste T10 - Filtros de Alertas	
Descrição	Verificar se é possível filtrar alertas na seção de Alertas de dispositivos, localizado no menu lateral.
Entradas	Usuário seleciona status, marca e data, podendo também filtrar por IMEI e nome do proprietário
Resultado Esperado	Somente alertas relacionados aos filtros selecionados são exibidos.
Resultado Obtido	Bem-sucedido
Análise do Resultado	Filtro aplicado corretamente.
Observações	Nenhuma.

Tabela 10: Caso de teste T10 - Filtro de Alertas por Localização

Caso de Teste T11 - Visualizar Painel Administrativo	
Descrição	Verificar se administradores podem acessar o dashboard de gestão.
Entradas	Administrador acessa a seção Dashboard, no menu lateral.
Resultado Esperado	Dashboard com métricas e gráficos visíveis.
Resultado Obtido	Bem-sucedido
Análise do Resultado	Acesso restrito validado corretamente.
Observações	Perfil comum bloqueado corretamente.

Tabela 11: Caso de teste T11 - Visualizar Painel Administrativo

Caso de Teste T12 - Excluir Dispositivo	
Descrição	Verificar se o usuário pode excluir um dispositivo de sua conta.
Entradas	Usuário acessa lista de dispositivos, clica em excluir e confirma a ação.
Resultado Esperado	Dispositivo removido da conta.
Resultado Obtido	Bem-sucedido
Análise do Resultado	Exclusão processada com sucesso.
Observações	Dispositivos com alerta ativo podem ser excluídos.

Tabela 12: Caso de teste T12 - Excluir Dispositivo

Caso de Teste T13 - Exportar Tabela de Usuários e Alertas	
Descrição	Verificar a funcionalidade de exportar os dados da tabela de usuários e alertas, para usários administradores.
Entradas	Usuário clica em 'Exportar planilha' na tabela de usuários e escolhe se quer exportar a lista de usuários e/ou lista de alertas.
Resultado Esperado	Arquivo CSV/XLS da lista de usuários e/ou alertas gerados corretamente.
Resultado Obtido	Parcialmente bem-sucedido
Análise do Resultado	Dados de usuários exportados corretamente. Porém lista de alertas apresentou erro ao exportar.
Observações	Necessários validar exportação de alertas.

Tabela 13: Caso de teste T13 - Exportar Tabela de Alertas

Caso de Teste T14 - Mudar Status do Dispositivo para Recuperado	
Descrição	Verificar se o usuário pode alterar o status de um dispositivo para 'Recuperado'.
Entradas	Usuário edita o status do dispositivo roubado para recuperado.
Resultado Esperado	Status atualizado para 'Recuperado'.
Resultado Obtido	Bem-sucedido
Análise do Resultado	Atualização refletida corretamente no sistema.
Observações	Nenhuma.

Tabela 14: Caso de teste T14 - Mudar Status do Dispositivo para Recuperado

4.3 Discussão dos resultados

A ferramenta de criação de diagramas propôs uma experiência simplificada de documentação da aplicação, já que disponibiliza as principais funcionalidades para criação de diagramas no geral, seja diagramas de caso de uso, diagramas de contexto, digramas de container, diagramas de classe etc. Além de receber códigos em formato mermaid que são convertidos em diagramas, o que agiliza o processo de criação e manuteção dos mesmos.

Com a análise dos resultados dos testes foi possível observar que a aplicação Procura.Aí atendeu de forma satisfatória a maioria dos testes propostos, o que demonstra que a ferramenta propõe um ambiente estável, seguro e de fácil usabilidade para se ter uma maior eficiência na denúncia e registro de dispositivos roubados e furtados.

Os testes das funcionalidades essenciais, como cadastro de usuários, login, registro de dispositivos, visualização de dashboard e emissão de alertas, foram concluídos sem falhas, evidenciando que os fluxos principais da aplicação estão funcionando corretamente, de forma consistente e segura. A autenticação de usuários e o gerenciamento de sessão e a proteção dos dados armazenados indicam que as boas práticas de segurança foram aplicadas adequadamente dentro do escopo proposto.

5 CONCLUSÕES E TRABALHOS FUTUROS

O trabalho apresentado mostra que há uma carência de sistemas que, além de informativos, ofereçam recursos para registro e acompanhamento contínuo do estado clínico dos pacientes. Também é notável que as soluções existentes são, em sua maioria, sistemas que não tem uma comunicação efetiva entre os cidadãos e os orgãos responsáveis por combater crimes de furtos e roubos de dispositivos móveis. A utilização de tecnologias e ferramentas modernas, como ReactJs, NextJs, Tailwind, AppWrite e Leaflet, foi fundamental para garantir a escalabilidade e a eficiência da aplicação, além de proporcionar uma arquitetura bem estruturada baseada no modelo C4, favorecendo a manutenção e evolução contínua do sistema. Além disso, a integração de práticas como autenticação segura, mapeamento de eventos e visualização de dados em tempo real demonstra uma proposta de projeto para apoiar a segurança pública e os usuários que foram afetados por crimes relacionados.

