PROJETO CRUD - FRONTEND

SUMÁRIO

OBJETIVO

Criar uma API RESTful com todas as funcionalidades de CRUD. Criar um frontend intuitivo, sincronizado com o backend, para realizar as operações de visualização, inserção, edição e deleção dos dados (CRUD).

DEPENDÊNCIAS

O projeto será implementado conforme abaixo:

- **IDE:** Visual Studio Code (v. 1.88.1)
- **Gerenciador de Dependências:** Node (v. 18.19) / Yarn (v. 1.22.21)
- **Linguagem:** JavaScript/TypeScript
- **Versionamento:** Git e GitHub
- **Projeto:** VITE

CASO DE USO

Uma Universidade precisa cadastrar os seus alunos, conforme segue abaixo:

- NOME
- CPF
- DATA DE NASCIMENTO
- RENDA

TESTES A SEREM REALIZADOS NO POSTMAN

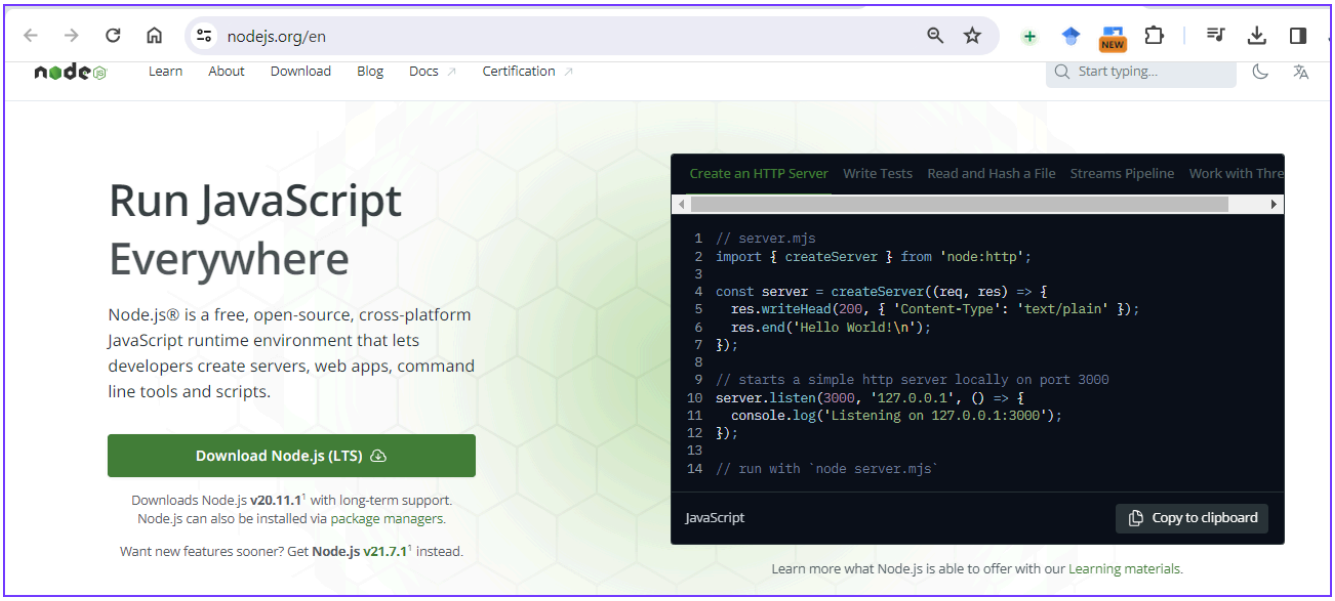
- VISUALIZAR TODOS OS ALUNOS NUMA TABELA
- INSERIR UM NOVO ALUNO
- EDITAR UM ALUNO EXISTENTE
- DELETAR UM ALUNO EXISTENTE

***OBS:** Não será possível inserir um formulário vazio, bem como todos os campos devem ser preenchidos conforme regras de validação.*

