C.R.O.N.O.S

Alunos: Rodrigo Eduardo, Tarcisio e Tarcyo

C.R.O.N.O.S.: Central de Repositório e Organização de dados em Nuvem Simplificado.

Desenvolvimento de Sistema de armazenamento na Nuvem:

Neste projeto, propomos a criação de um sistema de armazenamento em nuvem com capacidades de upload e download de arquivos, bem como a diferenciação de usuários com base em categorias de contas que definem suas capacidades de armazenamento. Além disso, implementaremos um sistema de login e autenticação de contas para garantir a segurança e a privacidade dos dados dos usuários.

Repositório no github

Link do repositório: https://github.com/redusilva/ifg-cronos

Documentação

Link da documentação: https://app.swaggerhub.com/apis-docs/RodrigoEduardo347/cronos/1.0.0#

Arquitetura do Sistema

Sistema de Login e Autenticação

- Implementaremos um sistema de login seguro que utiliza hashes e redux para criar sessões únicas, impedindo a clonagem ou sequestro de sessões.

Categorias de Contas

- Os usuários serão diferenciados por categorias de contas que estabelecem limites de capacidade máxima de armazenamento para cada conta.

Banco de Dados

- Utilizaremos o MongoDB para criar dois bancos de dados separados: um para armazenar informações de login e outro para armazenar os arquivos dos usuários. O banco de dados da nuvem será capaz de acomodar todos os formatos de arquivos, deste que não ultrapassem o máximo permitido pela categoria de conta..

Integridade e Confiabilidade dos Arquivos

- Antes de permitir o upload de arquivos, o sistema realizará uma verificação de integridade e confiabilidade por meio da API do VirusTotal. Os arquivos serão enviados ao banco de dados da nuvem somente se forem validados por mais de 90% dos agentes verificadores.

Tecnologias Utilizadas

- O backend será desenvolvido em NodeJS e integrado à API do VirusTotal.
- O frontend será construído no framework Flutter, usando a linguagem Dart.

Benefícios do Sistema

- Segurança de Sessão: O sistema de autenticação robusto garante que as sessões dos usuários sejam seguras contra possíveis ataques.
- Diferenciação de Contas: As categorias de contas proporcionam aos usuários diferentes níveis de armazenamento, atendendo a diversas necessidades.
- Verificação de Arquivos: A verificação com a API do VirusTotal garante que apenas arquivos seguros sejam armazenados na nuvem.
- Compatibilidade de Formatos: O banco de dados da nuvem é capaz de acomodar todos os tipos de arquivos.
- Usabilidade: O frontend Flutter oferece uma interface amigável e responsiva para os usuários.

