

**DOCUMENTAÇÃO DE SOFTWARE**

(Projeto Ativements)

**Histórico de Revisões**

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Versão | Descrição | Autor |
| 30/08/2024 | 1.0 | Elaboração para análise da primeira versão do documento. | Kauã de Melo Rodrigues |

**Sumário**

[1. Visão Geral 5](#_30j0zll)

[Objetivo 5](#_1fob9te)

[Escopo 5](#_3znysh7)

[2. Introdução 5](#_2et92p0)

[Contexto 5](#_tyjcwt)

[3. Requisitos 6](#_3dy6vkm)

[Requisitos Funcionais 6](#_1t3h5sf)

[Requisitos Não Funcionais 6](#_4d34og8)

[4. Regras de Negócio 6](#_2s8eyo1)

[5. Restrições 6](#_17dp8vu)

[6. Arquitetura 6](#_3rdcrjn)

[Visão Geral da Arquitetura 6](#_26in1rg)

[Diagramas de Arquitetura 6](#_lnxbz9)

[Diagrama de caso de uso 6](#_35nkun2)

[Diagrama de classe 6](#_1ksv4uv)

[Modelagem do banco 6](#_44sinio)

[7. Design do Sistema 7](#_2jxsxqh)

[Padrões de Design 7](#_z337ya)

[Estrutura do Código 7](#_3j2qqm3)

[8. Detalhamento dos Módulos 7](#_1y810tw)

[Módulo x 7](#_4i7ojhp)

[9. Configuração e Instalação 7](#_2xcytpi)

[Pré-requisitos 7](#_1ci93xb)

[Passos de Instalação 7](#_3whwml4)

[10. Uso 8](#_2bn6wsx)

[Guia do Usuário 8](#_qsh70q)

[Exemplos de Uso 8](#_3as4poj)

[11. Testes 8](#_1pxezwc)

[Estratégias de Testes 8](#_49x2ik5)

[Casos de Teste 8](#_2p2csry)

[Resultados dos Testes 8](#_147n2zr)

[12. Manutenção 8](#_3o7alnk)

[Plano de Manutenção 8](#_23ckvvd)

[Registro de Alterações 8](#_ihv636)

[13. Referências 9](#_32hioqz)

1. Visão Geral

Objetivo

O Ativements é um sistema que tem como objetivo gerenciar ativos da escola **“S Informática”** proporcionando um controle eficaz dos equipamentos e otimizando os recursos da escola.

Escopo

O sistema possibilitará:

* Que os usuários se cadastrem no sistema
* Que os usuários se autentiquem no sistema
* Que os usuários acessem a lista de todos os ativos
* Que os usuários cadastrem um ativo
* Que os usuários alterem dados de um ativo
* Que os usuários excluam um ativo

1. Introdução

Contexto

A S Informática possui um grande número de equipamentos de TI, incluindo computadores, notebooks, impressoras, projetores e outros dispositivos , utilizados em seus cursos. O controle desses ativos se torna um desafio, com dificuldades no:

Rastreamento: Dificuldade em localizar e identificar cada ativo, gerando perdas e desperdícios.

Após a implementação do sistema Ativements, será possível:

* **Organizar e catalogar:** Registrar todos os ativos como informações detalhadas (modelo, serial, data de aquisição, etc.).
* **Controlar localização:** Utilizar etiquetas de identificação e softwares de rastreamento para localização precisa.

