**DOCUMENTAÇÃO DE SOFTWARE**

(Modelo de documentação de software)

**Histórico de Revisões**

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Versão | Descrição | Autor |
| 30/08/2024 | 1.0 | Elaboração para análise da primeira versão do documento. | Eduardo Merbach Brenn |
|  |  |  |  |

**Sumário**

[1. Visão Geral 5](#_Toc173502504)

[Objetivo 5](#_Toc173502505)

[Escopo 5](#_Toc173502506)

[2. Introdução 5](#_Toc173502507)

[Contexto 5](#_Toc173502508)

[3. Requisitos 6](#_Toc173502509)

[Requisitos Funcionais 6](#_Toc173502510)

[Requisitos Não Funcionais 6](#_Toc173502511)

[4. Regras de Negócio 6](#_Toc173502512)

[5. Restrições 6](#_Toc173502513)

[6. Arquitetura 6](#_Toc173502514)

[Visão Geral da Arquitetura 6](#_Toc173502515)

[Diagramas de Arquitetura 6](#_Toc173502516)

[Diagrama de caso de uso 6](#_Toc173502517)

[Diagrama de classe 6](#_Toc173502518)

[Modelagem do banco 6](#_Toc173502519)

[7. Design do Sistema 7](#_Toc173502520)

[Padrões de Design 7](#_Toc173502521)

[Estrutura do Código 7](#_Toc173502522)

[8. Detalhamento dos Módulos 7](#_Toc173502523)

[Módulo x 7](#_Toc173502524)

[9. Configuração e Instalação 7](#_Toc173502525)

[Pré-requisitos 7](#_Toc173502526)

[Passos de Instalação 7](#_Toc173502527)

[10. Uso 8](#_Toc173502528)

[Guia do Usuário 8](#_Toc173502529)

[Exemplos de Uso 8](#_Toc173502530)

[11. Testes 8](#_Toc173502531)

[Estratégias de Testes 8](#_Toc173502532)

[Casos de Teste 8](#_Toc173502533)

[Resultados dos Testes 8](#_Toc173502534)

[12. Manutenção 8](#_Toc173502535)

[Plano de Manutenção 8](#_Toc173502536)

[Registro de Alterações 8](#_Toc173502537)

[13. Protótipo 9](#_Toc173502538)

14. Referências.......................................................10

1. Visão Geral

Objetivo

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

Escopo (Todas as funcionalidades que o aplicativo deverá ter)

O sistema possibilitará que os usuários:

- Se cadastrem no sistema

- Se autentiquem no sistema

- Acessem a lista de todos os ativos

- Cadastrem um ativo

- Alterem os 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, entre 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 com 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

Listar os requisitos funcionais do sistema.

Requisitos Não Funcionais

Listar os requisitos funcionais do sistema.

1. Regras de Negócio

Listar as regras de negócio do sistema.

1. Restrições

Listar as restrições do sistema.

1. Arquitetura

Visão Geral da Arquitetura

O sistema deverá seguir uma estrutura de 2 arquiteturas:

**FrontEnd:** Desenvolvido em ReactJs, responsável pela interface do usuário

**BackEnd:** 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 e 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

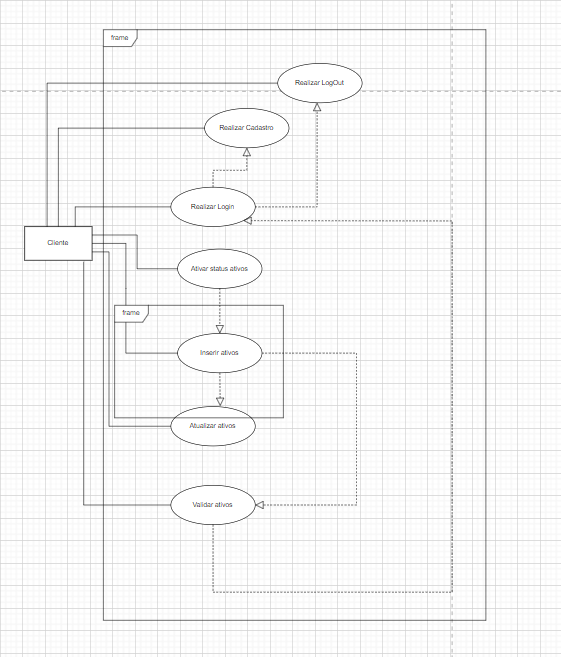
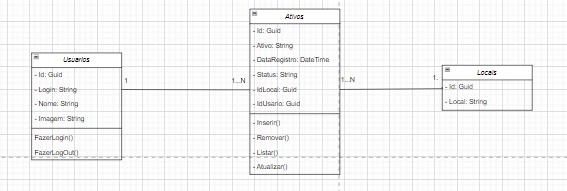
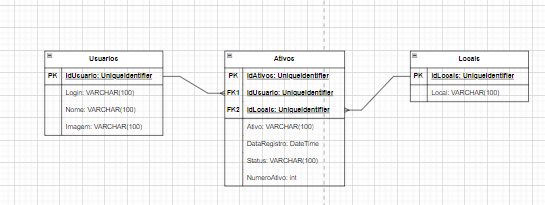


Diagrama de classe



Modelagem do banco

1. Design do Sistema

Padrões de Design

O sistema usa como base para o frontend o **Component based architecture (**Arquitetura baseada em componentes**)** que é um estilo de design de software em que a aplicação é dividida em componentes reutilizáveis e independentes. Cada componente encapsula uma parte específica da funcionalidade da aplicação, incluindo a interface de usuário, lógica de negócios, e dados associados.

