

library-management-system

Web app for managing a library. Built using React, Django and SQLite

Conceitos e organização

library-api é a Api REST desenvolvida em node usando o framework express.

library-manager é a interface web desenvolvida usando o framework React.js com react-router.

library-database é o arquivo de banco de dados da biblioteca em sqlite



Tecnologias Utilizadas

- Este projeto foi desenvolvido utilizando uma stack moderna e eficiente, integrando diversas linguagens e ferramentas para proporcionar um sistema completo e robusto.



Linguagens e Tecnologias Principais

Tecnologia	Descrição
TypeScript	Linguagem principal utilizada no projeto, tanto no frontend quanto no backend. É fortemente tipada e traz maior segurança e produtividade ao desenvolvimento. O uso é evidenciado por arquivos <code>.ts</code> e <code>.tsx</code> .
Node.js + JavaScript	O backend é executado em Node.js, sendo o ambiente de runtime para o TypeScript compilado. Utiliza o framework Express para a construção da API, conforme indicado no <code>package.json</code> e no arquivo <code>server.ts</code> .
React	Biblioteca JavaScript (usada com TypeScript neste projeto) utilizada para a construção da interface de usuário no frontend. Permite criar componentes reutilizáveis e interativos de forma eficiente.
SQLite + SQL	O banco de dados utilizado é o SQLite , um sistema leve e prático para aplicações menores. As interações com o banco são realizadas por meio de

Tecnologia	Descrição
	SQL , utilizando o ORM Sequelize no backend.
SCSS / CSS	Para estilização da interface, o projeto utiliza SCSS , um pré-processador CSS que permite escrever estilos de forma mais organizada, reutilizável e com recursos adicionais como variáveis e mixins.

Configurando e executando o projeto

1. Instalar o node

- Linux
 - Para sistemas baseados em Ubuntu `sudo apt install nodejs npm`
 - Para sistemas baseados em fedora `sudo dnf install node` . O gerenciador de pacotes npm será instalado automaticamente.
- Windows <https://nodejs.org/en/download>

2. Instalando as dependências

Depencias npm geralmente são instaladas localmente para cada projeto, a lista de dependências de um projeto node está sempre localizada em `package.json > dependencies` .

Para instalar as dependências desse projeto:

- Acessar `library-api` e executar `npm install` os seguintes pacotes serão instalados:
 - Express
 - Sequelize
 - Sqlite3
- Acessar `library-manager` e executar `npm install` , os seguintes pacotes serão instalados:
 - React
 - React Router
 - React-Bootstrap
 - React-Toastify
 - Bootstrap icons

3. Executando

- Criar um terminal e acessar o diretório `library-api` e executar `npm run dev`, a api será iniciada na porta 3000.
- Criar um novo terminal e acessar o diretório `library-manager` e executar `npm run dev`, o frontend será iniciado e o link de acesso ficara disponível no terminal. Por padrão: <http://localhost:5173/>

As variáveis de ambiente já estão configuradas com os valores necessários para o funcionamento, portanto basta executar os dois comandos acima em terminais diferentes e interagir com a interface web.



Funcionalidades da Aplicação

Catálogo de Obras (Home)

- A página inicial da biblioteca apresenta um catálogo com todas as obras disponíveis, exibindo informações como:
 - Título da obra
 - Autor
 - Editora
 - ISBN
 - Capa do livro

É possível pesquisar livros por qualquer um desses critérios. Cada obra possui um botão “Ver detalhes”, que redireciona para uma página com informações complementares.



Cadastro de Usuários

- A aplicação possui duas telas de cadastro:
 - **Cadastro de Aluno:** acessível a partir da Home, permite preencher os dados pessoais, RA e curso do aluno.
 - **Cadastro de Servidor:** acessível apenas por administradores. Permite inserir o registro funcional e selecionar o departamento do servidor.



Dashboard do Administrador

- Além do catálogo de obras, o administrador tem acesso a um painel de controle, com as seguintes funcionalidades:
 - Cadastrar novos servidores
 - Excluir alunos e servidores existentes



Dashboard do Aluno

- Além do catálogo, os alunos têm acesso a uma aba de empréstimos, que exibe:
 - Obras atualmente emprestadas
 - Prazo de devolução
 - Quantidade de exemplares emprestados
 - Status de devolução de cada exemplar



Dashboard do Servidor

- O dashboard dos servidores oferece acesso a três seções adicionais:



Gerenciamento de Obras

- Visualização de ISBN e título das obras cadastradas
- Link para visualizar detalhes da obra
- Opção para excluir obras
- Barra de pesquisa para localizar obras específicas
- Cadastro de novas obras com número de exemplares