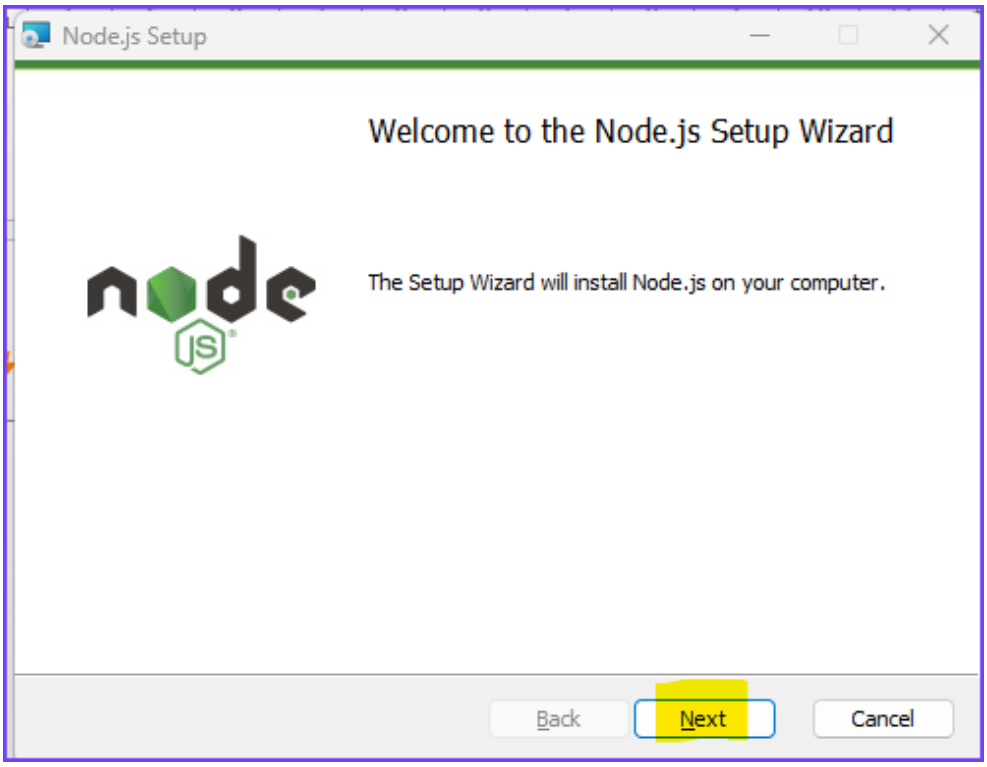
INSTALAÇÃO DO NODEJS

BAIXAR

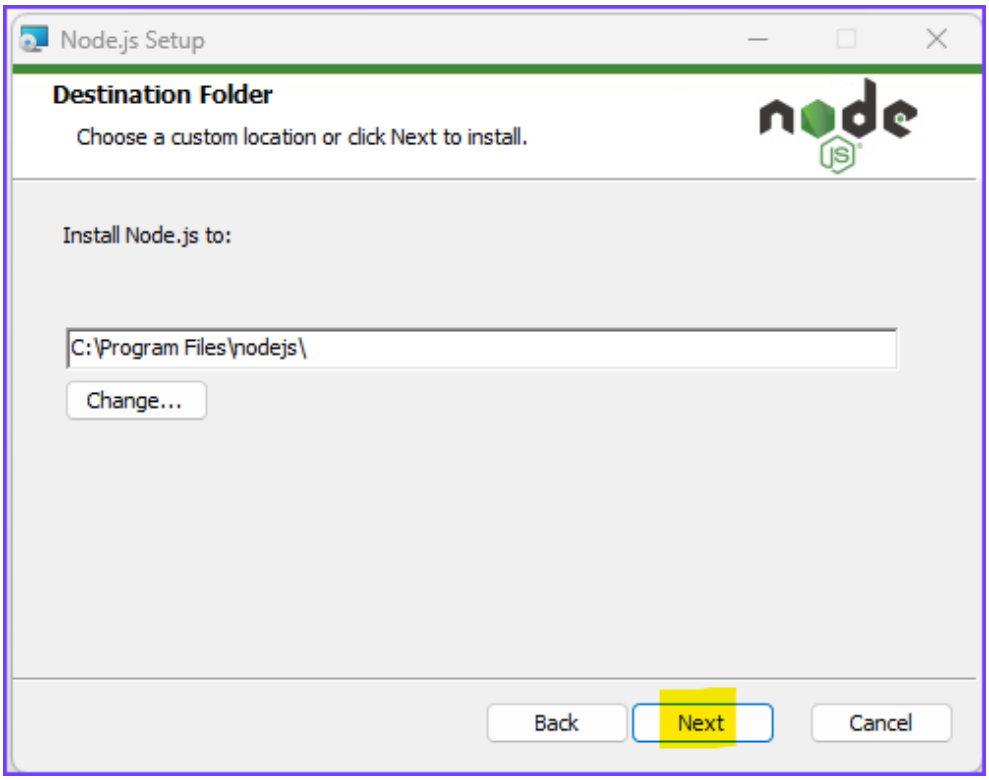
- https://nodejs.org

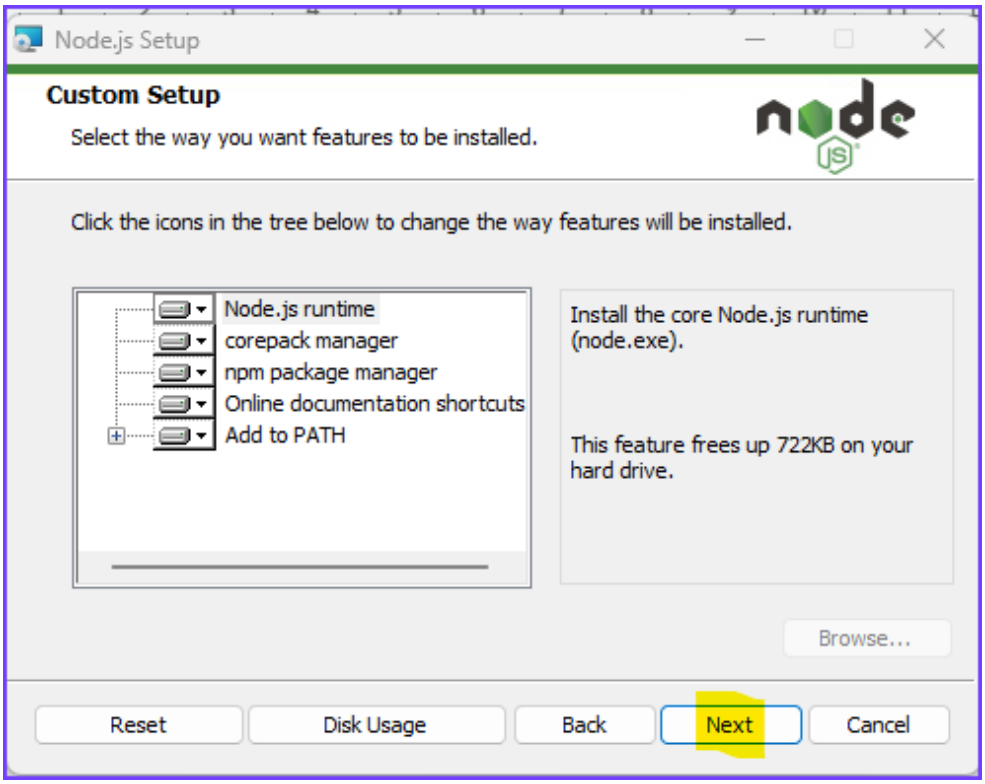


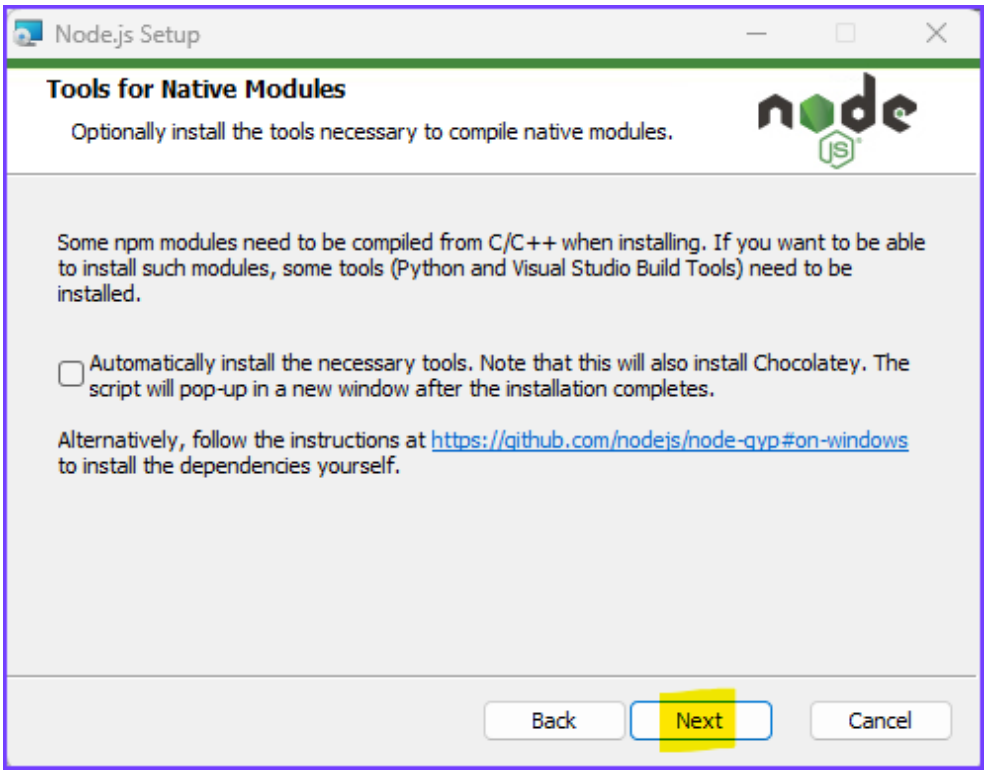
EXECUTAR

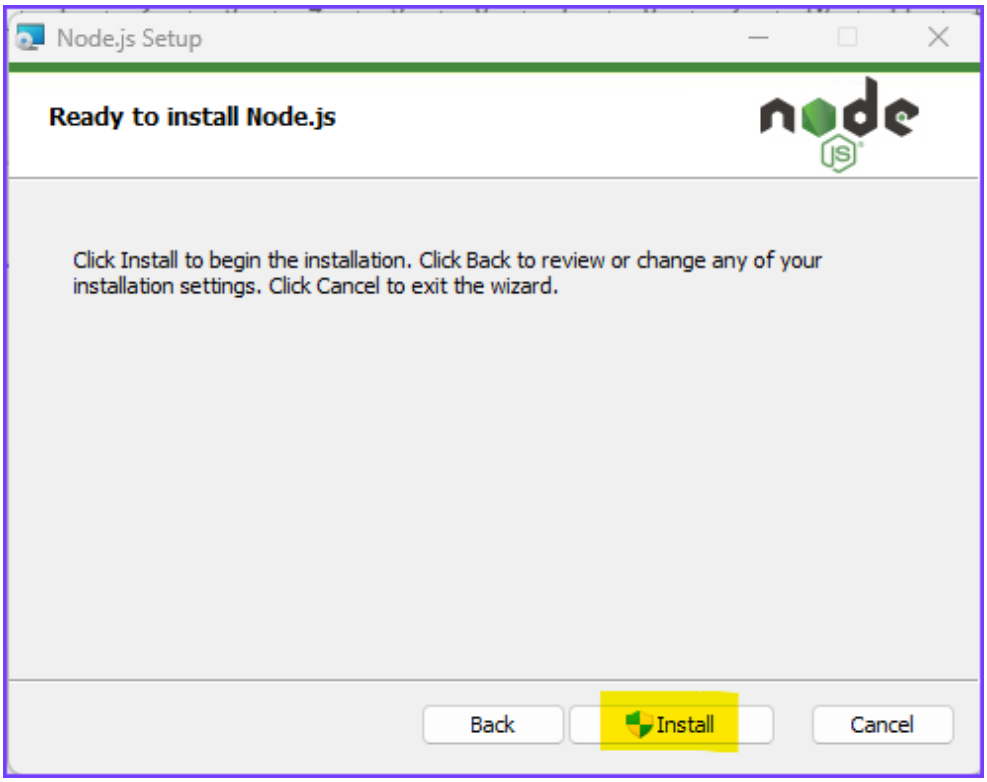

Node.js Setup
Welcome to the Node.js Setup Wizard
The Setup Wizard will install Node.js on your computer.
Back
Next
Cancel

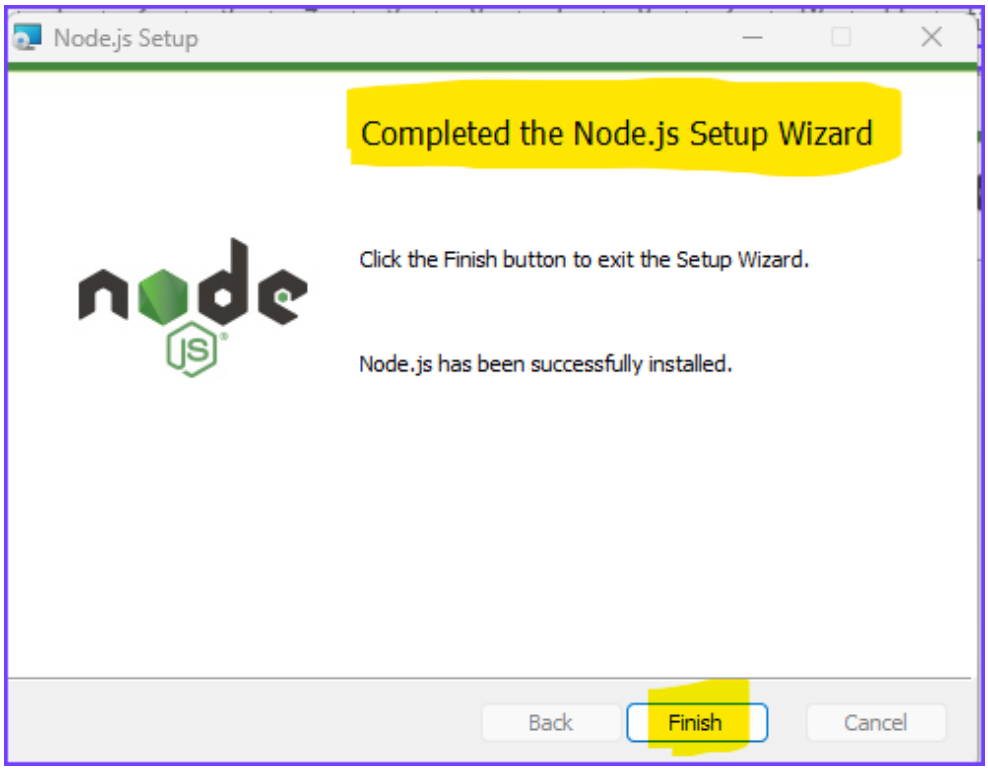

Node.js Setup
End-User License Agreement
Please read the following license agreement carefully
Node.js is licensed for use as follows:
Copyright Node.js contributors. All rights reserved.
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject
I accept the terms in the License Agreement
Print
Back
Next
Cancel


Node.js Setup
Destination Folder
Choose a custom location or click Next to install.
Install Node.js to:
C:\Program Files\nodejs\
Change...
Back
Next
Cancel


Node.js Setup
Custom Setup
Select the way you want features to be installed.
Click the icons in the tree below to change the way features will be installed.
Node.js runtime
corepack manager
npm package manager
Online documentation shortcuts
Add to PATH
Install the core Node.js runtime (node.exe).
This feature frees up 722KB on your hard drive.
Browse...
Reset
Disk Usage
Back
Next
Cancel

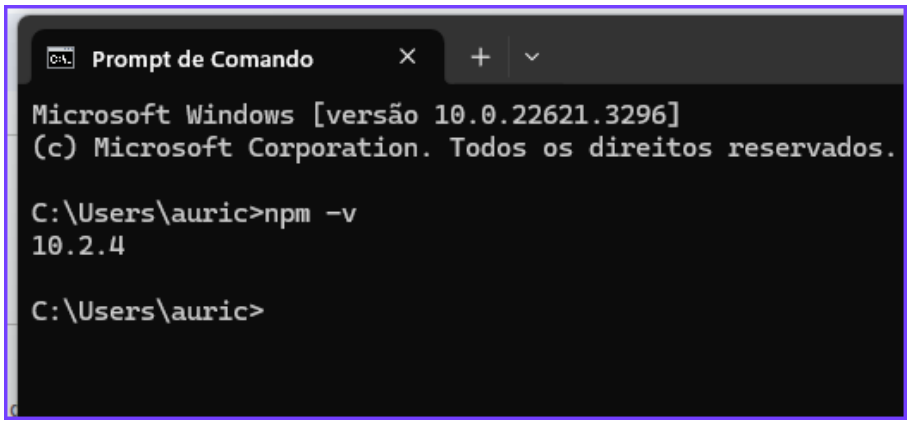

Node.js Setup
Tools for Native Modules
Optionally install the tools necessary to compile native modules.
Some npm modules need to be compiled from C/C++ when installing. If you want to be able to install such modules, some tools (Python and Visual Studio Build Tools) need to be installed.
Automatically install the necessary tools. Note that this will also install Chocolatey. The script will pop-up in a new window after the installation completes.
Alternatively, follow the instructions at https://github.com/nodejs/node-gyp#on-windows to install the dependencies yourself.
Back
Next
Cancel


Node.js Setup
Ready to install Node.js
Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.
Back
Install
Cancel



