**Documentação do Projeto de Blog Pessoal**

**1. Visão Geral do Projeto**

**Nome do Projeto:** Blog Pessoal  
**Descrição:** Este projeto consiste na criação de um blog pessoal onde o usuário pode fazer login, criar postagens contendo texto e multimídia, realizar operações CRUD de postagens, além de integrar o blog com redes sociais como Facebook e Instagram. A aplicação será uma plataforma full-stack, incluindo backend e frontend, com armazenamento de dados em um banco de dados. O projeto também incluirá a utilização de Docker para deploy, serviços de nuvem na AWS e versionamento de código.

**Objetivos:**

* Desenvolver uma aplicação web completa para gerenciamento de postagens de blog.
* Implementar uma interface de administração para facilitar o gerenciamento das postagens.
* Permitir login e cadastro de usuários via username e senha.
* Integrar o blog com redes sociais para compartilhamento de postagens.
* Utilizar boas práticas de desenvolvimento de software, incluindo versionamento de código, deploy automatizado e documentação completa.
* Implementar segurança na aplicação usando JWT.

**2. Escopo do Projeto**

**Funcionalidades Principais:**

1. **Cadastro e Login de Usuário:**
   * Cadastro de usuários via username.
   * Login utilizando username e senha.
   * Recuperação de senha.
2. **Gestão de Postagens:**
   * Criação de postagens com título, subtítulo, texto e imagem.
   * Edição de postagens existentes.
   * Remoção de postagens.
   * Listagem de postagens no blog.
3. **Interface de Administração:**
   * Dashboard para gerenciar postagens.
   * Formulário para criação/edição de postagens.
   * Listagem de postagens para edição/removal.
4. **Armazenamento de Dados:**
   * Banco de dados para armazenar usuários e postagens.
5. **Integração com Redes Sociais:**
   * Compartilhamento de postagens no Facebook e Instagram.

**Funcionalidades Futuras:**

* Implementação de comentários nas postagens.
* Suporte para tags e categorias nas postagens.
* Análises e métricas de acesso às postagens.

**3. Requisitos do Sistema**

**Requisitos Funcionais:**

1. **Cadastro e Login de Usuário:**
   * O sistema deve permitir que o usuário se cadastre utilizando seu username.
   * O sistema deve permitir que o usuário faça login utilizando seu username e senha.
   * O sistema deve permitir que o usuário recupere sua senha via e-mail.
2. **Gestão de Postagens:**
   * O sistema deve permitir que o usuário crie, edite e remova postagens.
   * O sistema deve exibir uma lista de todas as postagens no blog.
   * O sistema deve permitir o upload de imagens para as postagens.
3. **Interface de Administração:**
   * O sistema deve fornecer uma interface de administração para gerenciar postagens.
   * A interface deve permitir a criação, edição e remoção de postagens.
   * A interface deve exibir uma lista de todas as postagens com opções de edição/removal.
4. **Armazenamento de Dados:**
   * O sistema deve armazenar informações de usuários e postagens em um banco de dados relacional.
5. **Integração com Redes Sociais:**
   * O sistema deve permitir o compartilhamento de postagens no Facebook e Instagram.

**Requisitos Não Funcionais:**

1. **Desempenho:**
   * O sistema deve ser capaz de lidar com um número razoável de acessos simultâneos.
2. **Usabilidade:**
   * A interface deve ser intuitiva e fácil de usar.
3. **Segurança:**
   * O sistema deve proteger os dados dos usuários, especialmente senhas.
   * O sistema deve seguir as melhores práticas de segurança na autenticação e autorização.
   * Implementação de segurança com JWT para autenticação de usuários.
4. **Escalabilidade:**
   * O sistema deve ser escalável para lidar com o aumento de usuários e postagens.

