APLICATIVO MOBILE: POVO DENUNCIA

INTEGRANTES:
ALCIMAR ROSAL BENVINDO
LÍGIA KAROLINNE ARAÚJO

SUMÁRIO

1) INTRODUÇÃO:

Este relatório apresenta uma visão geral do desenvolvimento do aplicativo Povo Denuncia, criado com o intuito de oferecer à população uma plataforma acessível e eficiente para o registro de denúncias relacionadas a problemas urbanos, como abandono, buracos nas ruas, lixo abandonado e casos de violência. O projeto foi desenvolvido utilizando Flutter, integrando funcionalidades modernas como geolocalização, mapeamento interativo e armazenamento local de dados via SQLite.

1.1) Objetivo

O principal objetivo do aplicativo Povo Denuncia é permitir que cidadãos possam registrar e visualizar denúncias relacionadas ao seu entorno de forma prática e segura. A proposta é que cada usuário consiga inserir informações detalhadas sobre ocorrências, como título, uma breve descrição, endereço, tipo de denúncia, data e uma imagem ilustrativa. A ferramenta visa ser um canal de apoio à cidadania, fornecendo visibilidade a problemas reais enfrentados por comunidades locais.

1) FUNCIONALIDADE IMPLEMENTADAS

Nesta seção, são descritas as funcionalidades que foram efetivamente desenvolvidas no aplicativo. O objetivo é apresentar, de forma clara, quais ações os usuários podem realizar dentro do sistema e quais recursos foram incluídos para garantir uma experiência completa e funcional.

- Tela de cadastro e login de usuários
- Feed com visualização das denúncias cadastradas.

- Cadastro de novas denúncias com título, descrição, tipo, data, imagem e endereço.

- Conversão automática do endereço em coordenadas geográficas.
- Mapa interativo exibindo a localização da denúncia.
- Visualização detalhada de cada denúncia.
- Edição e exclusão de denúncias.
- Logout do sistema.

2) EXPLICAÇÃO TÉCNICA

A seguir, há uma explicação técnica resumida dos principais componentes do aplicativo. O objetivo é fornecer uma visão geral da estrutura do código, destacando como as partes mais importantes foram organizadas, implementadas e integradas entre si, a fim de garantir a funcionalidade do sistema.

- As telas do app estão organizadas na pasta lib/screens , promovendo clareza e manutenção facilitada. Além disso, há a pasta “db” com o arquivo “db_helper”(banco de dados), a pasta “models” com os arquivos de modelo de denúncia e de usuário.
- *lib/screens/login_screen:* O código implementa uma tela única de login e cadastro em Flutter. Ele usa controladores para capturar e-mail e senha, e alterna entre os modos com a variável isLogin. A função principal autentica o usuário ou realiza o cadastro no banco de dados usando o DBHelper. Mensagens de erro ou sucesso são exibidas com SnackBar, e após login bem-sucedido, o usuário é redirecionado para a tela de feed. A interface inclui campos de texto e botões para ação e alternância entre login e cadastro.
- lib/screens/feed_screen: é a tela principal do código, que exibe uma lista de denúncias salvas no banco de dados local. Ele carrega os dados ao iniciar (_loadReports) e permite que o usuário visualize, edite

ou exclua cada denúncia ao tocar em um item da lista, usando um BottomSheet com essas opções. O botão flutuante abre a tela para cadastrar nova denúncia. O botão com imagem de perfil permite fazer logout após uma confirmação. A interface usa ListView.builder para montar os cards de denúncias, exibindo imagem, título e descrição. Ao excluir uma denúncia, ela é removida do banco e a lista é atualizada.