1. Requisitos

**Requisitos Funcionais**

|  |  |
| --- | --- |
| **RF1- Efetuar Login** | Autenticação dos usuários do sistema |
| **RF2 - Cadastrar Ativos** | Cadastrar ativos |
| **RF3 - Cadastrar Usuários** | Cadastro de usuários |
| **RF4 - Listagem Ativos** | Exibição dos ativos cadastrados |
| **RF5 - Atualizar Ativos** | Atualizar informações sobre um ativo |
| **RF6 - Efetuar Logout** | Remover a autenticação do usuário |

**Requisitos Não Funcionais**

|  |  |
| --- | --- |
| **RNF 1- Performance** | O tempo de resposta para cada operação do sistema deve ser inferior a 2 segundos. |
| **RNF 2 -Segurança** | O acesso ao sistema deve ser restrito a usuários autenticados por login |
| **RNF 3 - Usabilidade** | O sistema deve ser intuitivo, simples e ter uma interface amigável |
| **RNF 4 - Disponibilidade** | O sistema deve estar disponível 24 horas por dia, 7 dias por semana, com interrupção para manutenção planejada com antecedência |

1. **Regras de Negócio**

|  |  |
| --- | --- |
| **RN1** | A autenticação deverá ser realizado através do username do usuário do perfil do seu GitHub |
| **RN2** | O cadastros deve ser feito através do usuário deve ser feito através do username do usuário do seu GitHub |
| **RN3** | Se já existir um usuário com username informado, o sistema deve impedir um novo cadastro |
| **RN4** | No cadastro do ativo, deve conter as seguintes informações: nº de identificação do ativo, nome do ativo, status de ativo ou inativo, data e hora do registro, local que ficará o ativo |
| **RN5** | Deverá ficar registrado o usuário que cadastrou o ativo |
| **RN6** | No cadastro de ativos, todos os campos (informações) deverão estar preenchidos |
| **RN7** | O usuário poderá filtrar pelo local |
| **RN8** | O usuário poderá alternar o status do ativo para inativo e de inativo para ativo |
| **RN9** | O usuário, ao atualizar as informações sobre um ativo, o número do ativo não poderá ser alterado |
| **RN10** | Ao cadastrar um ativo caso já exista um local desejado, deverá alocar o ativo a esse local caso ainda não exista esse local, então deve-se cadastrar |
| **RN11** | Ao atualizar um ativo, deverá ser registrado o usuário que fez a ação, bem como a data e hora |
| **RN12** | Ao sair do sistema, ficará registrado a data ea hora |

**Restrições**

|  |  |
| --- | --- |
| **RS1** | Sistema operacional no mínimo Windows 10. |
| **RS2** | O sistema deve ter capacidade de armazenamento flexível |

1. Arquitetura

Visão Geral da Arquitetura

O sistema segue uma arquitetura de 2 camadas:

* **Front End:** Desenvolvido em ReactJS, responsável pela interface do usuário.
* **Back End:** Json Server para simular toda a persistência de dados.

**Resumo:**

* O usuário interage com o frontend (ReactJS)
* O frontend faz requisições para o json server através de uma rota que disponibiliza os recursos.

Diagramas de Arquitetura

Incluir diagramas de arquitetura, como diagramas de caso de uso e diagrama de classe

