Documento de Visão do Projeto SnapGram

Participante: Everton Hian dos Santos Pinheiro

1. Introdução

1.1 Objetivo

O SnapGram é uma plataforma de rede social moderna e intuitiva, desenvolvida para conectar pessoas e permitir que compartilhem momentos, fotos e interações por meio de postagens, curtidas e comentários. O objetivo principal é oferecer uma experiência de usuário leve, segura e escalável, com foco em desempenho e usabilidade.

1.2 Escopo

O projeto abrange:

Frontend: Interface gráfica web responsiva e interativa, construída com HTML, CSS e JavaScript.

Backend: Sistema robusto para processamento de solicitações, autenticação e gerenciamento de dados.

API RESTful: Conjunto de endpoints para integração com o frontend e possíveis aplicações externas.

2. Descrição Geral

2.1 Problema

- As redes sociais existentes frequentemente apresentam:
- Interfaces pesadas e pouco responsivas.
- Falta de transparência no uso de dados dos usuários.
- Dificuldade de uso em dispositivos com recursos limitados.

2.2 Solução

- O SnapGram resolve esses problemas ao:
- Oferecer uma interface leve e acessível, com foco em dispositivos móveis.
- Utilizar tecnologias modernas e escaláveis para garantir desempenho e segurança.
- Implementar autenticação robusta e criptografia para proteger os dados dos usuários.

3. Atores

Ator	Interação
Usuários Finais	Pessoas que utilizam a plataforma para criar contas, fazer postagens e interagir.
Administrador	Responsável por gerenciar usuários, monitorar atividades e moderar conteúdos.
API Consumers	Aplicações externas que consomem a API para integrar funcionalidades

4. Requisitos

4.1 Requisitos Funcionais

Id	Requisito Funcional
RF01	O usuário deve poder criar uma conta com e-mail e senha.
RF02	O sistema deve autenticar os usuários por meio do login.
RF03	O usuário deve poder criar, editar e excluir postagens.
RF04	O usuário deve poder enviar imagens junto às postagens.
RF05	Os usuários devem poder curtir e comentar nas postagens.
RF06	O usuário deve poder editar seu perfil (foto, nome, biografia).
RF07	O administrador deve poder bloquear ou excluir contas de usuários.
RF08	Deve haver endpoints para consultar postagens, criar usuários e autenticar.

4.2 Requisitos Não Funcionais

Id	Requisito Não Funcional
RNF01	O tempo de resposta da API deve ser inferior a 300ms para solicitações comuns.
RNF02	Todas as comunicações devem ser criptografadas com HTTPS.
RNF03	Senhas devem ser armazenadas de forma segura.
RNF04	A interface deve ser intuitiva e responsiva para dispositivos móveis.
RNF05	O sistema deve suportar 10.000 usuários simultâneos no início, com possibilidade de expansão.

5. Tecnologias Envolvidas

5.1 Backend

- **Linguagem:** Python
- **Framework**: Flask
- **Banco de Dados:** SQLite (inicialmente), com migração futura para PostgreSQL.
- **Autenticação**: Flask-Login e Flask-JWT-Extended.
- **API**: Flask-RESTful
- **Outras Bibliotecas:**
    - Flask-SQLAlchemy (ORM para banco de dados).
    - Flask-Bcrypt (criptografia de senhas).
    - Flask-Migrate (gerenciamento de migrações).
    - Flask-WTF (validação de formulários).

5.2 Frontend

- HTML/CSS: Estrutura e estilização básica.
- JavaScript: Funcionalidades dinâmicas e interações com a API.

6. Modelos de Casos de Uso

6.1 Caso de Uso: Criar Postagem

Usuário:

Fluxo principal do usuário:

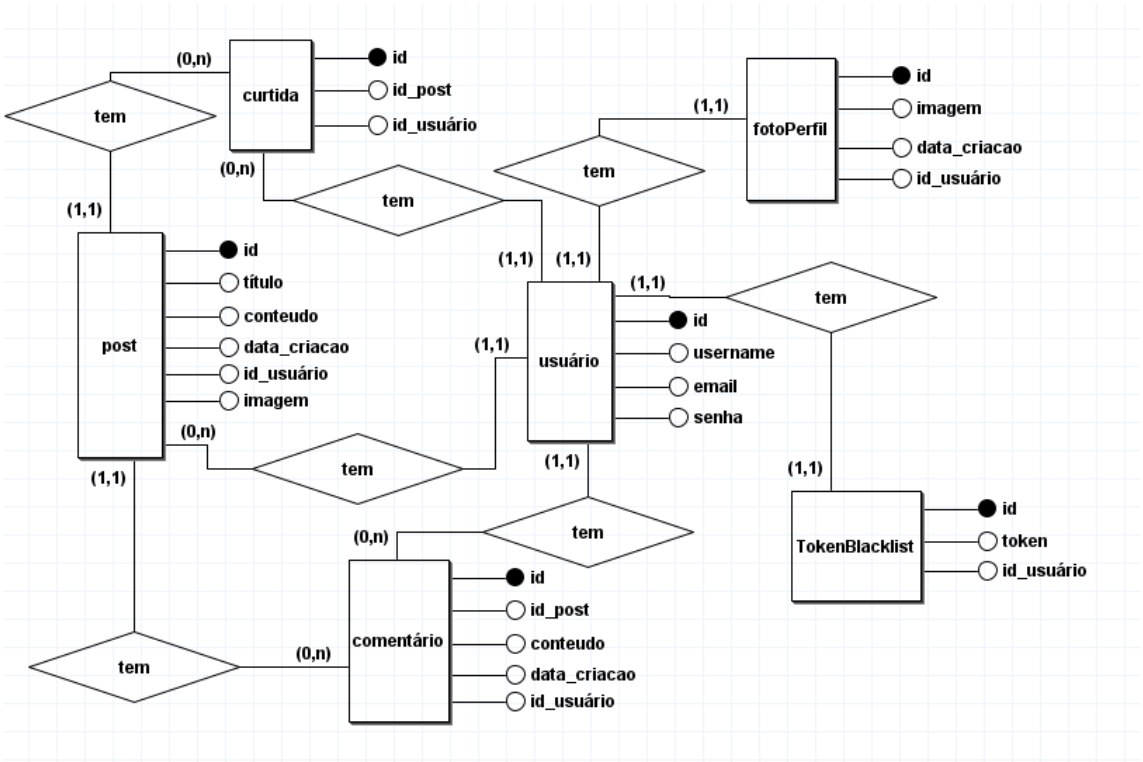
1. O usuário acessa sua conta.
2. Clica em "Nova Postagem".
3. Escreve o conteúdo e faz upload de uma imagem.
4. Salva a postagem.
5. A postagem é exibida no feed.

6.2 Caso de Uso: Curtir Postagem

Fluxo principal do usuário:

1. O usuário acessa sua conta e navega até o feed.
2. Clica no botão de "curtir" em uma postagem.
3. O número de curtidas é incrementado.

7. Estrutura do Banco de Dados



8. Protótipo Inicial

8.1 Wireframes

Página de Login

Página de Feed

Página de Perfil