Projeto - Emburacar

- Participantes -
- Ygor Jivago Leal Félix ;
- Augusto Cesar ;
- Talyson Machado;

1. VISÃO GERAL

O projeto consiste em um sistema web de denúncia de buracos nas ruas, onde usuários podem submeter formulários com dados e fotos georreferenciadas. O back-end é desenvolvido com Apache Tomcat e se comunica com um banco de dados PostgreSQL. As imagens são enviadas e armazenadas no Google Drive através da API Google Drive v3.

2. ESTRUTURA DO PROJETO
---

a) FormularioController.java

Responsável por processar requisições HTTP POST no endpoint /upload.
- Recebe os campos de formulário enviados pelo frontend (localização, risco, tamanho, descrição e imagem).
- Realiza o parsing do arquivo de imagem e extrai metadados (data e coordenadas GPS).
- Usa a classe GeografiaReversa para obter nome de rua e bairro a partir das coordenadas.
- Classifica o buraco com base em tamanho e risco.
- Envia a imagem para o Google Drive através do serviço GoogleDriveService.
- Persiste os dados no banco de dados PostgreSQL utilizando JDBC..

b) GoogleDriveService.java

Serviço responsável por encapsular a lógica de upload de arquivos para o Google Drive. Utiliza a classe utilitária DriveUploader para realizar a autenticação e envio do arquivo.

c) DriveUploader.java

Classe utilitária que realiza a autenticação com o Google e efetua o upload de arquivos. Utiliza OAuth2 e o client da API do Google Drive.https://drive.google.com/drive/u/2/folders/1CuJLqPsk0rc3poQyDlhyt5dtSSVDs6ys

d) GeografiaReversa.java

Realiza chamada à API de geolocalização (google street view, location iq) para obter nome de rua e bairro a partir de coordenadas GPS extraídas da imagem.

e) Banco de Dados

Inicialmente era acessado de forma manual via JDBC. Foi utilizado o supabase para hospedagem do banco de dados.

f) Metadados.java

Classe utilitária que extrai informações de metadados da imagem, como data de captura e coordenadas GPS, usando bibliotecas como `metadata-extractor`.

3. DIFICULDADES ENFRENTADAS
---

a) Integração Back-end com Front-end
- O principal desafio foi adaptar a estrutura tradicional de servlets com `HttpServletRequest` para o modelo reativo e anotado do Spring Boot (`@RestController`, `@PostMapping`, `MultipartFile`) em menos de 1 dia, o que não foi possível, fazendo-nos permanecer com apache tomcat.
- Problemas com `rotas` foram frequentes até a configuração adequada do google drive.
- Falta de padronização nas rotas e contratos entre front e back inicialmente.

b) Integração com Banco de Dados
- Dificuldades em configurar o PostgreSQL no início, especialmente com `driver-class-name` e credenciais.
- A abordagem do JDBC tradicional exigiu reorganização estrutural e refatoração do código legado.

c) API do Google Drive
- A autenticação OAuth2 para aplicações Java Server exigiu geração manual de credenciais e manipulação de tokens.
- A documentação da Google API não era direta para Java com upload programático.
- Integração com arquivos temporários e manipulação de streams também geraram bugs intermitentes.

d) Organização do Projeto
- O acoplamento inicial entre controller, lógica de negócio e persistência dificultava a leitura e testes.
- A medida que o projeto evoluiu, foi necessária uma reestruturação com injeção de dependências via `@Servelet`, separando responsabilidades.

4. CONCLUSÃO

O projeto serviu como aprendizado prático de integração fullstack com tecnologias modernas. Apesar das dificuldades iniciais com APIs externas e adaptação ao Spring Boot, a equipe conseguiu implementar um sistema funcional, modularizado e escalável.