**Documentação do Sistema**

**1. Introdução**

* **Nome do aluno(ª):** Ramon Souza de Santana
* **Matrícula:**
* **Nome do projeto:** Sistema de Checklist de Segurança Abertura e Fechamento
* **Objetivo:** O sistema gerencia a execução de checklists de segurança para a abertura e o fechamento de uma instalação. Ele resolve o problema da falta de um registro digital e geolocalizado, garantindo que os procedimentos sejam seguidos e auditados de forma centralizada e segura.
* **Público-alvo:** Líderes de equipe, fiscais, supervisores e vigilantes responsáveis pelos procedimentos de segurança de um local.

**2. Requisitos do Sistema**

**2.1 Funcionais**

* [x] **Autenticação:** O sistema deve permitir autenticação de usuários com login e senha.
* [x] **CRUD Básico:** O sistema deve realizar operações de Cadastro (novos usuários, novos registros de checklist) e Listagem (registros de abertura e fechamento).
* [x] **API Externa:** O sistema deve se conectar à API de Geolocalização do Navegador para obter as coordenadas exatas de onde o checklist foi preenchido.
* [x] **Banco de Dados:** O sistema deve armazenar todas as informações de usuários e checklists em um banco de dados relacional.

**2.2 Não Funcionais**

* **Desenvolvido em:** Python
* **Banco de dados utilizado:** SQLite
* **Frameworks/Bibliotecas:** Flask (backend) e Bootstrap 5 (frontend).
* **Usabilidade:** O sistema simples, fluido e responsivo

**3. Funcionalidades do Sistema**

* **Cadastro de Usuários:** Permite a criação de novas contas de usuário para acesso ao sistema, com login e senha criptografada.
* **Login de Usuário:** Garante que apenas usuários autenticados possam acessar as funcionalidades principais do sistema, através de um sistema de sessão.
* **Gerenciamento de Checklists:** Permite o cadastro de checklists de Abertura e Fechamento, salvando informações como responsável, fiscal, itens verificados e data/hora.
* **Captura de Coordenadas Geográficas:** Utiliza a API de Geolocalização do navegador para registrar automaticamente a latitude e longitude no momento em que um checklist é salvo, garantindo a veracidade do local do registro.
* **Listagem de Registros:** Exibe em tabelas separadas todos os checklists de abertura e fechamento já realizados, permitindo uma visualização rápida do histórico de operações.

**4. Tecnologias Utilizadas**

* **Linguagem de programação:** Python 3
* **Banco de dados:** SQLite 3
* **Frameworks:** Flask
* **Frontend:** HTML5, CSS3, JavaScript e Bootstrap 5
* **API utilizada:** Browser Geolocation API (API de Geolocalização do Navegador)
* **Outras ferramentas:** Visual Studio Code (Editor de Código), DB Browser for SQLite (Gerenciador de Banco de Dados).

**5. Instruções de Execução**

1. **Pré-requisito:** Certifique-se de ter o Python e o pip instalados em sua máquina.
2. **Download do Projeto:** Faça o download ou clone todos os arquivos do projeto (app.py, schema.sql, as pastas templates e static) para um diretório local.
3. **Instalar Dependências:** Abra um terminal na pasta do projeto e instale o Flask com o comando: pip install Flask
4. **Criar o Banco de Dados:** Utilize o programa **DB Browser for SQLite** para criar um novo banco de dados chamado database.db na raiz do projeto. Em seguida, use a aba "Executar SQL" para rodar o script do arquivo schema.sql e clique em "Gravar alterações".
5. **Executar o Projeto:** No mesmo terminal, execute o servidor de desenvolvimento com o comando: python app.py
6. **Acessar o Sistema:** Abra seu navegador de internet e acesse o seguinte endereço: http://127.0.0.1:5000

**6. Conclusão**

* **Aprendizado:** O desenvolvimento deste sistema proporcionou um aprendizado prático e completo sobre a criação de uma aplicação web full-stack. Os principais conhecimentos adquiridos incluem a estruturação de um projeto com Flask, o gerenciamento de rotas, a interação com um banco de dados SQL (CRUD), a implementação de um sistema seguro de autenticação com hashing de senhas e sessões, a integração de APIs do lado do cliente (Geolocalização) e a criação de uma interface de usuário moderna e responsiva com Bootstrap.
* **Melhorias Futuras:**
  + **CRUD Completo:** Implementar as funcionalidades de **Editar** e **Excluir** os registros de checklist, talvez com permissões de acesso (ex: apenas um "administrador" pode excluir).
  + **Relatórios:** Criar uma página de relatórios que permita filtrar os checklists por data, por usuário ou por tipo (abertura/fechamento).
  + **Dashboard Visual:** Desenvolver um painel inicial com gráficos mostrando estatísticas, como o número de registros por dia ou os horários mais comuns.
  + **Assinatura Digital:** Adicionar um campo para o usuário desenhar uma assinatura digital no momento do registro, para aumentar a validade do documento.