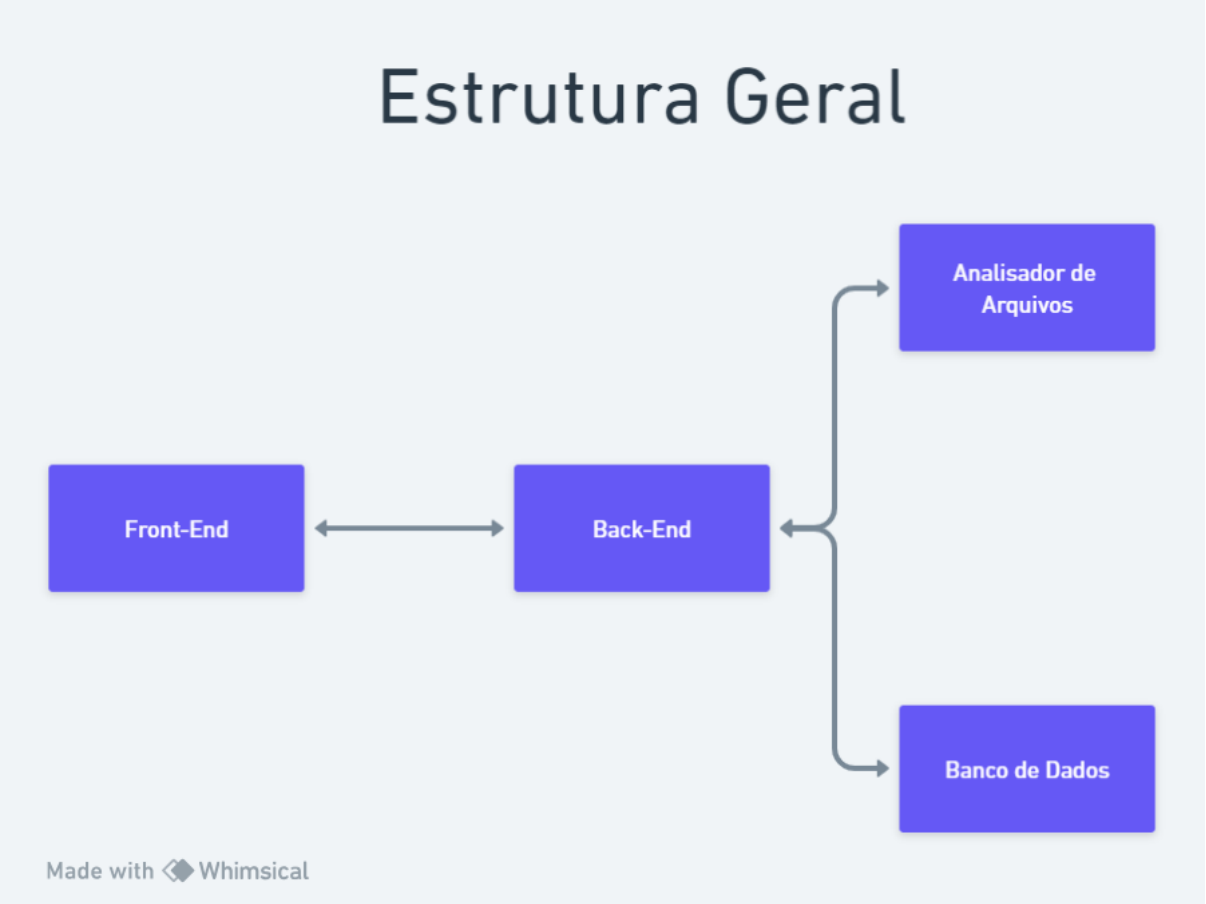
Este projeto visa criar um sistema de armazenamento em nuvem robusto, seguro e acessível, atendendo às necessidades dos usuários e garantindo a integridade de seus dados.