TESTAR NO TERMINAL

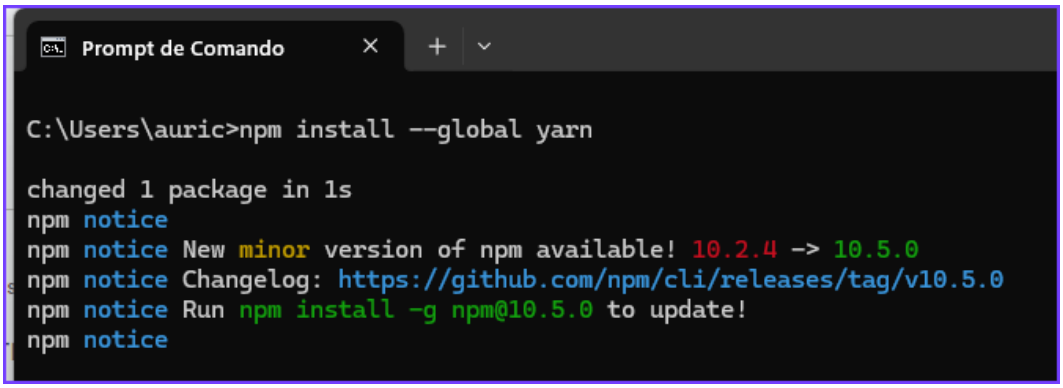
- npm -v



IMPLEMENTAR O YARN

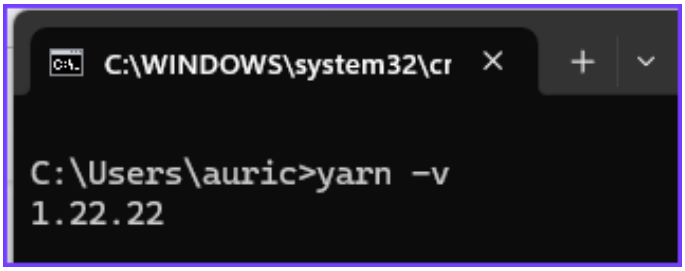
INSTALAR

- npm install --global yarn



TESTAR

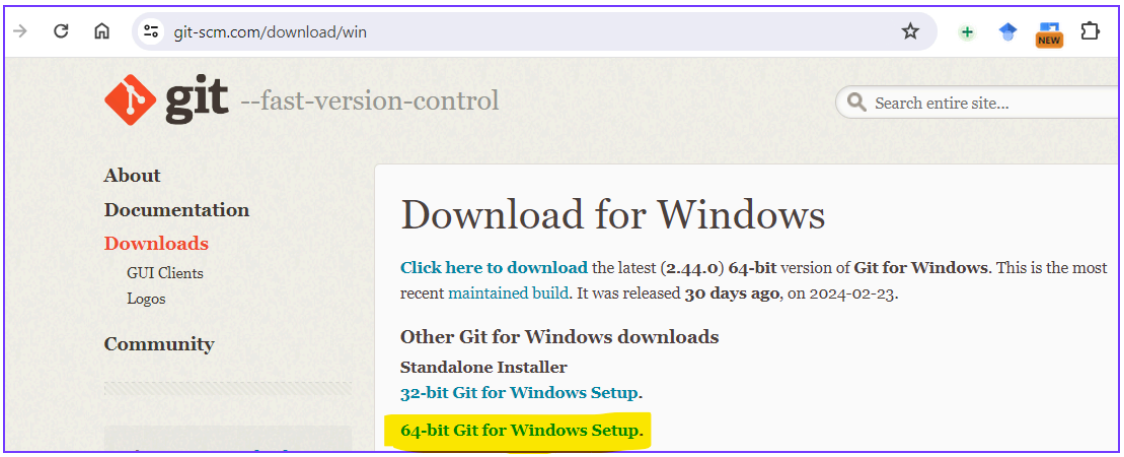
- yarn -v



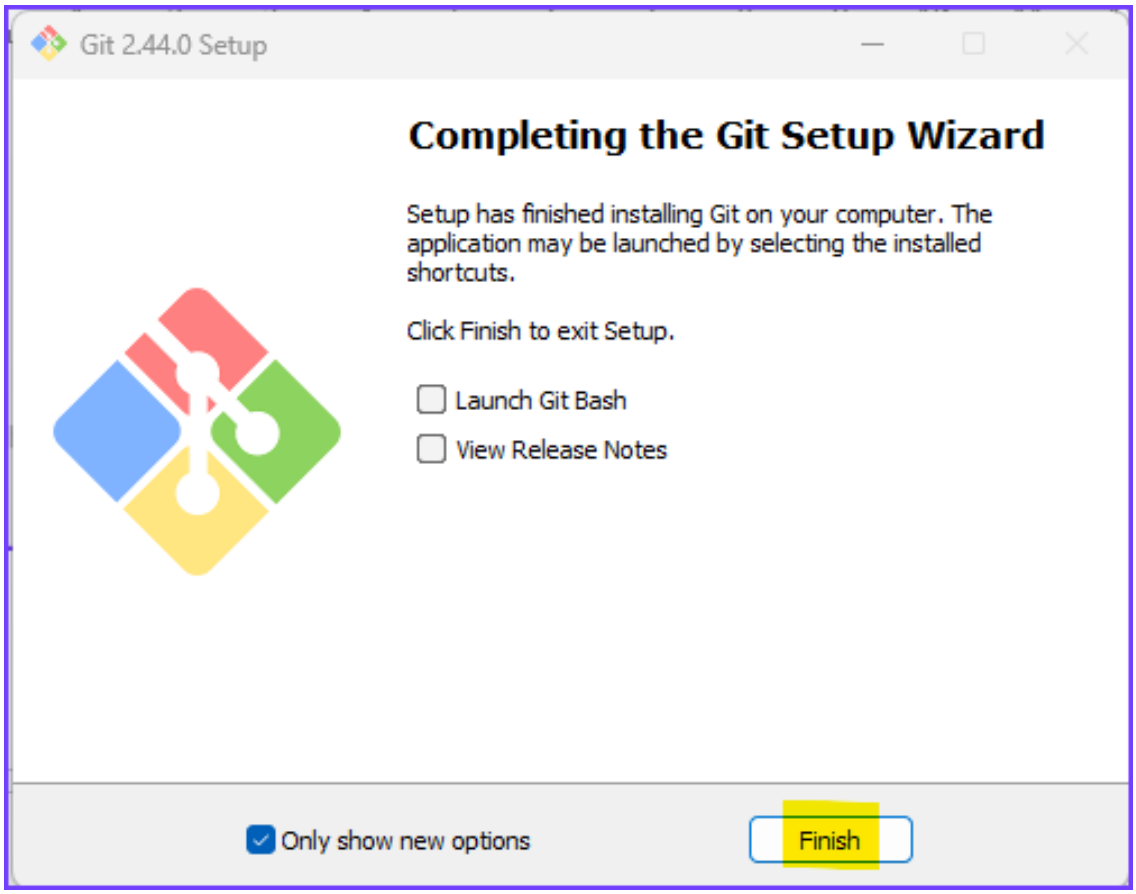
IMPLEMENTAR O GIT

BAIXAR

- https://git-scm.com/download/win

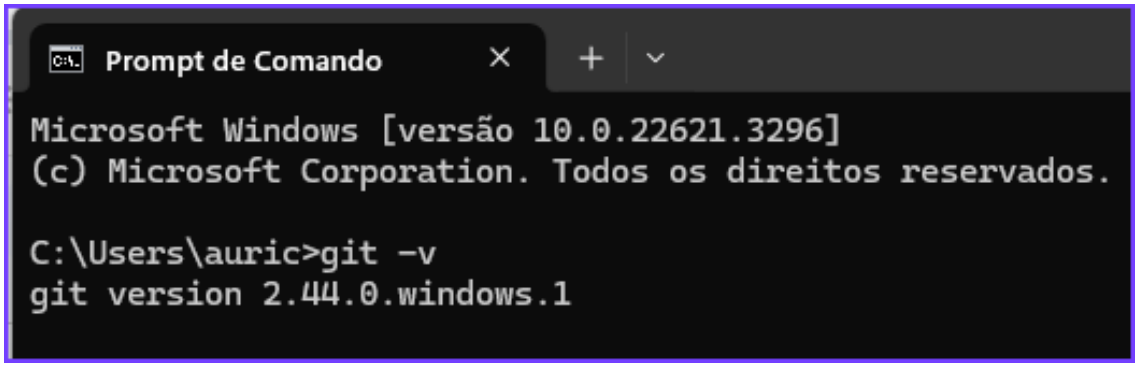


INSTALAR


Git 2.44.0 Setup
Completing the Git Setup Wizard
Setup has finished installing Git on your computer. The application may be launched by selecting the installed shortcuts.
Click Finish to exit Setup.
Launch Git Bash
View Release Notes
Only show new options
Finish

TESTAR

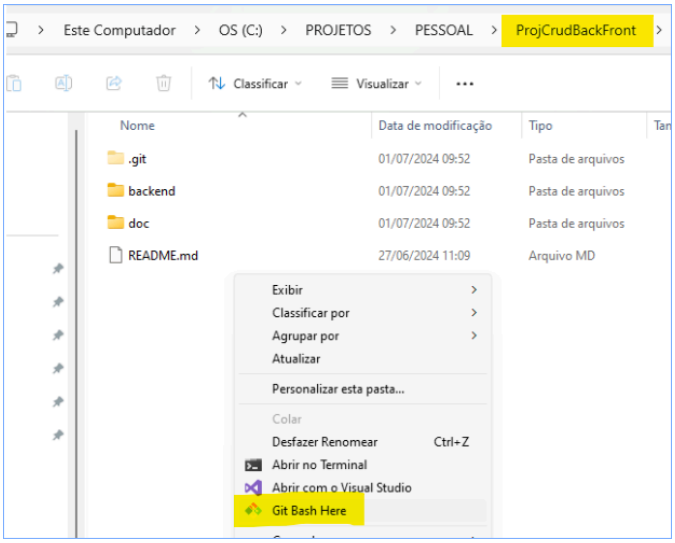
- git -v


Prompt de Comando
Microsoft Windows [versão 10.0.22621.3296]
(c) Microsoft Corporation. Todos os direitos reservados.
C:\Users\auric>git -v
git version 2.44.0.windows.1