Embora a aplicação tenha atingido seus objetivos iniciais, algumas melhorias e expansões podem ser propostas para o futuro:

- Aprimoramento de inteligência de dados: Implementar análises estatísticas e modelos preditivos para identificar padrões de roubo e sugerir ações preventivas para os usuários.
- Notificações em tempo real via push: Implementar o uso do Firebase Cloud Messaging para notificar usuários próximos a novas ocorrências de roubo em sua região.
- Integração com sistemas de segurança pública: Estabelecer comunicação direta com bancos de dados oficiais para agilizar a validação de dispositivos recuperados e denúncias de roubos.
- Responsividade de telas: Aumentar a responsividade de telas para abranger uma maior quantidade de dispositivos a utilizar a aplicação.
- Recuperação de senha com envio de email: Criar a funcionalidade de envio de emails e integrar com a recuperação de senha.
- Dispositivos móveis: Criar versão da aplicação de forma nativa para dispositivos móveis.

Essas melhorias tem como objetivo tornar o sistema ainda mais eficiente, abrangente e alinhado às necessidades dos usuários, promovendo um ambiente mais seguro e colaborativo no combate ao roubo de celulares.

REFERÊNCIAS

[1] AMAZON. AWS Amplify. Disponível em: https://aws.amazon.com/amplify/. Acesso em: 11 de fevereiro de 2025.

[2] API BRASIL. As tendências futuras para APIs. Disponível em: https://apibrasil.blog/as-tendencias-futuras-para-apis/. Acesso em: 29 de março de 2025.

[3] APPWRITE. Documentação oficial do Appwrite. Disponível em: https://appwrite.io/docs andReduceHarm.aspx. Acesso em: 11 de fevereiro de 2025.

[4] BALLA, Dániel; GEDE, Mátyás. **Beautiful thematic maps in Leaflet with automatic data classification. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, v. 48, p. 3-10, 2024**. Disponível em: https://isprs-archives.copernicus.org/articles/XLVIII-4-W12-2024/3/2024/isprs-archives-XLVIII-4-W12-2024-3-2024.html

[5] BALLAMUDI, V. K. R. et al. **Getting Started Modern Web Development with Next. js: An Indispensable React Framework. Digitalization & Sustainability Review, v. 1, n. 1, p. 1-11, 2021.** Disponível em: https://encurtador.com.br/A3mT2

[6] BRITO, Carlos Eduardo Carvalho de. **Interopelabilidade Dos Sistemas Informatizados na Segurança Pública. 2018.** Disponível em: https://repositorio.esg.br/bitstream/123456789/870/1/CARLOS%20EDUARDO%20CARVALI
%20VF.pdf

[7] DIOGO, Miguel; CABRAL, Bruno; BERNARDINO, Jorge. **Consistency models of NoSQL databases. Future Internet, v. 11, n. 2, p. 43.** 2019. Disponível em: https://www.mdpi.com/1999-5903/11/2/43

[8] FÓRUM BRASILEIRO DE SEGURANÇA PÚBLICA. **18º Anuário Brasileiro de Segurança Pública. São Paulo: Fórum Brasileiro de Segurança Pública, 2024.** Disponível em: https://publicacoes.forumseguranca.org.br/handle/123456789/253.

[9] GOOGLE. Firebase. Disponível em: https://firebase.google.com/. Acesso em: 11 de fevereiro de 2025.

[10] HERTZ, Günter Matheus. **Desenvolvimento de aplicação mobile para avaliação de docentes e disciplinas**. 2022. Disponível em: https://lume.ufrgs.br/handle/10183/252445

[11] KODALI, Nikhil. **Tailwind CSS Integration in Angular: A Technical Overview. International Journal of Innovative Research in Science Engineering and Technology, v. 13, n. 16652, p. 10.15680. 2024** . Disponível em: https://encurtador.com.br/rY6If

[12] LAZUARDY, Mochammad Fariz Syah; ANGGRAINI, Dyah. **Modern front end web architectures with react. js and next. js. Research Journal of Advanced Engineering and Science, v. 7, n. 1, p. 132-141, 2022.** Disponível em: https://irjaes.com/wp-content/uploads/2022/02/IRJAES-V7N1P162Y22.pdf

[13] LI, Nian; ZHANG, Bo. **The Research on Single Page Application Front-end development Based on Vue. 2021.** Disponível em: https://iopscience.iop.org/article/10.1088/1742-6596/1883/1/012030/meta

[14] MACHADO, Isaac Newton Melo. **Avaliação de Plataformas Serverless que implementam containers-as-a-service**. 2022. Tese de Doutorado. Disponível em: https://comum.rcaap.pt/handle/10400.26/40152

[15] PAES-MACHADO, Eduardo; MALTEZ, Juliana Campos. **A segurança quotidiana contra roubos de telefones celulares. Dilemas: Revista de Estudos de Conflito e Controle Social, Rio de Janeiro, v. 17, n. 3, e62784, 2024.** Disponível em: https://www.scielo.br/j/dilemas/a/VrsBGKFJHNBBkSqQT85kgmx/?format=pdf&lang=pt

[16] PARSE PLATFORM. Parse. Disponível em: https://parseplatform.org/. Acesso em: 11 de fevereiro de 2025.

