#### Cadastro de Alunos

## Descrição do Projeto

O projeto "Cadastro de Alunos" é uma aplicação desenvolvida em **React** para o front-end e **Node.js** com **MySQL** para o back-end, seguindo os princípios de componentes reutilizáveis e comunicação entre componentes via **props**. O sistema permite cadastrar, listar, atualizar e excluir alunos de um banco de dados MySQL, utilizando uma API REST integrada via **Axios**.

## **Objetivo**

O objetivo deste projeto é consolidar o entendimento sobre:

- Criação e reutilização de componentes em React.
- Troca de dados entre componentes via **props**.
- Uso do atributo children.

### **Componentes Utilizados**

#### 1. Navbar

- a. Exibe o título da aplicação.
- b. Simples barra de navegação que contém o nome "Cadastro de Alunos".

#### 2. Formulario De Aluno

- a. Componente responsável pelo formulário de cadastro e edição de alunos.
- b. Utiliza **props** para passar e manipular dados do estado principal.
- c. Contém campos para **nome**, **idade** e **curso** e um botão para submeter as informações.

#### 3. Aluno

a. Componente responsável por exibir as informações de um aluno específico.

b. Utiliza o atributo **children** para customizar o conteúdo adicional, como botões de editar e excluir.

#### 4. ListaDeAlunos

- a. Componente responsável por renderizar dinamicamente uma lista de alunos cadastrados.
- b. Recebe a lista de alunos via **props** e passa cada aluno para o componente Aluno.

### Fluxo de Dados

- A aplicação é baseada no fluxo de dados do React, utilizando props para comunicação entre componentes.
- O componente principal (App.jsx) gerencia os estados e passa dados e funções via props para os componentes filhos, como FormularioDeAluno e ListaDeAlunos.
- Axios é utilizado para fazer requisições HTTP à API do back-end, que realiza as operações de criação, leitura, atualização e exclusão no banco de dados MySQL.

# Configuração do Projeto

# Front-End (React com Vite)

1. Clone o repositório:

git clone <link-do-repositorio>

2. Instale as dependências:

npm install

3. Inicie o servidor de desenvolvimento:

npm run dev

# Back-End (Node.js com MySQL)

1. Clone o repositório:

```
git clone <link-do-repositorio>
```

2. Instale as dependências:

```
npm install
```

3. Configure as variáveis de ambiente: Crie um arquivo .env na raiz do projeto com as seguintes informações:

```
DB_HOST=localhost
DB_USER=root
DB_PASSWORD=sua_senha
DB_NAME=cadastro_alunos
```

4. Execute as migrações do banco de dados MySQL:

```
USE cadastro_alunos;

CREATE TABLE alunos (
   id INT AUTO_INCREMENT PRIMARY KEY,
   nome VARCHAR(255) NOT NULL,
   idade INT NOT NULL,
   curso VARCHAR(255) NOT NULL
);
```

CREATE DATABASE cadastro\_alunos;

5. Inicie o servidor:

```
npm start
```

### Rotas da API

- 1. Adicionar Aluno (POST /adicionar Aluno)
  - a. Adiciona um novo aluno no banco de dados.
- 2. **Listar Alunos** (GET /listarAlunos)
  - a. Retorna a lista de alunos cadastrados.
- 3. Atualizar Aluno (PUT /atualizar Aluno/:id)
  - a. Atualiza as informações de um aluno específico.
- 4. **Deletar Aluno** (DELETE /deletarAluno/:id)
  - a. Remove um aluno do banco de dados.

# Como Usar a Aplicação

#### 1. Cadastro de Alunos:

- a. Preencha os campos "Nome", "Idade" e "Curso" no formulário e clique no botão **Adicionar Aluno**.
- b. O aluno será inserido no banco de dados e exibido na lista de alunos.

# 2. Edição de Alunos:

- a. Clique no botão **Editar** ao lado do aluno que deseja modificar.
- b. Os dados do aluno serão preenchidos no formulário. Altere os dados e clique em **Atualizar Aluno**.

### 3. Exclusão de Alunos:

- a. Clique no botão **Excluir** ao lado do aluno que deseja remover.
- b. O aluno será removido do banco de dados e da lista.

# **Tecnologias Utilizadas**

#### • Front-End:

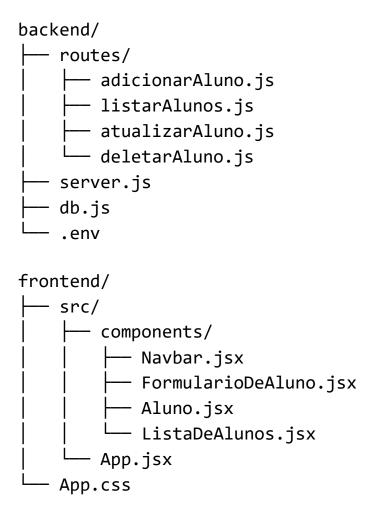
- o React (com Vite)
- Axios
- o CSS

### • Back-End:

- Node.js
- o Express.js
- MySQL

o dotenv (para gerenciamento de variáveis de ambiente)

# **Estrutura do Projeto**



### **Testes**

- A aplicação foi testada utilizando o Postman para testar as rotas do back-end (CRUD: Create, Read, Update, Delete).
- O front-end foi testado em diferentes resoluções de tela para garantir a responsividade.

## **Considerações Finais**

Este projeto cumpre os requisitos propostos, como a criação e reutilização de componentes, uso de **props** para comunicação entre

componentes, e o uso de **children** para personalizar conteúdo dentro de um componente. Além disso, a aplicação está devidamente integrada com o back-end utilizando Axios e segue uma arquitetura REST para gerenciar o cadastro de alunos no banco de dados MySQL.