IMPLEMENTAR O FRONTEND

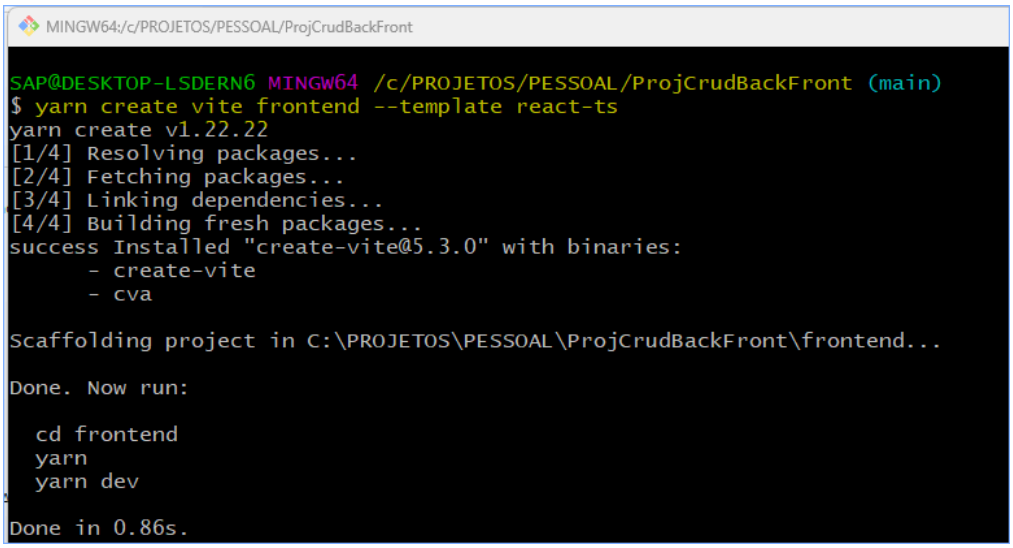
ABRIR O GIT NO WORKSPACE DO PROJETO

- C:\PROJETOS\PESSOAL\ProjCrudBackFront



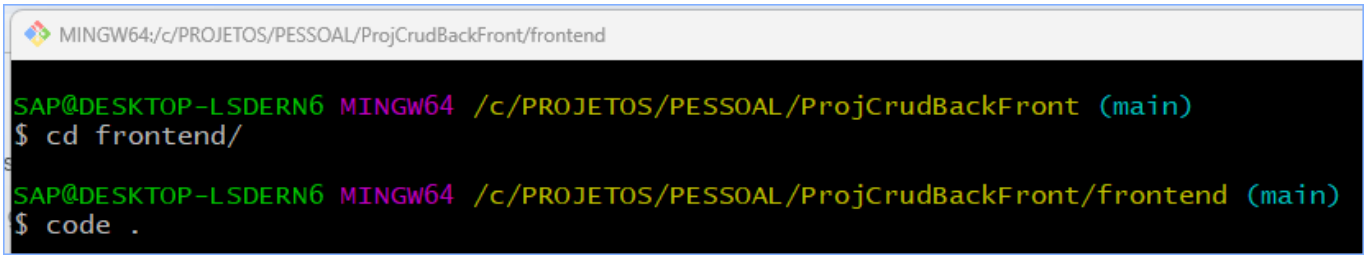
CRIAR O PROJETO

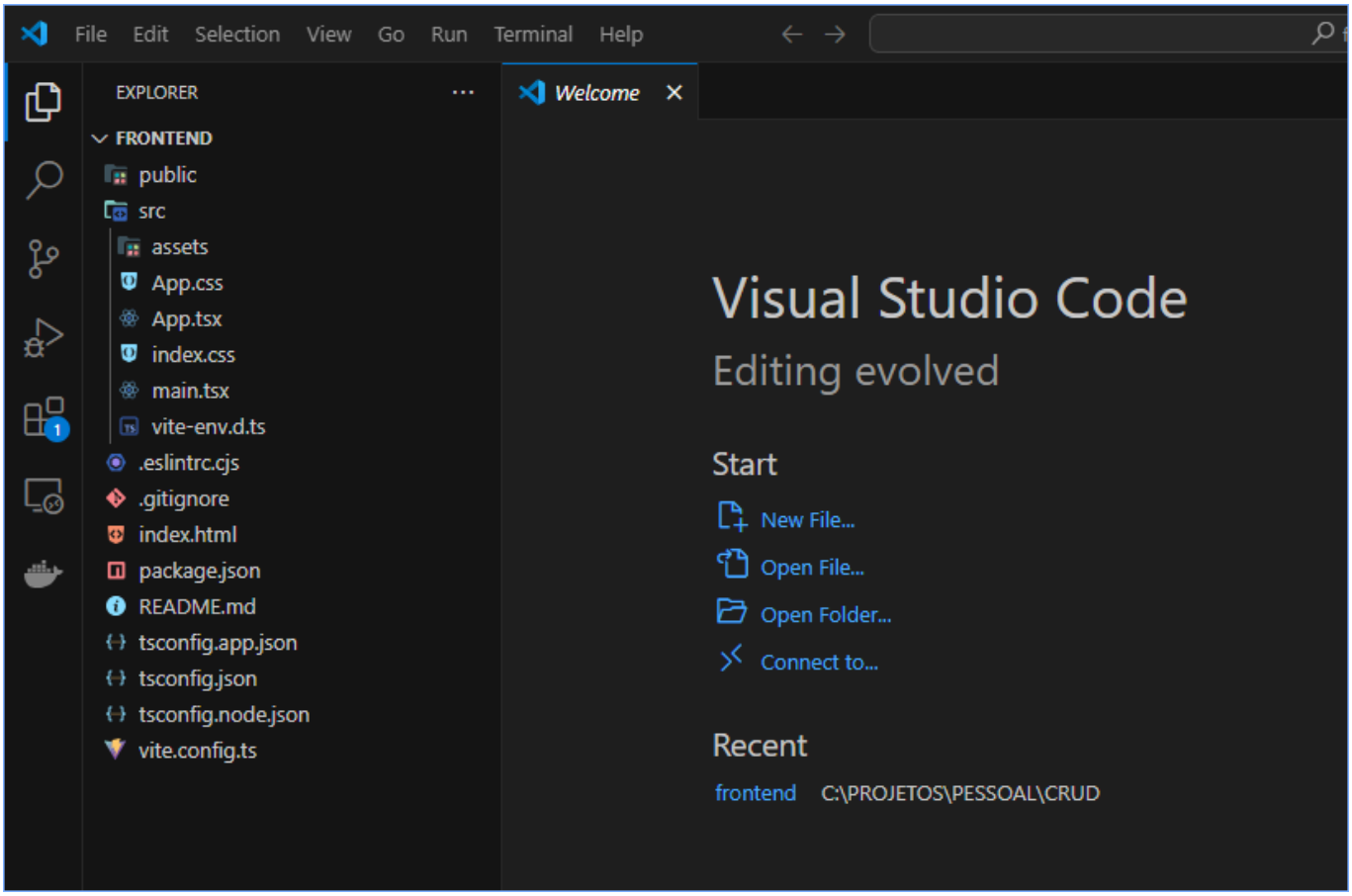
- yarn create vite frontend --template react-ts



ABRIR COM O VSCODE

- cd /frontend
- code .