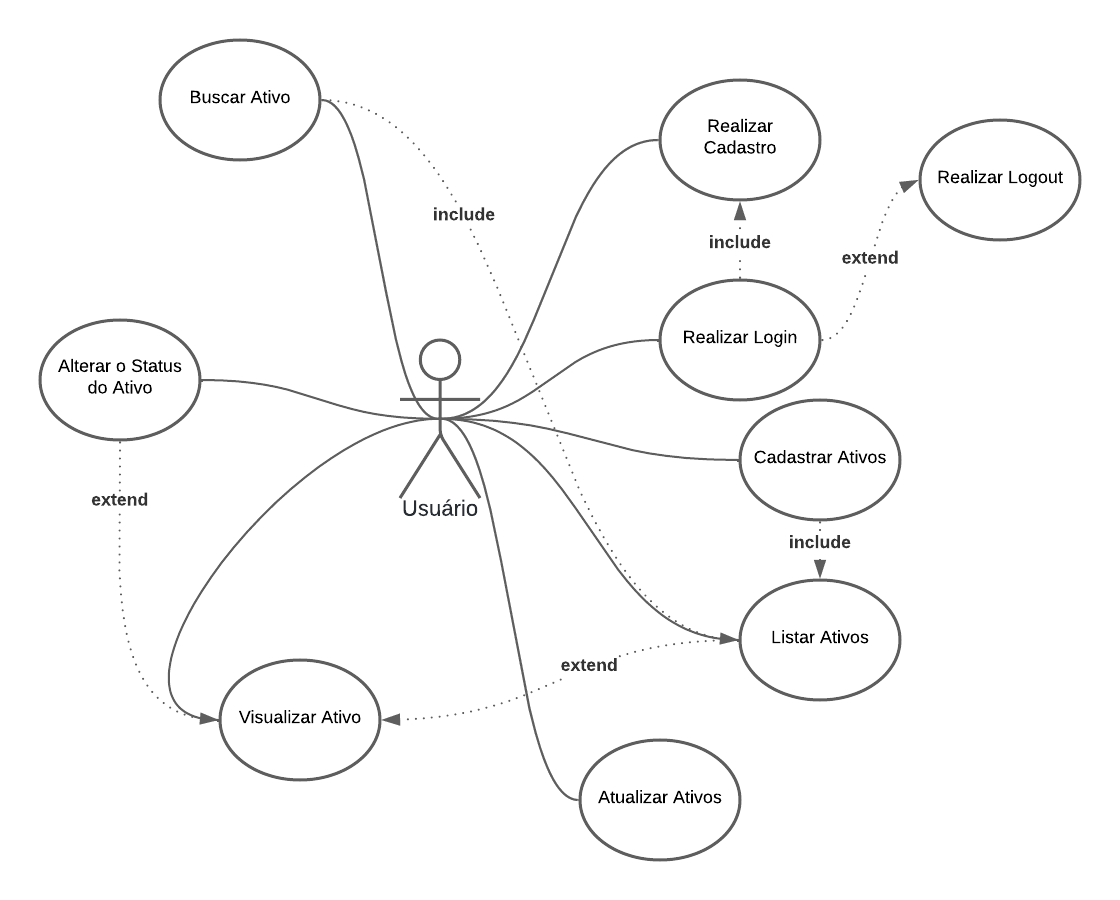
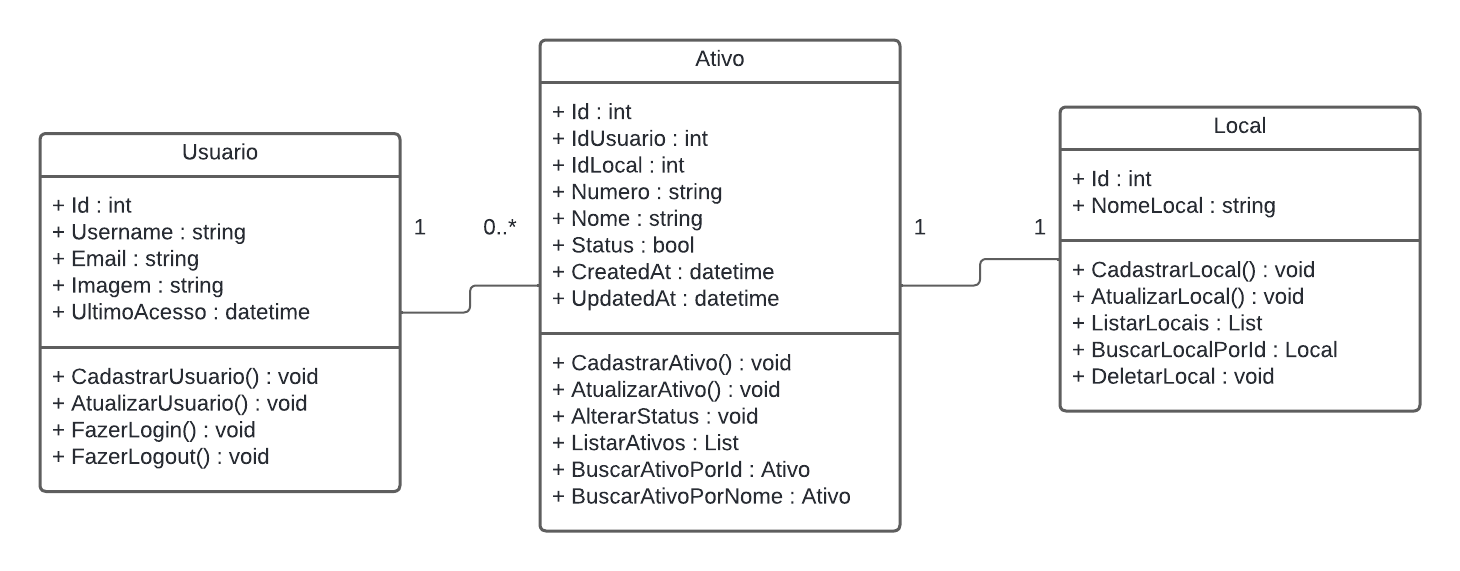