[17] SUPABASE. Supabase. Disponível em: https://supabase.io/. Acesso em: 11 de fevereiro de 2025.

[18] SPILLNER, Josef. **Quantitative Analysis of Cloud Function Evolution in the AWS Serverless Application Repository**. 2019. Disponível em: https://arxiv.org/abs/1905.04800

[19] TALEB, Yacine et al. **Tailwind: Fast and Atomic RDMA-based Replication. In: 2018 USENIX Annual Technical Conference (USENIX ATC 18)**. 2018. p. 851-863. Disponível em: https://www.usenix.org/conference/atc18/presentation/taleb

[20] WEN, Jinfeng; CHEN, Zhenpeng, JIN, Xin; LIU, Xuanzhe. **Rise of the Planet of Serverless Computing: A Systematic Review. 2022**. Disponível em: https://arxiv.org/abs/2206.12275

APÊNDICE A – EXEMPLO IMPLEMENTAÇÃO DA DOCUMENTAÇÃO DO DIAGRAMA DE CONTEXTO

```
graph TD
    User[Usuario]
    App[Procura.Ai<br>Aplicacao para prevencao de furto de celulares]
    Frontend[Frontend Web App<br>NextJs, ReactJs, Leaflet]
    Backend[Backend API<br>AppWrite, Firebase]
    Auth[Autenticacao<br>NextAuth]

    User -->|Acessa| App
    App -->|Renderiza e interage com| Frontend
    App -->|Consulta e atualiza dados| Backend
    App -->|Autenticacao| Auth
    Auth -->|Valida credenciais| Backend
```

APÊNDICE B – EXEMPLO IMPLEMENTAÇÃO DA DOCUMENTAÇÃO DO DIAGRAMA DE CONTAINER

```
graph TD
    User[Usuario]

    App[Aplicacao Web<br>NextJs + ReactJs + Tailwind]
    Auth[NextAuth<br>Servico de autenticacao]
    Frontend[Interface<br>ReactJs + Leaflet]
    Backend[API Backend<br>AppWrite + Firebase]
    DB[Banco de Dados<br>AppWrite]
    Map[Servico de Mapa<br>Leaflet / OSM]

    User -->|Acessa| App
    App -->|Renderiza UI| Frontend
    App -->|Autentica| Auth
    App -->|Requisicoes| Backend
    Backend -->|Leitura/Escrita| DB
    Frontend -->|Exibe mapa| Map
    Auth -->|Valida credenciais| DB
```

APÊNDICE C – EXEMPLO IMPLEMENTAÇÃO DA DOCUMENTAÇÃO DO DIAGRAMA DE COMPONENTE

```
---
config:
  layout: fixed
---
flowchart TD
    App["Aplicacao Web <br> NextJs"] -->
    AuthPage["Componente: Pagina de Login"] &
    Dashboard["Componente: Dashboard do Usuario"] &
    MapViewer["Componente: Visualizador de Mapa - Leaflet"] &
    AlertTable["Componente: Tabela de Alertas"] &
    Profile["Componente: Perfil do Usuario"] &
    AdminPanel["Componente: Painel Administrativo"] &
    APIClient["Componente: Cliente API<br>Axios / Fetch + Hooks"] &
    AuthProvider["Componente: Auth Context / NextAuth"] &
    Notifications["Componente: Sistema de Notificacoes"]
    Dashboard --> APIClient
    MapViewer --> APIClient
    AlertTable --> APIClient
    Profile --> APIClient
    AdminPanel --> APIClient
    AuthPage --> AuthProvider
```

APÊNDICE D – EXEMPLO IMPLEMENTAÇÃO DA DOCUMENTAÇÃO DO DIAGRAMA DE CÓDIGO

```
classDiagram

class Usuario {
  +string id
  +string nome
  +string email
  +string senha
  +string telefone
  +List~Dispositivo~ dispositivos
}

class Dispositivo {
  +string id
  +string modelo
  +string imei
  +string status
  +string localizacao
  +string dataCadastro
  +Usuario dono
  +List~AlertaOcorrencia~ alertas
}

class AlertaOcorrencia {
  +string id
  +string tipo
  +string descricao
  +Date data
  +string localizacao
  +Dispositivo dispositivo
}
```

```
class DispositivoRecuperado {
  +string id
  +string informacoesGerais
  +string orgaoResponsavel
  +Date setorResponsavel
  +string localRetirada
  +bool notificarProprietario
  +bool notificarContatos
  +string documentos
  +Dispositivo dispositivo
  +Usuario criador
}

class ContatoConfianca {
  +string id
  +string nome
  +string telefone
  +Usuario vinculadoA
}

Usuario --> "1..*" Dispositivo
Dispositivo --> "0..*" AlertaOcorrencia
Dispositivo --> "0..*" DispositivoRecuperado
Usuario --> "0..*" ContatoConfianca
```