Alunos: Ellen Patricia, José Joaquim, Thiago Barbosa

Justificativa Técnica: Escalabilidade e Tolerância a Falhas

Escalabilidade:

A arquitetura proposta foi desenvolvida para ser altamente escalável, permitindo o crescimento da aplicação sem perda de desempenho.

1. **Uso de Armazenamento em Nuvem (Amazon S3):**
   O Amazon S3 é um serviço de armazenamento distribuído e gerenciado, que se adapta automaticamente ao aumento da demanda.

2. **Arquitetura Desacoplada:**
   O projeto é dividido em três camadas principais:
   - **Frontend**: responsável pela interface com o usuário, utilizando HTML, CSS com Bootstrap e JS
   - **Backend**: processa as requisições e gera URLs seguras para upload e download.
   - **Armazenamento**: guarda os arquivos enviados.

   Essa separação permite que cada componente seja escalado de forma independente, conforme a necessidade.

3. **Backend Stateless:**
   O backend não armazena dados localmente, apenas repassa e gerencia o acesso ao S3.
   Isso permite criar múltiplas instâncias do servidor Flask sem risco de inconsistência, facilitando a escalabilidade horizontal (adição de novas instâncias).

Tolerância a Falhas

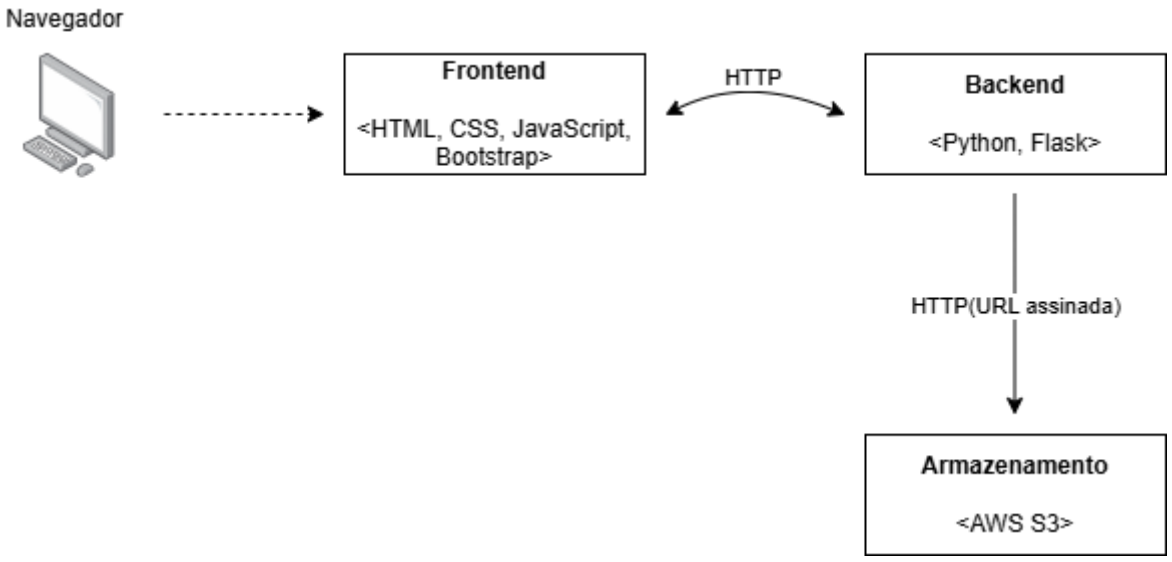
A solução também foi planejada para ser resiliente a falhas, garantindo disponibilidade e integridade dos dados mesmo diante de problemas.

1. **Alta Disponibilidade do S3:**
   O Amazon S3 replica automaticamente os arquivos em múltiplas zonas de disponibilidade dentro da mesma região da AWS. Caso uma zona apresente falha, os dados permanecem acessíveis a partir de outras réplicas.

2. **URLs Presignadas para Download e Upload:**
   O backend Flask gera URLs temporárias que permitem o envio e o download direto dos arquivos do S3, sem precisar passar todo o tráfego pelo servidor. Isso reduz a sobrecarga e evita indisponibilidade caso o backend esteja fora do ar momentaneamente.

3. **Isolamento entre Componentes:**
   O armazenamento em nuvem é independente do backend. Assim, mesmo que o servidor Flask apresente falhas, os arquivos e dados permanecem intactos e acessíveis.

Diagrama



Legenda:

Frontend: Interface web simples, o JavaScript faz requisições de POST para fazer o Upload das imagens e dois GET para listagem da imagens e download das imagens

Backend: O Backend usa o SDK da AWS (Boto3) para listar, fazer upload e gerar URLs assinadas para os arquivos no S3.

Armazenamento: Após o Backend gerar e retornar uma URL assinada, o Frontend usa essa URL para baixar o arquivo *diretamente* do S3.