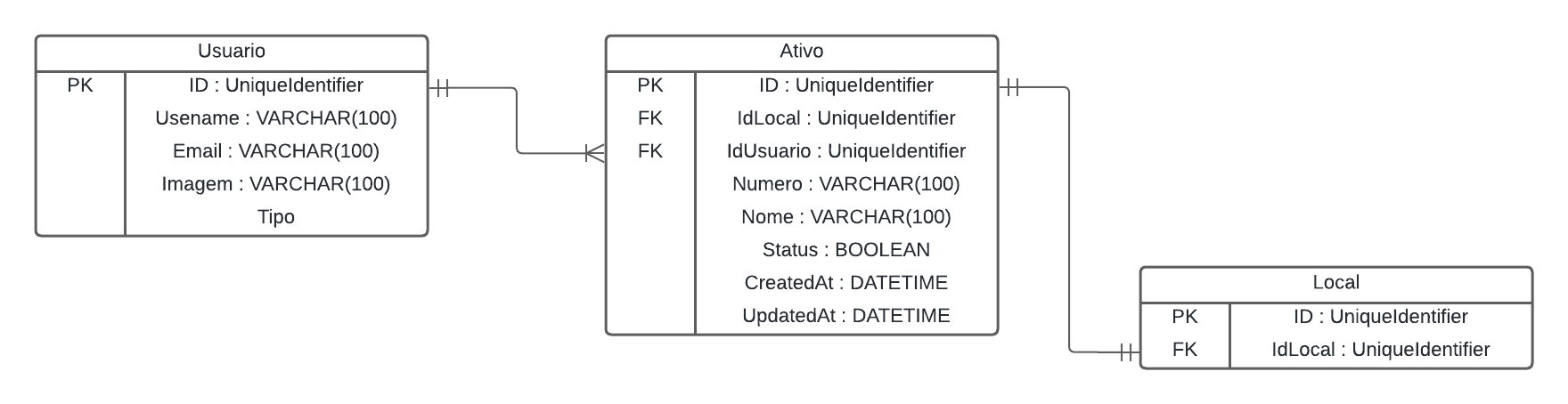
Diagrama de caso de uso

