Introdução

Criar um sistema web que permita o cadastro de usuários e a gestão de acessos temporários a determinados recursos. Administradores podem conceder permissões com prazos definidos e revogar acessos quando necessário. O sistema deve garantir segurança, controle de expiração automática e uma interface intuitiva para facilitar a administração dos acessos.

Descrição geral do sistema

Objetivos do sistema

Cria-se documentos, com título, corpo do texto, imagens e pdf, estes podem ser compartilhados com usuários do tipo 'User', liberando acessos de leitura, edição e comentários. Os dados são salvos no LocalStorage para persistência de usuários, documentos e permissões temporárias.

Repositório GitHub

https://github.com/andressateixeira13/gestao-de-acessos.git

Principais funcionalidades

- Manter Usuários
- Manter Documentos
- Role: Admin e User
- Gerenciamento de Acessos
- Modo Dark/Light
- Idiomas (Inglês e Português)

Restrições ou requisitos especiais

- Expiração de Acessos Automático
- Possibilidade de revogar o acesso antes do prazo de expiração
- Possibilidade de estender acesso
- Destaca acessos prestes a vencer

User Init

Admin- Email: admin@teste.com; Senha: 123456

User- Email: user@teste.com; Senha: 123456

Arquitetura do sistema

Tecnologias utilizadas

- Vue.js 3
- Quasar Framework
- TypeScript
- Pinia
- TailwindCSS

- Vue I18n
- Docker

Estrutura de pastas

```
├── src/
│  ├── boot/ # Inicialização de i18n e Pinia
│  ├── components/ # Componentes reutilizáveis (LoginRegister, Popups...)
│  ├── layouts/ # Layouts Quasar (MainLayout, LayoutLogin)
│  ├── pages/ # Páginas Vue (DocumentsPage, PermissoesPage..)
│  ├── stores/ # Stores Pinia: user.ts, documents.ts, ui.ts
│  ├── i18n/ # Arquivos de tradução
│  └── App.vue / main.ts
├── quasar.config.ts
├── package.json
├── Dockerfile
├── docker-compose.yml
└── README.md
```

Principais módulos, classes e funções

Autenticação - registro, login, logout, persistência e expiração de sessão.
Documentos - criação, edição e exclusão.
Permissões temporárias - leitura, comentário, edição com expiração configurável via date/time picker.
Indicadores visuais - cores para "válido", "quase expirado" (<1h) e "expirado".
Comentários - usuários com permissão podem comentar no documento.
Configuração de línguas- textos em PT e EN via Vue I18n.
Date/Time Picker - permissões contam com seleção de expiração configurável.

Instalações de Uso

```
# Instalar dependências
npm install

# Rodar em ambiente dev
quasar dev

# Build para produção
quasar build

# Rodar com Docker
docker-compose up --build
```

Anexos / referências

Backlog Utilizado

- ★ Fase 1 – Configuração Inicial

Criar repositório no GitHub com README inicial.

Configurar projeto com Vue 3 + Quasar Framework + TypeScript.

Integrar TailwindCSS ao Quasar para customização de estilos.

Configurar Vue Router para navegação entre páginas.

Estruturar pastas organizadas (components, views, store, utils, assets).

Configurar Pinia para cache em memória.

Configurar vue-i18n com pelo menos 2 idiomas (pt-BR e en-US).

- ★ Fase 2 – Autenticação e Usuários

Criar modelo User (interface TypeScript).

Criar tela de cadastro de usuários (nome, e-mail, senha).

Criar tela de login.

Registrar automaticamente um usuário Admin padrão ao iniciar aplicação.

Implementar edição de perfil do usuário.

Armazenar usuários em LocalStorage/IndexedDB/JSON Server.

Implementar hash local de senha para não armazenar texto puro.

- ★ Fase 3 – Recursos (Documentos)

Definir recurso como Documento (título + corpo + imagens).

Criar modelo Document (interface TS).

Criar tela de listagem de documentos.

Criar formulário de criação/edição de documento.

Permitir inserir imagens no corpo do documento.

- ★ Fase 4 – Permissões e Acessos

Adicionar Access (usuário + recurso + tipoPermissão + expiração).

Tipos de permissão: leitura, comentário, edição (enum).

Criar tela de gerenciamento de acessos (Admin).

Implementar modal para conceder acesso (usuário + recurso + permissão + expiração).

Implementar date/time picker para expiração.

Implementar regra de expiração automática (comparar data atual)..

Destacar acessos prestes a expirar (menos de 1h → cor amarela/vermelha).

★ Fase 6 – Extras (Diferenciais)

Implementar dark/light mode (toggle).

Prototipar telas no Figma.

Adicionar técnicas de acessibilidade

Criar testes unitários com Jest/Vitest (componentes principais: login, cadastro, acessos).

★ Fase 7 – Deploy e Documentação

Criar Dockerfile e docker-compose.yml.

Configurar build com Quasar CLI para rodar no container.

Publicar no GitHub com Docker + README de instruções.

Criar documento técnico.

Finalizar README detalhado.

➢ Entregáveis finais

SPA completa em Vue + Quasar + Tailwind.

Controle de usuários, documentos e acessos temporários.

Interface intuitiva com todos os componentes obrigatórios.

Funcionalidades bônus implementadas (Pinia, i18n, acessibilidade, Docker).

Repositório GitHub com documentação e imagens do sistema.

Links de pesquisa

Docs VueJS: https://br.vuejs.org/v2/guide/syntax

Docs Quasar: https://quasar.dev/start/how-to-use-vue

Docs Pinia: https://pinia.vuejs.org/core-concepts/state.html

Artigo Base: https://medium.com/@paulobenk/simples-crud-com-vue-quasar-typescript-i18n-54f7b41b9b78

Vídeo Aula: https://www.youtube.com/playlist?list=PLC8QA_SSHxxu3xBicWcQqddRq8BVAgrJj