Empréstimo

- Definição do prazo de devolução
- Informar o CPF do aluno
- Seleção das obras emprestadas
- Registro do empréstimo



Devolução

- Pesquisa de empréstimos ativos por CPF
- Visualização das obras emprestadas
- Indicação de status de devolução
- Botão para registrar devolução de cada obra

Login

- Tela de autenticação com:
 - Campo para CPF (login)
 - Campo para senha
 - Link para criação de conta por alunos ainda não cadastrados

Página da Obra

- Exibe informações detalhadas da obra selecionada:
 - Número de páginas
 - Descrição da obra
 - Lista de exemplares disponíveis para reserva
 - Indicação de exemplares já emprestados

Banco de Dados da Biblioteca

Este repositório contém a estrutura de um banco de dados SQLite utilizado para o gerenciamento de uma biblioteca. Abaixo está a descrição do esquema da base de dados

◆ Tabela usuarios

- Armazena informações de todos os usuários cadastrados, sejam alunos, servidores ou administradores.

Campo	Tipo	Descrição
cpf	VARCHAR(255)	Identificador único do usuário (PK)
nome	VARCHAR(255)	Nome completo do usuário
endereço	VARCHAR(255)	Endereço residencial
dataNascimento	DATETIME	Data de nascimento
contato	VARCHAR(255)	Telefone ou e-mail

Campo	Tipo	Descrição
tipo	VARCHAR(255)	Tipo de usuário (Alu , Ser , Adm)
senha	VARCHAR(255)	Senha de acesso

◆ Tabela alunos

- Relaciona os usuários do tipo aluno com seus dados acadêmicos.

Campo	Tipo	Descrição
ra	VARCHAR(255)	Registro acadêmico (PK)
curso	VARCHAR(255)	Curso do aluno
cpf	VARCHAR(255)	CPF do usuário (FK para usuarios)

◆ Tabela servidores

- Contém os dados dos usuários que são servidores da instituição.

Campo	Tipo	Descrição
registro	VARCHAR(255)	Registro funcional (PK)
departamento	VARCHAR(255)	Departamento de atuação
cpf	VARCHAR(255)	CPF do usuário (FK para usuarios)

◆ Tabela obras

- Armazena os dados bibliográficos das obras disponíveis na biblioteca.

Campo	Tipo	Descrição
isbn	VARCHAR(255)	Código ISBN da obra (PK)
titulo	VARCHAR(255)	Título do livro

Campo	Tipo	Descrição
autor	VARCHAR(255)	Nome do autor
editora	VARCHAR(255)	Nome da editora
paginas	INTEGER	Quantidade de páginas
descricao	TEXT	Descrição ou sinopse da obra
linkCapa	VARCHAR(255)	URL da imagem da capa

◆ Tabela exemplares

- Contém os exemplares físicos das obras.

Campo	Tipo	Descrição
tombo	INTEGER	Número de tombamento do exemplar (PK)
dataAquisicao	DATETIME	Data de aquisição
sessao	VARCHAR(255)	Localização física na estante
status	VARCHAR(255)	Status do exemplar (ex: disp para disponível)
isbn	VARCHAR(255)	ISBN da obra (FK para obras)

◆ Tabela emprestimos

- Registra os empréstimos realizados pelos usuários.

Campo	Tipo	Descrição
id	INTEGER	Identificador do empréstimo (PK)
dataHoraEmprestimo	DATETIME	Data e hora em que o empréstimo ocorreu
prazoDevolucao	DATETIME	Data limite para devolução
dataHoraDevolucao	DATETIME	Data em que o exemplar foi devolvido
cpf	VARCHAR(255)	CPF do usuário (FK para usuarios)

◆ Tabela emprestimo-exemplares

- Tabela de associação entre empréstimos e exemplares (muitos-para-muitos).

Campo	Tipo	Descrição
EmprestimoId	INTEGER	ID do empréstimo (FK para <code>emprestimos</code>)
ExemplarTombo	INTEGER	Número do tombo (FK para <code>exemplares</code>)
createdAt	DATETIME	Data de criação do registro
updatedAt	DATETIME	Data da última atualização

Relacionamentos Principais

- usuarios \longleftrightarrow alunos / servidores: via campo `cpf`
- obras \longleftrightarrow exemplares: via campo `isbn`
- usuarios \longleftrightarrow emprestimos: via campo `cpf`
- emprestimos \longleftrightarrow exemplares: via tabela intermediária `emprestimo-exemplares`

Referências

Auth: <https://medium.com/@sustiono19/how-to-create-a-protected-route-in-react-with-react-router-dom-v7-6680dae765fb>