Diagrama de classe



Modelagem do banco



1. Design do Sistema

Padrões de Design

**Component-based Architecture**: Esse tipo de design de software, é baseado na componentização da aplicação, em módulos menores, permitindo uma maior reutilização de código em diversas partes da aplicação, além de tornar o sistema mais eficiente e rápido, esse tipo de arquitetura já é o padrão na criação de aplicações ReactJS.

**JsonServer**: Será utilizado o JsonServer para a criação de uma falsa api, porém os dados serão armazenados em um arquivo json na raiz do projeto, permitindo a criação de uma simulação com requisições e com persistência de dados, com uma estrutura mais simples para o Backend.

Estrutura do Código

Detalhar a estrutura do código, incluindo pacotes, módulos, etc.

1. Detalhamento dos Módulos

Módulo x

Descrever em detalhes o Módulo e suas funcionalidades, interfaces, etc.

1. Configuração e Instalação

Pré-requisitos

* Instalação do Node.js

Passos de Instalação

* Clone o repositório do projeto
* No diretório frontend, executem npm install para instalar dependências
* Execute npm start no diretório do frontend para iniciar a aplicação React
* Acesse <http://localhost:3000> no navegador para utilizar o sistema

1. Uso

Guia do Usuário

Fornecer um guia de uso para os usuários finais.

Exemplos de Uso

Incluir exemplos de uso do software.

1. Testes

Estratégias de Testes

Por enquanto não foi elaborado nenhum caso de teste.

Casos de Teste

Por enquanto não foi elaborado nenhum caso de teste.

Resultados dos Testes

Por enquanto não foi elaborado nenhum caso de teste.