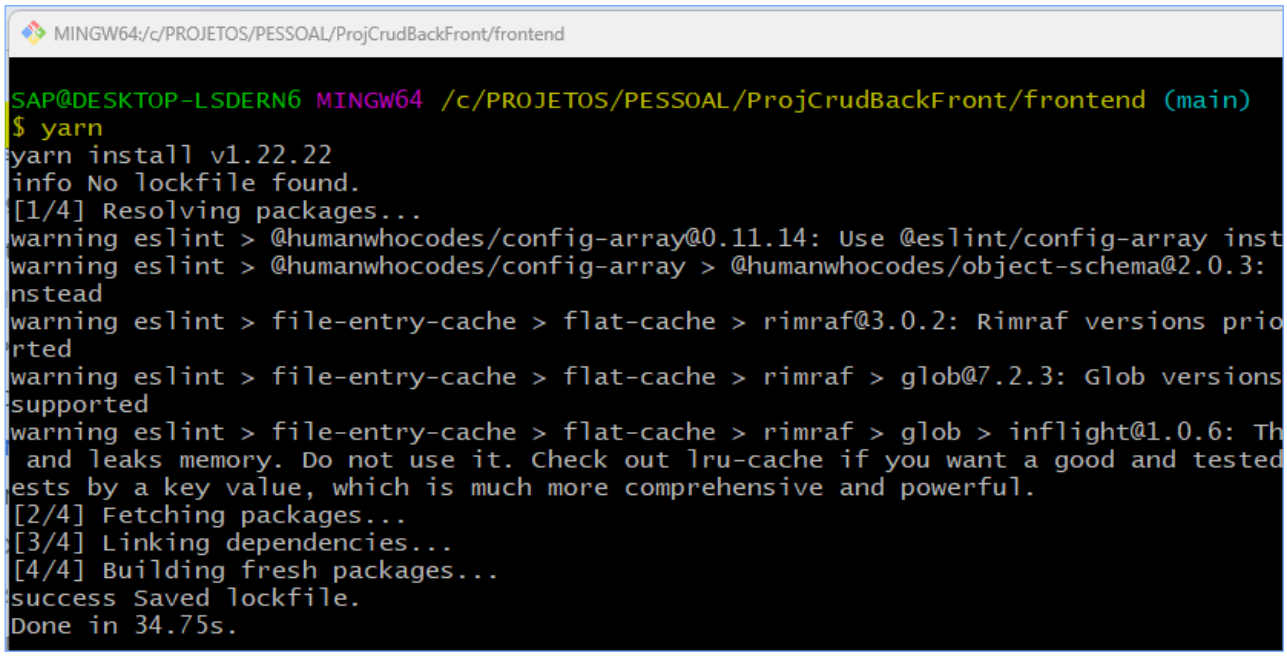




DEPENDÊNCIAS

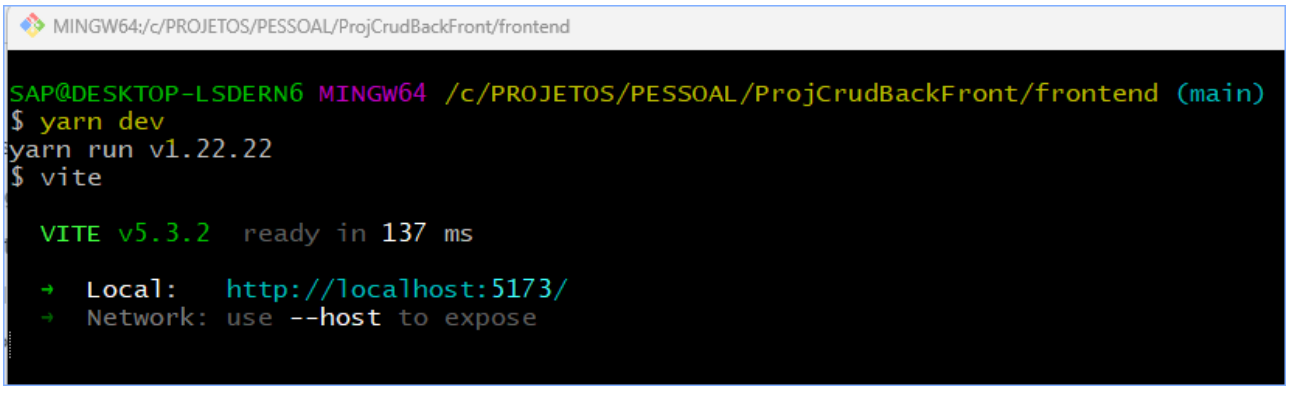
ATUALIZAR AS DEPENDÊNCIAS DO YARN

- yarn

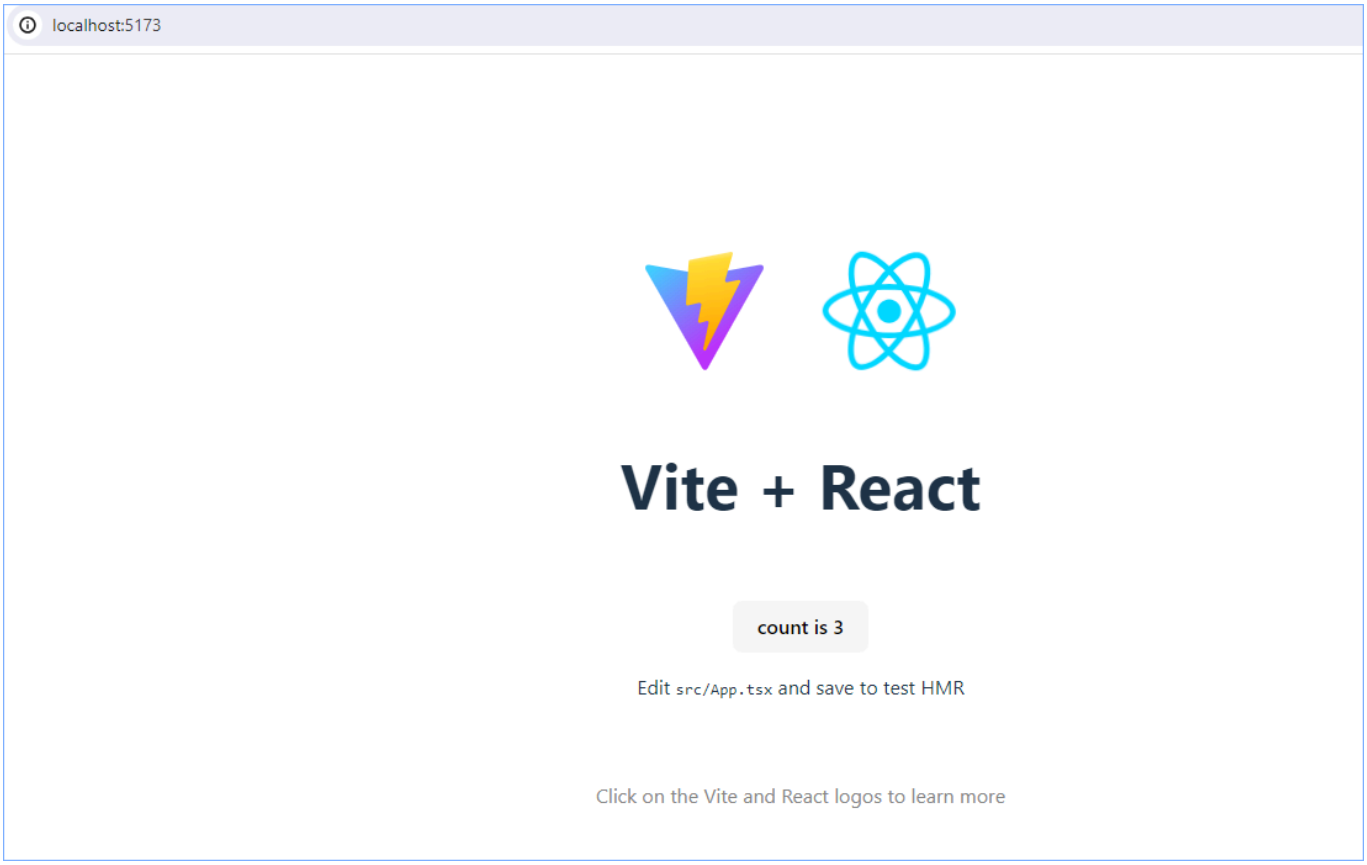


RODAR O PROJETO

- yarn dev

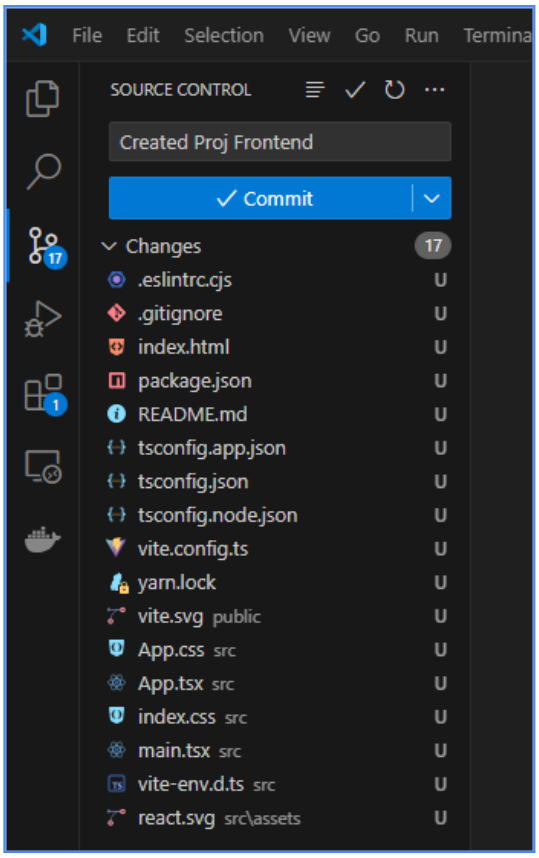


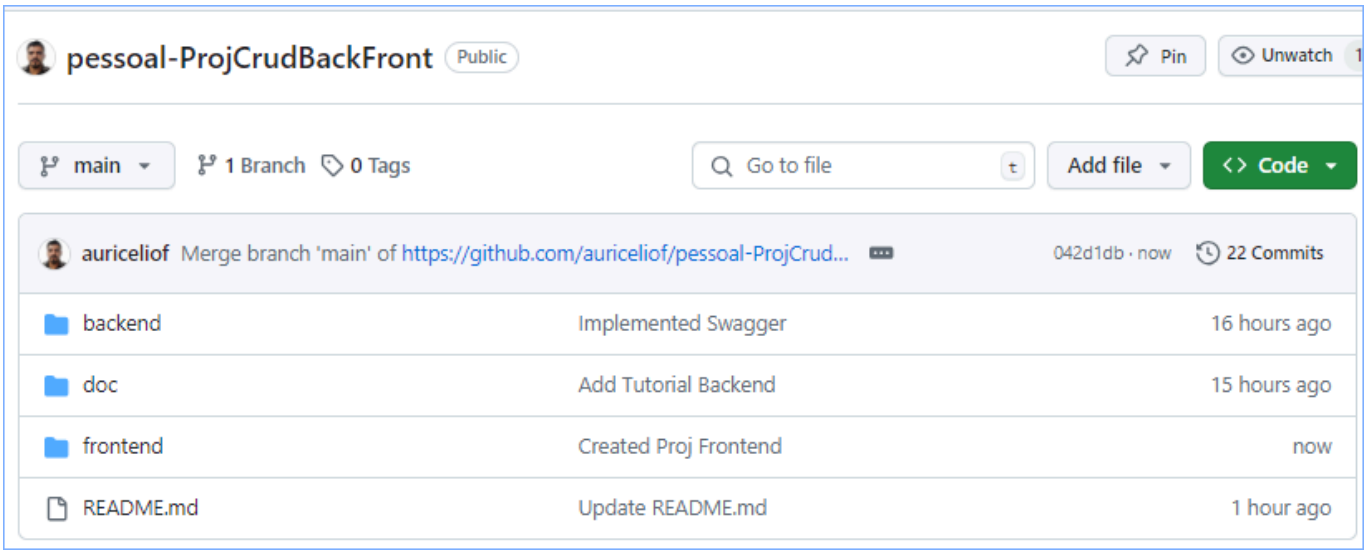
VISUALIZAÇÃO WEB



GitHub-1

NOTA: *Neste momento, o projeto já deve ter sido criado anteriormente no git, para o projeto backend.*





LIMPAR O PROJETO

INDEX.HTML

Antes

index.html

```html
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" type="image/svg+xml" href="/vite.svg" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Vite + React + TS</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.tsx"></script>
  </body>
</html>
```

Depois

- Mudar o nome do projeto

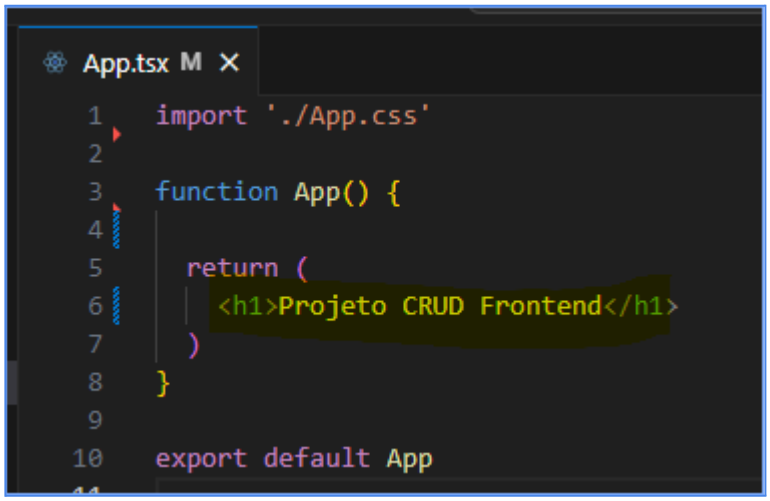
index.html M

```html
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" type="image/svg+xml" href="/vite.svg" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>ProjCrudFront</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.tsx"></script>
  </body>
</html>
```

APP.TSX

Antes

Depois



***NOTA**: Apaga todo o conteúdo, deixando apenas um retorno mínimo.*

INDEX.CSS

Antes

```css
:root {
  font-family: Inter, system-ui, Avenir, Helvetica, Arial, sans-serif;
  line-height: 1.5;
  font-weight: 400;

  color-scheme: light dark;
  color: rgba(255, 255, 255, 0.87);
  background-color: #242424;

  font-synthesis: none;
  text-rendering: optimizeLegibility;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
}

a {
  font-weight: 500;
  color: #646cff;
  text-decoration: inherit;
}
a:hover {
  color: #535bf2;
```

Depois

```css
:root {
  --color-red: #ff0000;
}

```

***NOTA**: Apaga todo o conteúdo, deixando apenas uma cor para teste.*

APP.CSS

Antes

```css
#root {
  max-width: 1280px;
  margin: 0 auto;
  padding: 2rem;
  text-align: center;
}

.logo {
  height: 6em;
  padding: 1.5em;
  will-change: filter;
  transition: filter 300ms;
}
.logo:hover {
  filter: drop-shadow(0 0 2em #646cffaa);
```