- lib/screens/new_denuncia: implementa a tela de criação e edição de denúncias no app. Ele permite ao usuário inserir título, descrição, tipo, data, endereço (convertido automaticamente em coordenadas geográficas), imagem (da câmera ou galeria) e selecionar uma localização no mapa com um toque. O formulário é validado antes de salvar, e dependendo se a denúncia já existe, ela é criada ou atualizada no banco SQLite. O mapa exibe um marcador na posição escolhida, e o botão final salva a denúncia e retorna à tela anterior.
- lib/screens/edit_denuncia: define a tela de edição de denúncias, preenchendo automaticamente os campos com os dados recebidos. Ele permite alterar título, descrição, tipo, data, endereço e imagem, além de atualizar a localização no mapa com base no novo endereço digitado. O mapa exibe um marcador na posição correspondente, e a data/hora pode ser escolhida via seletores visuais. Ao salvar, os dados validados são atualizados no banco SQLite e o usuário é redirecionado de volta.
- lib/sreens/denuncia_detalhes: define a tela de visualização dos detalhes de uma denúncia, exibindo a imagem, título, descrição, tipo, data e um mapa com a localização marcada. Os dados são recebidos via parâmetro.
- lib/db/db_helper: define a classe DBHelper, que gerencia a conexão com o banco SQLite local, criando as tabelas de usuários e denúncias ao inicializar; ela oferece métodos para cadastrar, autenticar e buscar usuários, além de inserir, listar, atualizar e deletar denúncias.
- lib/models/user_model: A classe User representa um usuário com os atributos id, email e senha, fornecendo métodos para converter instâncias em mapas (toMap) e criar objetos a partir de mapas

(fromMap), facilitando a manipulação dos dados no banco de dados SQLite.

- lib/models/denuncia_model: A classe Denuncia modela uma denúncia com os campos id, titulo, descricao, tipo, imagemPath, latitude, longitude e dataHora, todos necessários para representar as informações da denúncia. Ela inclui o método toMap() para converter uma instância em um mapa, adequado para armazenamento no banco de dados, e um construtor factory fromMap() que cria uma instância de Denuncia a partir de um mapa.
- lib/main: O código define a aplicação Flutter principal, começando pela função main() que executa o app instanciando MyApp. A classe MyApp é um StatelessWidget que retorna um MaterialApp configurado com o título "App de Denúncias". A tela inicial (home) é definida como LoginScreen. Além disso, o app configura rotas nomeadas para navegação, incluindo a rota /feed que leva à tela FeedScreen e a rota /new que abre a tela NewDenunciaScreen.

3) API UTILIZADA

A funcionalidade de geolocalização do aplicativo depende de APIs externas que possibilitam a exibição de mapas e a conversão de endereços em coordenadas. Esta seção apresenta as APIs utilizadas, citando a fonte e explicando sua relevância para o projeto.

- API's utilizadas: Maps SDK for Android,Geocoding API
- URL oficial: https://developers.google.com/maps
- Pacotes Flutter utilizados via Pub.dev: google_maps_flutter,geocoding,sqflite,path_provider

4) ARMAZENAMENTO LOCAL:

O app utiliza uma estratégia de armazenamento local baseada em banco de dados com o uso do SQLite.

- O aplicativo utiliza a biblioteca sqflite para acesso a um banco de dados SQLite.

- A estrutura da base de dados inclui campos como titulo, descricao, tipo, imagemPath, latitude, longitude e data.

- A criação e o gerenciamento da base são centralizados na classe DBHelper, que fornece métodos para inserir, atualizar, deletar e buscar usuários e denúncias.

- As imagens são salvas localmente no dispositivo, e apenas o caminho (path) é armazenado no banco.

- Os dados geográficos são armazenados como double, representando latitude e longitude.

5) CONSIDERAÇÕES FINAIS

O desenvolvimento do aplicativo **Povo Denuncia** resultou em uma ferramenta funcional e acessível, capaz de atender à demanda por um canal prático para a população registrar denúncias locais. A utilização do Flutter garantiu uma interface moderna e responsiva, enquanto o SQLite proporcionou um armazenamento local confiável e eficiente. A integração com a API do Google Maps enriqueceu a experiência do usuário, possibilitando a localização geográfica precisa das denúncias.

Entretanto, para tornar o aplicativo verdadeiramente pronto para uso público e escalável, algumas melhorias são recomendadas. A implementação de um backend em nuvem permitiria a sincronização dos dados entre dispositivos, evitando perda de informações e facilitando a análise centralizada das denúncias. Adicionalmente, a adoção de um sistema de

autenticação seguro garantiria a proteção da privacidade dos usuários e permitiria o controle de acesso adequado. Outra evolução importante seria a incorporação de notificações para acompanhamento do status das denúncias, aumentando o engajamento e a transparência.

Ainda assim, mesmo em sua versão atual, já demonstra viabilidade prática e relevância social, cumprindo o objetivo de facilitar o exercício da cidadania.