**Sistema de Botão de Pânico - IFPB**

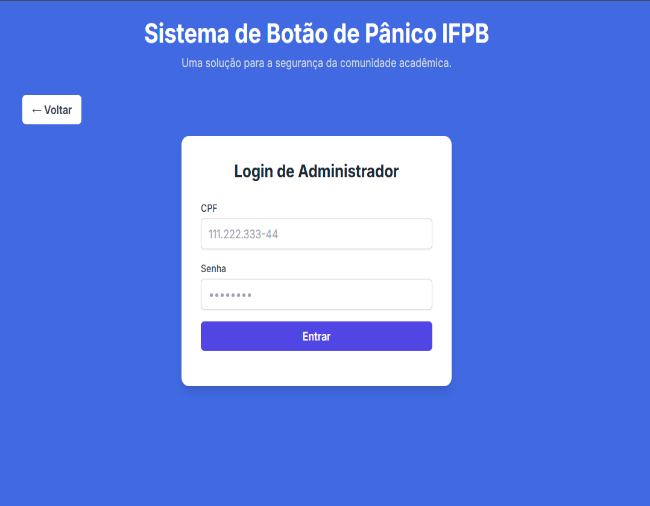
Este repositório contém o código-fonte e a documentação do **Sistema de Botão de Pânico**, um projeto acadêmico desenvolvido por **Alexandre Freitas de Lima Pacheco** e **Robson Luan Fernandes Pereira** como requisito para a disciplina de Padrões de Projetos do curso de Engenharia de Computação do IFPB - Câmpus Campina Grande.

**🚀 Demonstração ao Vivo**

**Acesse a aplicação em funcionamento através do link abaixo:**

**https://github.com/Alexandre-Pacheco/Sistema-de-Bot-o-de-P-nico-IFPB**

**✨ Visão Geral da Aplicação**

**🎯 Sobre o Projeto**

O Sistema de Botão de Pânico é uma solução web projetada para aumentar a segurança da comunidade acadêmica (alunos, professores e técnicos) dentro das instalações do campus. Ele oferece um meio rápido e eficaz de solicitar ajuda em situações de emergência, enviando um alerta geolocalizado em tempo real para uma central de monitoramento administrada pela equipe de segurança.

**📋 Funcionalidades Principais**

* **Dois Perfis de Acesso:**
* **Área do Usuário:** Interface limpa para alunos e professores acionarem o botão de pânico.
* **Área do Administrador:** Painel protegido por login para a equipe de segurança gerenciar alertas e usuários.
* **Acionamento Seguro:** O botão de pânico requer que o usuário pressione por 3 segundos para evitar disparos acidentais.
* **Central de Monitoramento:** Visualização de alertas ativos em tempo real, com informações do usuário e localização.
* **Gestão de Incidentes:** A equipe de segurança pode gerenciar todo o ciclo de vida de um alerta (Pendente → Em Atendimento → Resolvido/Falso Alarme).
* **Cadastro e Gestão de Usuários:** Administradores podem cadastrar e excluir usuários (alunos/professores) do sistema.
* **Geração de Relatórios:** Ferramenta para criar relatórios estatísticos sobre os incidentes ocorridos.

**🛠️ Arquitetura e Padrões de Projeto**

Este projeto foi construído como uma aplicação de página única (SPA) utilizando **HTML5, Tailwind CSS e JavaScript puro**. A arquitetura foi pensada para ser modular, resiliente e de fácil manutenção, aplicando **9 padrões de projeto** fundamentais para resolver desafios específicos:

|  |  |  |
| --- | --- | --- |
| **Padrão de Projeto** | **Onde foi Utilizado** | **Propósito** |
| **Singleton** | No ConfigurationManager e na MonitoringCentral. | Garantir uma instância única para configurações e para a central de alertas. |
| **Factory Method** | Na UserFactory, para criar objetos de Aluno e Professor. | Desacoplar a criação de objetos de usuários do sistema principal. |
| **Observer** | Para notificar a MonitoringCentral instantaneamente quando um novo alerta é acionado. | Permitir a comunicação em tempo real entre o usuário e a central. |
| **State** | Para gerenciar o ciclo de vida de um Incidente (Pendente, Em Atendimento, Resolvido, etc.). | Controlar o comportamento de um objeto conforme seu estado interno muda. |
| **Strategy** | No LocationProvider, permitindo alternar entre a coleta de localização por GPS ou Wi-Fi. | Tornar algoritmos de localização intercambiáveis e independentes. |
| **Command** | Para encapsular a ação de TriggerAlertCommand (acionar o pânico). | Transformar uma solicitação em um objeto, permitindo mais flexibilidade. |
| **Facade** | Na PanicSystemFacade, que simplifica a comunicação entre a interface do usuário e os subsistemas complexos. | Prover uma interface simplificada para um conjunto complexo de funcionalidades. |
| **Adapter** | Simulado no AdminAuthService para representar a integração com um sistema de autenticação legado. | Permitir que interfaces incompatíveis trabalhem juntas. |
| **Builder** | No IncidentReportBuilder, para construir relatórios complexos de forma flexível e passo a passo. | Separar a construção de um objeto complexo de sua representação. |

**📂 Estrutura do Código**

Para facilitar a demonstração e o compartilhamento, toda a aplicação está contida em um único arquivo index.html. Dentro dele, a estrutura está organizada da seguinte forma:

1. **<head>:** Importação de fontes e do Tailwind CSS, além de estilos básicos.
2. **<body>:** Estrutura HTML de todas as telas (Seleção de Perfil, Usuário, Login de Admin, Painel de Admin).
3. **<script>:** Todo o código JavaScript, separado em seções claras:

* Gerenciamento de Views (telas).
* Dados Globais e Função de Log.
* Implementação de cada um dos 9 Padrões de Projeto, com comentários explicativos.
* Lógica da Interface do Usuário (UI) e associação de eventos.

**👨‍💻 Autores**

* **Alexandre Freitas de Lima Pacheco**
* **Robson Luan Fernandes Pereira**

**📄 Licença**

Este projeto está sob a licença MIT. Veja o arquivo LICENSE para mais detalhes.