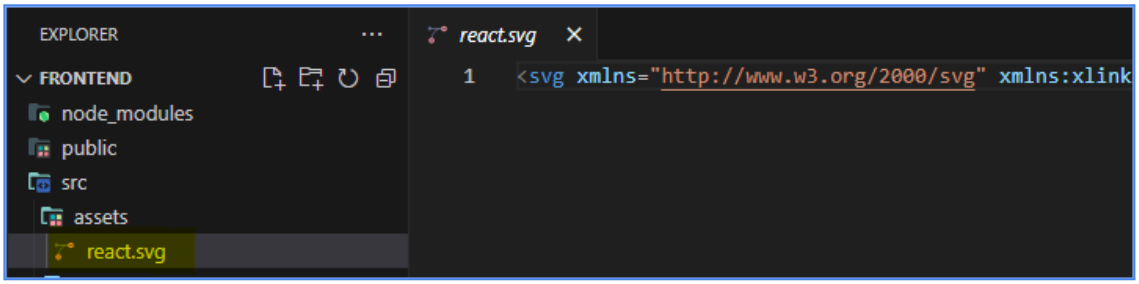
Depois

```css
h1 {
  color: var(--color-red);
}
```

***NOTA**: Apaga todo o css. Deixar apenas uma cor puxando a variável do index.css.*

ASSETS

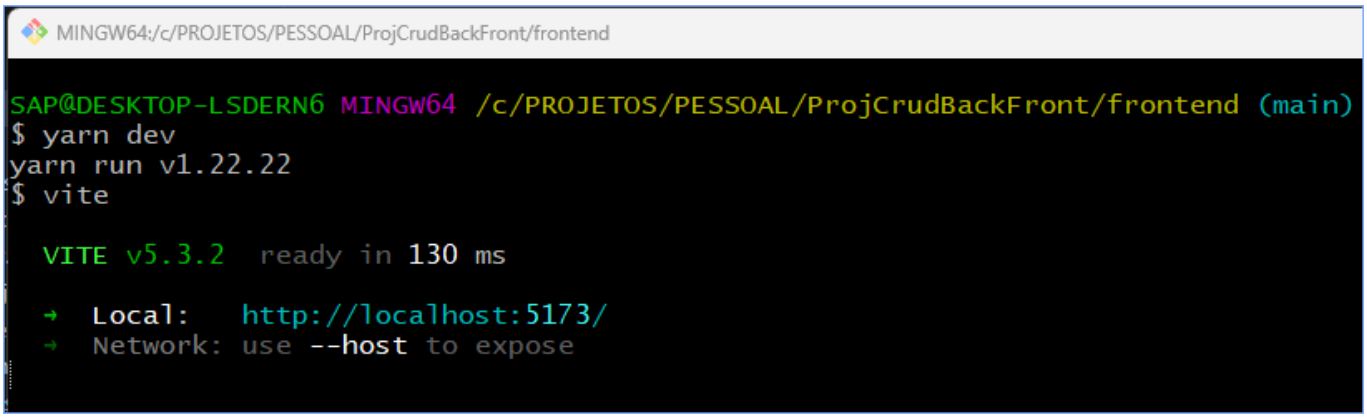
Antes



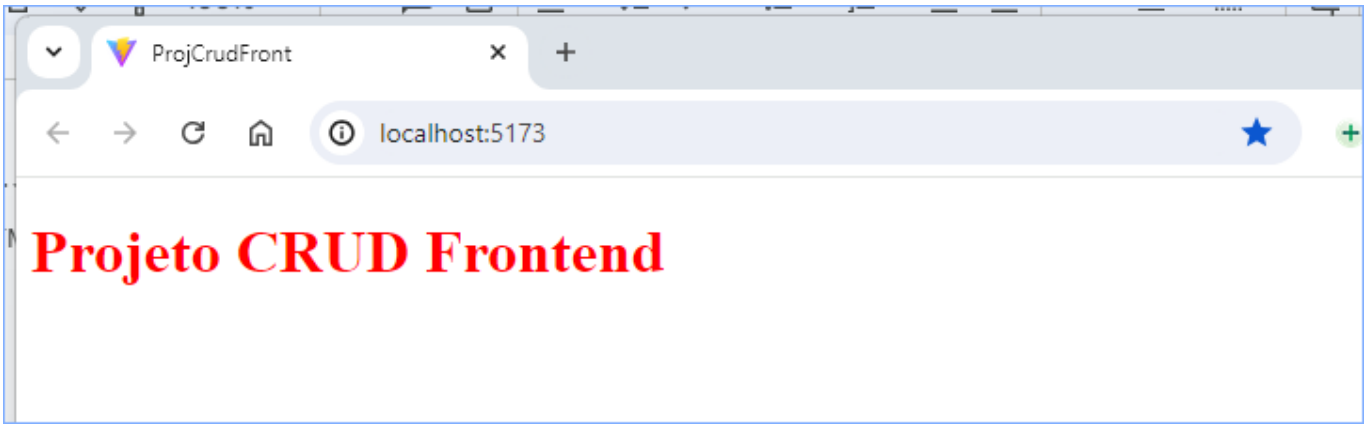
Depois

- *Deletar imagem.*

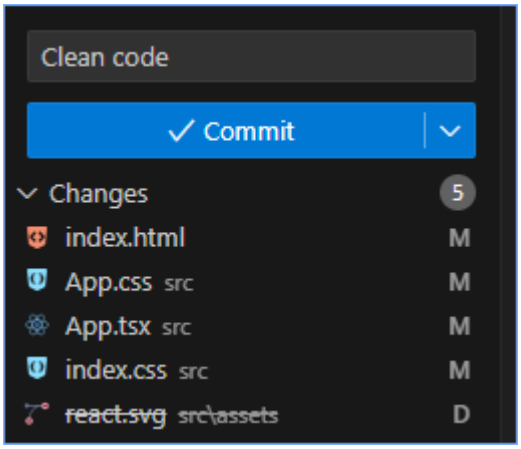
RODAR O PROJETO



VISUALIZAÇÃO WEB



GitHub-2



CSS

NOTA: *Aqui iremos colocar todo o CSS genérico para o nosso projeto.*

FONTE

```
@import url("https://fonts.googleapis.com/css2?family=Open+Sans:wght@400;700&display=swap");
```

index.css

```
@import url("https://fonts.googleapis.com/css2?family=Open+Sans:wght@400;700&display=swap");
```

COR

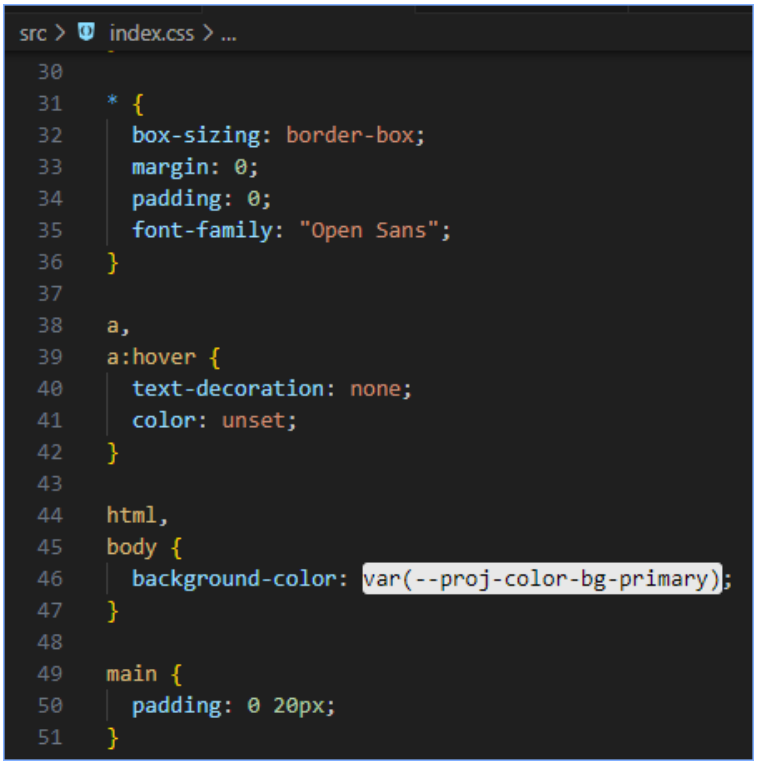
```
:root {
  --proj-color-bg-primary: #e8e8e8;
  --proj-color-bg-secondary: #074307;
  --proj-color-bg-tertiary: #cac7c7;
  --proj-color-bg-quaternary:#c9d9f2;
  --proj-color-bg-quinternary:  #b8c6b9;

  --proj-color-card-bg: #fff;
  --proj-color-card-border: #d9d9d9;

  --proj-color-btn-primary: #2571d2;
  --proj-color-btn-secondary: #f33;

  --proj-color-table-primary: #c3dbc3;
  --proj-color-table-secondary: #fff;
  --proj-color-table-tertiary: #e8e8e8;
  --proj-color-table-border: #fff;


  --proj-color-font-primary: #636363;
  --proj-color-font-secondary: #3483fa;
  --proj-color-font-tertiary: #fff;
  --proj-color-font-quaternary: #000;
  --proj-color-font-quinternary: #074307;

  --proj-color-font-placeholder: #d9d9d9;

  --proj-color-error: #f33;
}
```

src > index.css > ...

```
3   :root {
4     --proj-color-bg-primary: #e8e8e8;
5     --proj-color-bg-secondary: #074307;
6     --proj-color-bg-tertiary: #cac7c7;
7     --proj-color-bg-quaternary: #c9d9f2;
8     --proj-color-bg-quinternary: #b8c6b9;
9
10    --proj-color-card-bg: #fff;
11    --proj-color-card-border: #d9d9d9;
12
13    --proj-color-btn-primary: #2571d2;
14    --proj-color-btn-secondary: #f33;
15
16    --proj-color-table-primary: #c3dbc3;
17    --proj-color-table-secondary: #fff;
18    --proj-color-table-tertiary: #e8e8e8;
19    --proj-color-table-border: #fff;
20
21
22    --proj-color-font-primary: #636363;
23    --proj-color-font-secondary: #3483fa;
24    --proj-color-font-tertiary: #fff;
25    --proj-color-font-quaternary: #000;
26    --proj-color-font-quinternary: #074307;
27
28    --proj-color-font-placeholder: #d9d9d9;
29
30    --proj-color-error: #f33;
31  }
```

GERAL

```
* {
  box-sizing: border-box;
  margin: 0;
  padding: 0;
  font-family: "Open Sans";
}

a,
a:hover {
  text-decoration: none;
  color: unset;
}

html,
body {
  background-color: var(--pag-color-bg-primary);
}

main {
  padding: 0 20px;
}
```



ESTILOS GENÉRICOS

```
/*----------------------------------------------------------------*/
/* generic styles */

.proj-container {
  width: 100%;
  max-width: 960px;
  margin: 0 auto;
}

.proj-card {
  background-color: #fff;
  border-radius: 10px;
  height: 300px;
}

.proj-mb20 {
  margin-bottom: 20px;
}

.proj-mb40 {
  margin-bottom: 40px;
}

.proj-mt20 {
  margin-top: 20px;
}

.proj-mt40 {
  margin-top: 40px;
}

.proj-section-title {
  text-align: center;
  color: var(--proj-color-font-primary);
  font-size: 16px;
}

.proj-txt-left {
  text-align: left;
}

@media (min-width: 576px) {
  .proj-section-title {
    text-align: left;
    font-size: 24px;
```