1. Manutenção

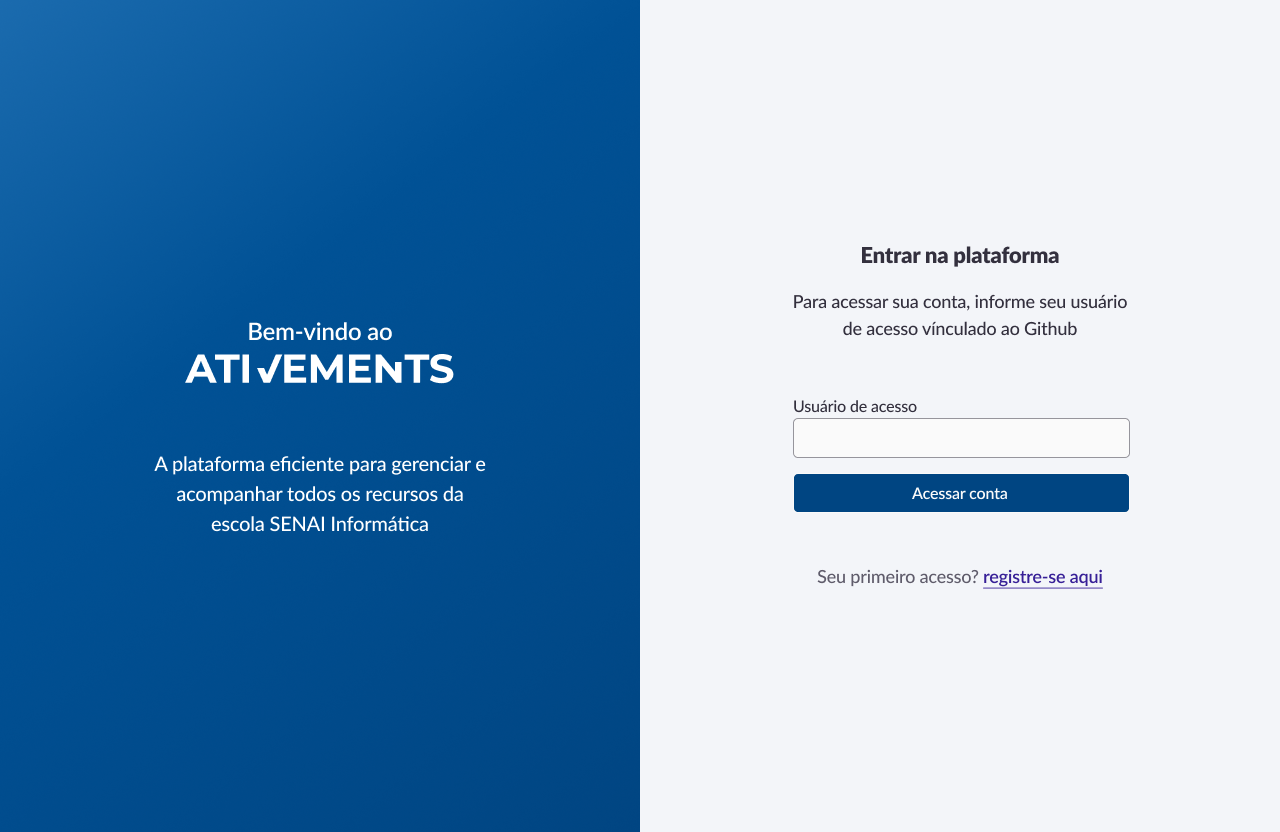
Plano de Manutenção

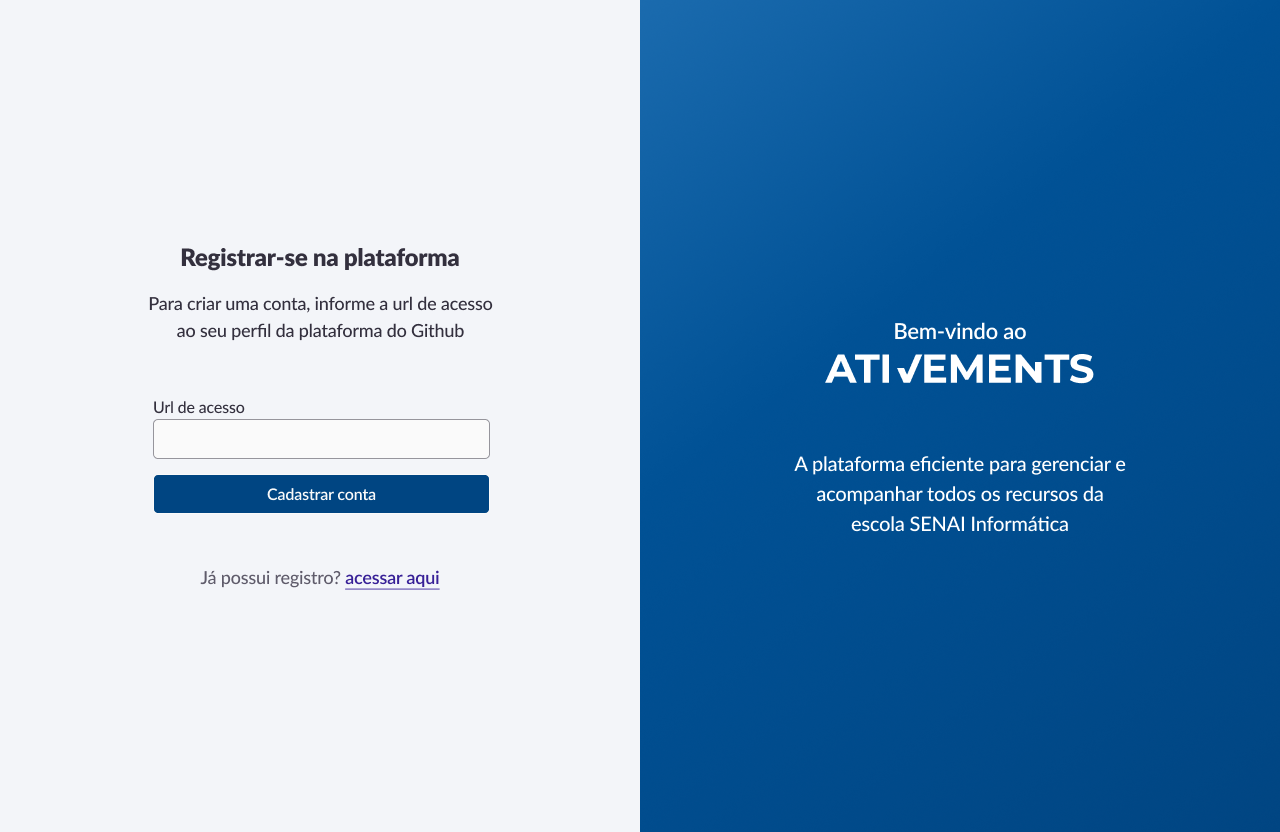
Por enquanto não foi elaborado nenhum plano de manutenção.