No contexto de ReactJs, cada componente é geralmente uma função ou classe que retorna um fragmento da interface do usuário (UI). Esses componentes podem ser combinados para formar páginas e funcionalidades completas da aplicação.

Isso promove a modularidade, facilitando a manutenção, testes, e reuso de código.

Pare o BackEnd, é usado o Json-Server que é uma biblioteca Node.js que simula uma API RESTful a partir de um arquivo JSON

Estrutura do Código

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

1. Detalhamento dos Módulos

Módulo x

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

1. Configuração e Instalação

Pré-requisitos

Node JS

Passos de Instalação

- Clone o repositório do projeto

- No diretório do FrontEnd, execute npm install para instalar as dependências

- Execute npm start no diretório do projeto para iniciar a aplicação React

- Acesse <http://localhost3000> no navegador para utilizar o sistema

1. Uso

Guia do Usuário

Fornecer um guia de uso para os usuários finais.

(Pode ser feito em formato de vídeo)

Exemplos de Uso

Incluir exemplos de uso do software.

1. Testes

Estratégias de Testes

Descrever a estratégia de testes adotada.

Casos de Teste

Por enquanto não foi elaborado nenhum caso de teste

Resultados dos Testes

Por enquanto não existem casos de teste

1. Manutenção

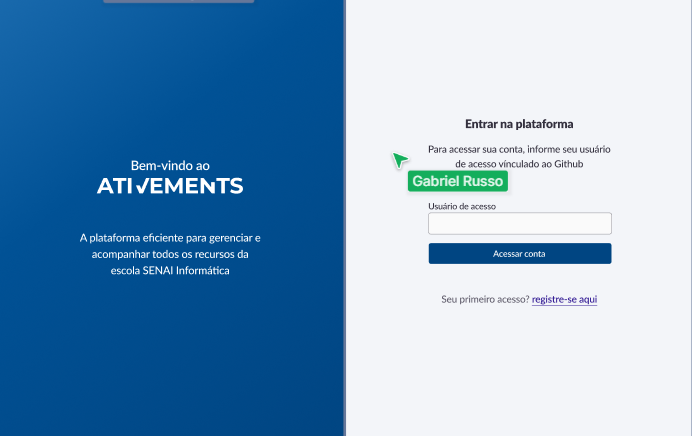
Plano de Manutenção

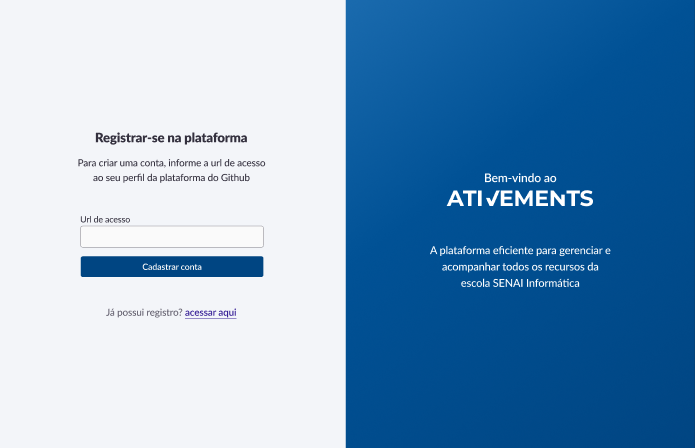
O plano de manutenção ainda será desenvolvido

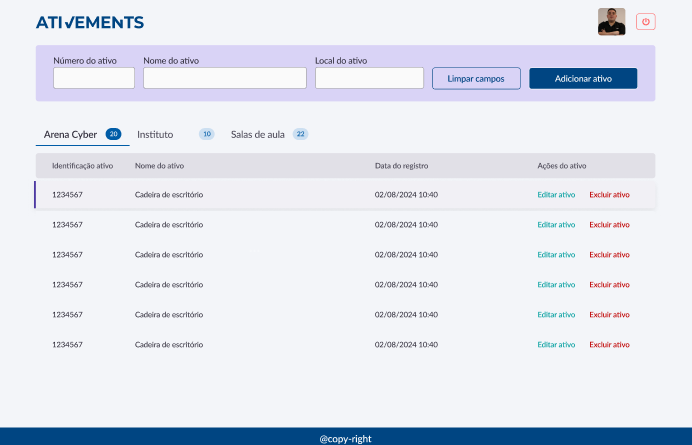
Registro de Alterações

No momento ainda não existem alterações

1. . Protótipo







1. Referências

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