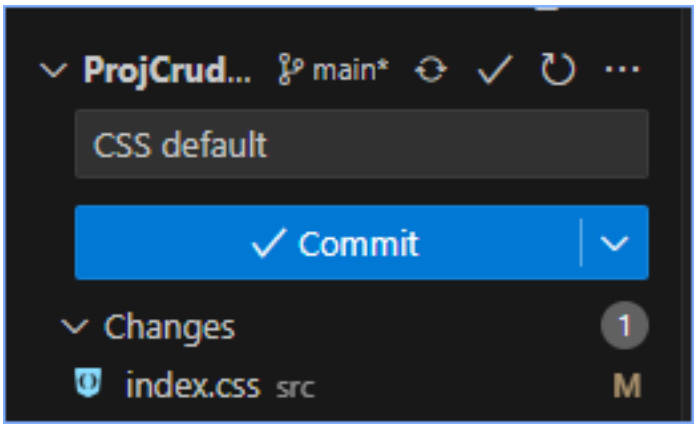
```
  }
   .proj-card {
    height: 550px;
  }
}
```

src > index.css > ...

```
56  /* generic styles */
57
58  .proj-container {
59    width: 100%;
60    max-width: 960px;
61    margin: 0 auto;
62  }
63
64  .proj-card {
65    background-color: #fff;
66    border-radius: 10px;
67    height: 300px;
68  }
69
70  .proj-mb20 {
71    margin-bottom: 20px;
72  }
73
74  .proj-mb40 {
75    margin-bottom: 40px;
76  }
77
78  .proj-mt20 {
79    margin-top: 20px;
80  }
81
82  .proj-mt40 {
83    margin-top: 40px;
84  }
85
86  .proj-section-title {
87    text-align: center;
88    color: var(--proj-color-font-primary);
89    font-size: 16px;
90  }
91
92  .proj-txt-left {
93    text-align: left;
94  }
95
96  @media (min-width: 576px) {
97    .proj-section-title {
98      text-align: left;
99      font-size: 24px;
100   }
101   .proj-card {
102     height: 550px;
103   }
104 }
```

VISUALIZAÇÃO WEB

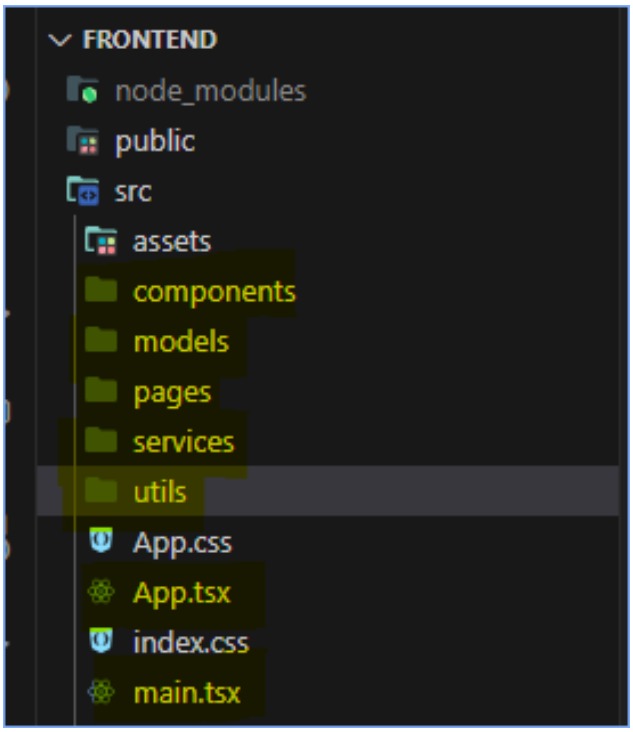
GitHub-3



IMPLEMENTAR A ESTRUTURA DO PROJETO

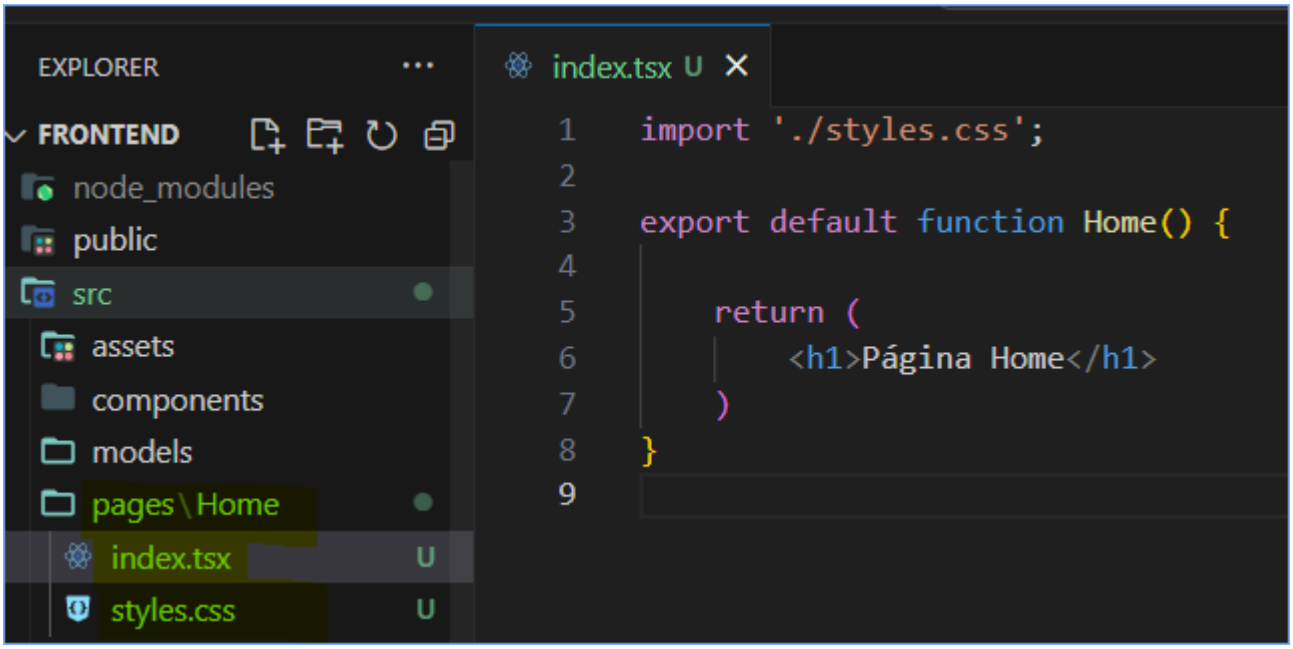
- **PAGES:** Concentrará todas as páginas do projeto que necessitem de rotas.
- **COMPONENTS:** Concentrará todos os componentes que irão integrar as páginas.
- **MODELS:** Todos os modelos de objetos ficarão armazenados nesta pasta.
- **SERVICES:** Onde será criada toda a regra de sincronismo com o backend para o CRUD.
- **UTILS:** Concentrará todos os utilitários que serão utilizados neste projeto.