**4. Arquitetura do Sistema**

**Tecnologias Utilizadas:**

* **Backend:** Spring Boot
* **Frontend:** Angular
* **Banco de Dados:** MySQL
* **Deploy:** Docker, AWS
* **Versionamento:** Git
* **Integração com Redes Sociais:** APIs do Facebook e Instagram

**Diagrama de Arquitetura:** (TBD - Aqui você pode criar um diagrama visual utilizando ferramentas como Lucidchart ou Draw.io)

**5. Modelo de Dados**

**Diagrama ER:** (TBD - Diagrama de Entidade-Relacionamento para o banco de dados)

**Principais Entidades:**

1. **Usuario**
   * id (UUID)
   * nome (String)
   * username (String)
   * password (String)
   * dataCadastro (LocalDateTime)
2. **Post**
   * id (UUID)
   * titulo (String)
   * subtitulo (String)
   * texto (String)
   * urlImagem (String)
   * dataHoraCriacao (LocalDateTime)
   * atualizacao (LocalDateTime)
   * categoria (Categoria)
   * usuario (Usuario)
3. **Categoria**
   * DIREITO
   * TECNOLOGIA
   * MUSICA
   * FILMES\_E\_SERIES
   * LIVROS
   * OPINIOES

**6. Casos de Uso**

**Caso de Uso: Cadastro de Usuário**

* **Ator Principal:** Usuário
* **Pré-condição:** Nenhuma
* **Pós-condição:** O usuário está cadastrado no sistema.
* **Fluxo Principal:**
  1. O usuário acessa a página de cadastro.
  2. O usuário preenche o formulário de cadastro.
  3. O usuário envia o formulário.
  4. O sistema valida as informações e cria um novo usuário.

**Caso de Uso: Login de Usuário**

* **Ator Principal:** Usuário
* **Pré-condição:** O usuário deve estar cadastrado.
* **Pós-condição:** O usuário está logado no sistema.
* **Fluxo Principal:**
  1. O usuário acessa a página de login.
  2. O usuário preenche o formulário de login.
  3. O usuário envia o formulário.
  4. O sistema valida as credenciais e autentica o usuário.

**Caso de Uso: Criação de Postagem**

* **Ator Principal:** Usuário Admin
* **Pré-condição:** O usuário deve estar logado.
* **Pós-condição:** A postagem é criada e armazenada no banco de dados.
* **Fluxo Principal:**
  1. O usuário acessa a interface de administração.
  2. O usuário preenche o formulário de criação de postagem.
  3. O usuário envia o formulário.
  4. O sistema valida as informações e cria uma nova postagem.

**7. Diagrama de Classes UML**

(TBD - Diagrama de classes para representar as entidades e suas relações)

**8. Plano de Desenvolvimento**

**Fases do Desenvolvimento:**

1. **Fase de Planejamento:**
   * Definição dos requisitos.
   * Criação da documentação inicial.
2. **Fase de Design:**
   * Criação dos diagramas de arquitetura e modelos de dados.
   * Definição da estrutura de diretórios do projeto.
3. **Fase de Implementação:**
   * Desenvolvimento do backend com Spring Boot.
   * Desenvolvimento do frontend com Angular.
   * Implementação da interface de administração.
   * Integração com o banco de dados.
   * Implementação da autenticação e autorização usando JWT.
4. **Fase de Testes:**
   * Testes unitários e de integração.
   * Testes de usabilidade e segurança.
5. **Fase de Deploy:**
   * Configuração do Docker.
   * Deploy na AWS.
   * Integração contínua e entrega contínua (CI/CD).
6. **Fase de Manutenção:**
   * Correção de bugs.
   * Adição de novas funcionalidades.
   * Melhorias contínuas.

**9. Considerações Finais**

Este documento serve como guia para o desenvolvimento do Blog Pessoal, oferecendo uma visão clara do que deve ser feito e como deve ser feito. A documentação deve ser atualizada conforme o projeto evolui, garantindo que todos os detalhes e mudanças sejam registrados.

**Apêndices**

**Ferramentas Recomendadas:**

* **IDE:** IntelliJ IDEA ou Eclipse para desenvolvimento Java.
* **Editor de Código:** Visual Studio Code para desenvolvimento frontend.
* **Banco de Dados:** MySQL Workbench para gerenciar o banco de dados.
* **Controle de Versão:** GitHub ou GitLab.
* **Diagramação:** Lucidchart ou Draw.io para criar diagramas (preferencialmente gratuitas).

**Agradecimentos**

Agradecimentos a todos os recursos e ferramentas de código aberto que facilitam o desenvolvimento de software de alta qualidade.

**Contato:**

* **Nome:** Daniel Azevedo de Oliveira Maia
* **Telefone:** (83) 98888-7116