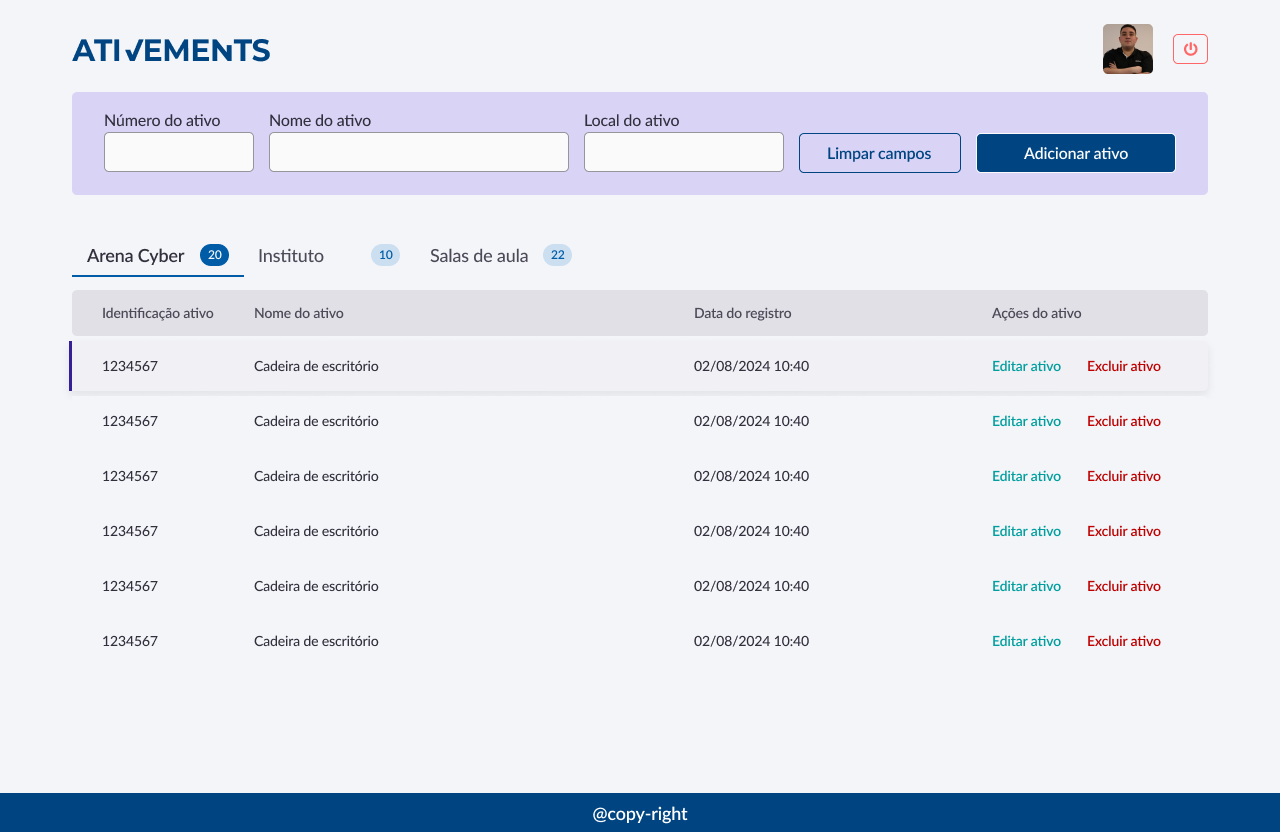
Registro de Alterações

Por enquanto não foi elaborado nenhum plano de alterações.

1. **Protótipo**

****

****

****

1. Referências

Listar todas as referências utilizadas para o desenvolvimento do software, como livros, artigos, sites, etc.