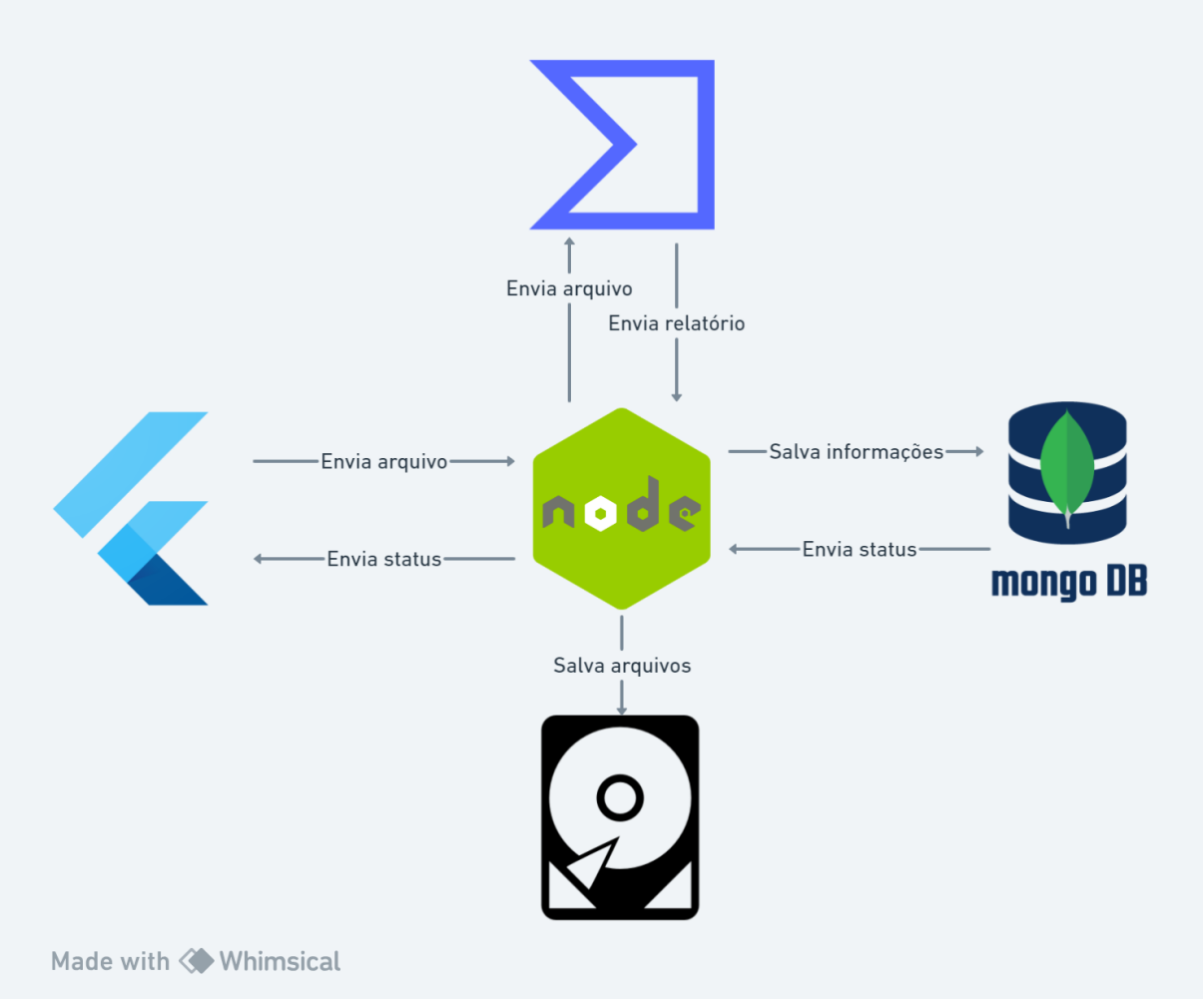
Bibliotecas utilizadas

No backend, serão utilizadas as seguintes bibliotecas:

1. **Express:** É um framework para Node.js que simplifica o processo de criação de aplicativos web e APIs. Ele fornece uma estrutura mínima, mas robusta, para construir servidores HTTP.
2. **Multer:** Uma middleware para lidar com dados de formulários em formato `multipart/form-data`, principalmente usado para upload de arquivos.
3. **Bcrypt:** Uma biblioteca de criptografia usada para proteger senhas. Ela aplica um hash às senhas antes de armazená-las no banco de dados, aumentando a segurança.
4. **Dotenv:** Uma biblioteca para carregar variáveis de ambiente a partir de um arquivo `.env`. É útil para manter informações sensíveis (como chaves de API) fora do código fonte.
5. **Jsonwebtoken:** Usado para criar e verificar tokens de autenticação em aplicações web. É comumente usado para autenticação de API e controle de acesso.
6. **MongoDB:** Um banco de dados NoSQL orientado a documentos. É altamente escalável e usado frequentemente em aplicações web modernas.
7. **Zod:** Uma biblioteca para validação de dados em TypeScript. É usada para definir esquemas de validação e garantir que os dados estejam corretos antes de serem processados pela aplicação.
8. **Tsup:** Uma ferramenta de compilação TypeScript para JavaScript moderno. Facilita a compilação de código TypeScript em JavaScript.
9. **TSX:** Uma extensão de arquivo usada para arquivos TypeScript com sintaxe de JSX. JSX é uma extensão de sintaxe para JavaScript, comumente associada ao React para criar interfaces de usuário.
10. **TypeScript:** Um superset de JavaScript que adiciona tipos estáticos ao código. Ele é frequentemente usado em projetos grandes para melhorar a segurança e a manutenibilidade do código.

Estrutura


Estrutura Geral
Analisador de Arquivos
Front-End
Back-End
Banco de Dados
Made with Whimsical


Envia arquivo
Envia relatório
Envia arquivo
Envia status
Salva informações
Envia status
mongo DB
Salva arquivos
Made with Whimsical