***NOTA**: As Rotas iremos concentrar no main.tsx e no App.tsx*

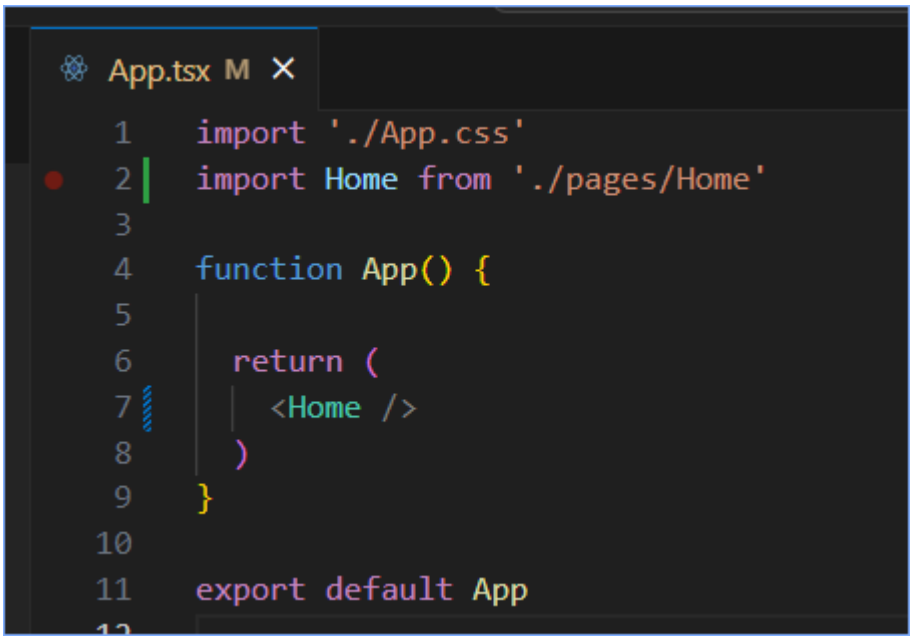


CRIAR A ESTRUTURA BASE DAS PAGINAS DEFAULT

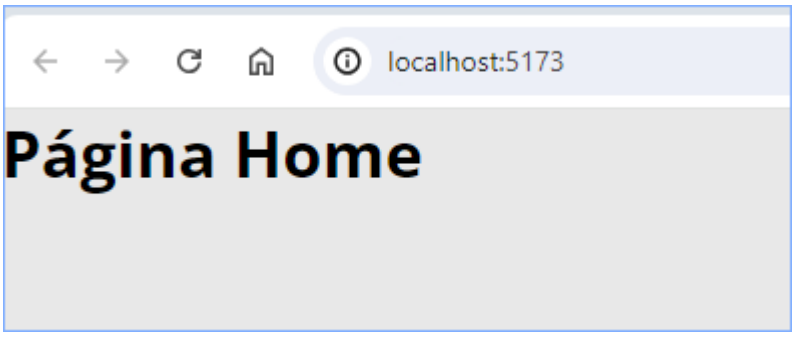
Home



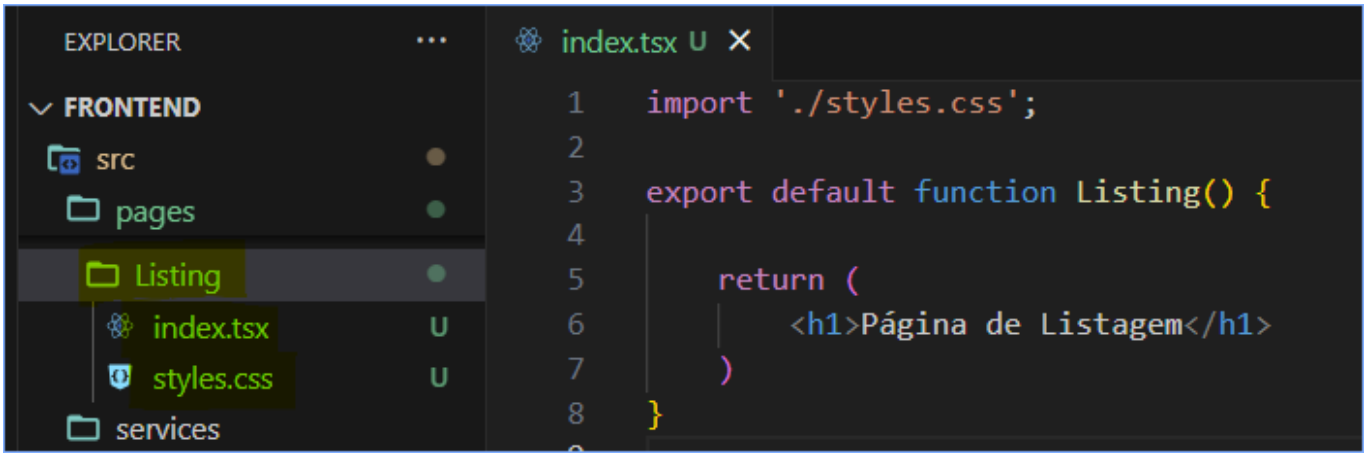
Renderizar a página no App.tsx



Visualização web



Listing

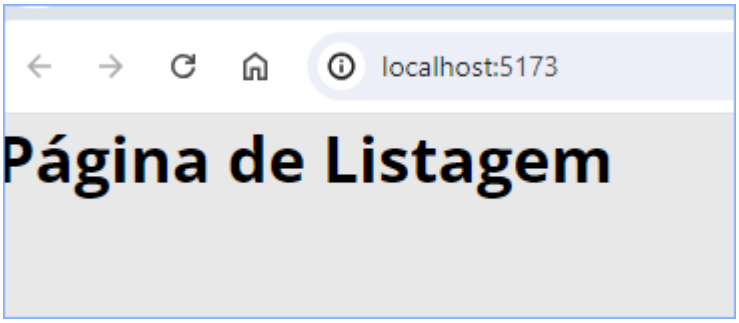


Renderizar a página no App.tsx

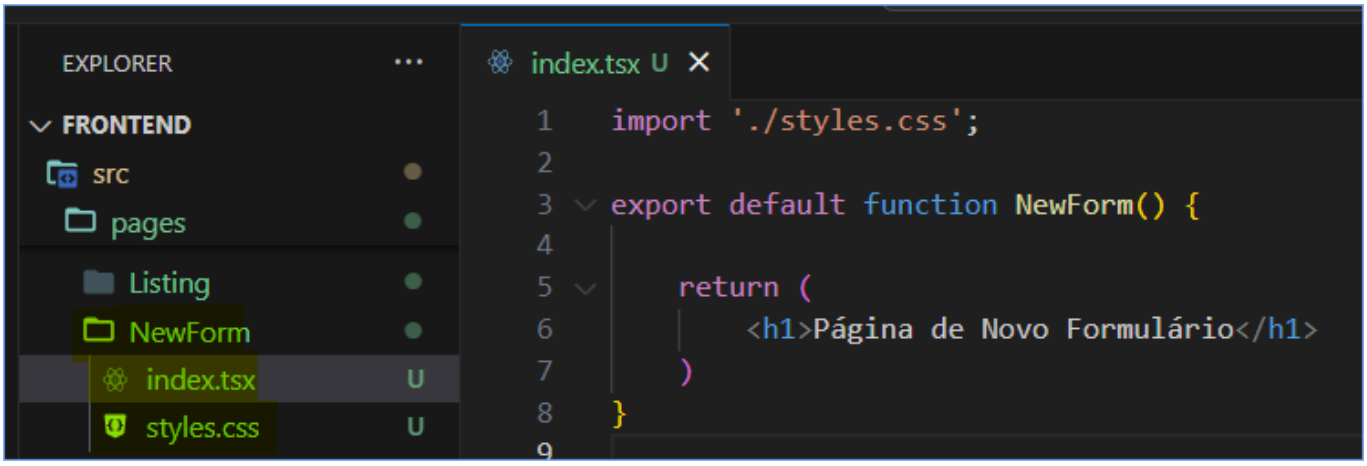
App.tsx M

```tsx
import './App.css'
import Listing from './pages/Listing'

function App() {

  return (
    <Listing />
  )
}

export default App

```

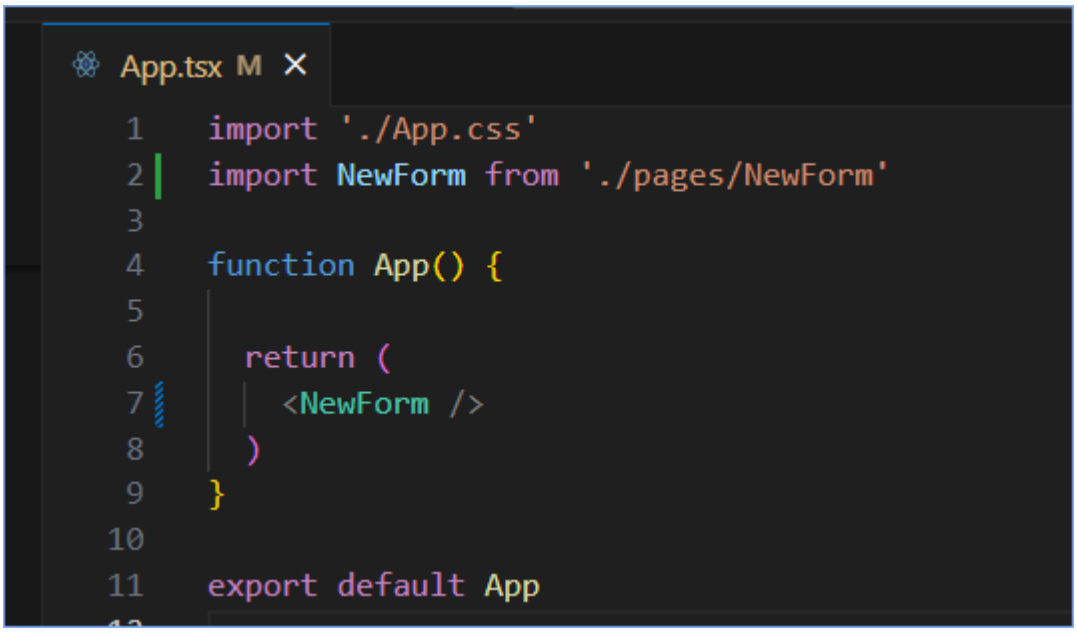
Visualização web



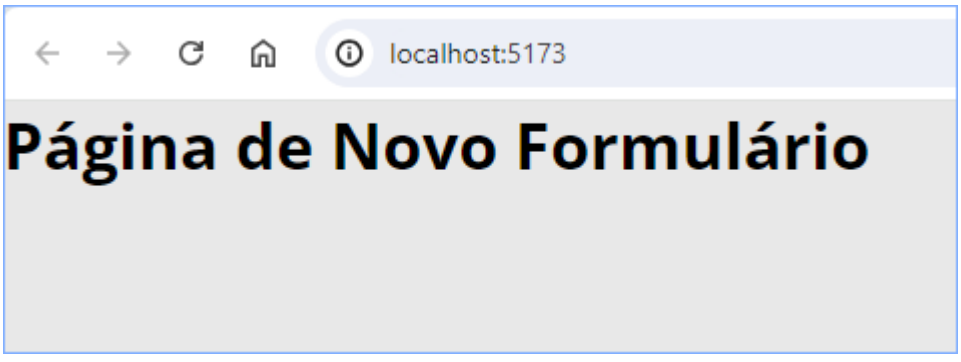
NewForm



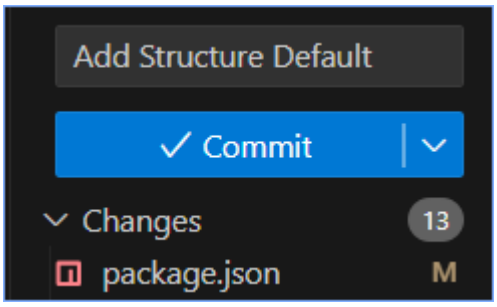
Renderizar a página no App.tsx



Visualização web



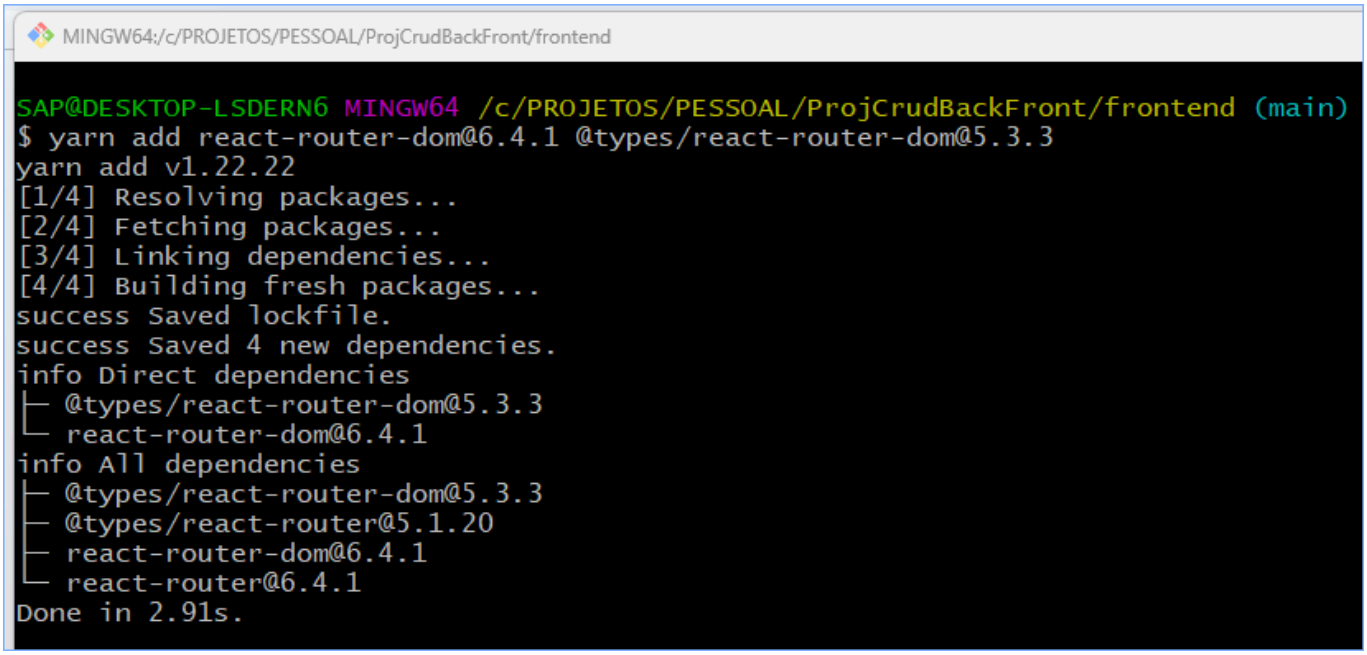
GitHub-4



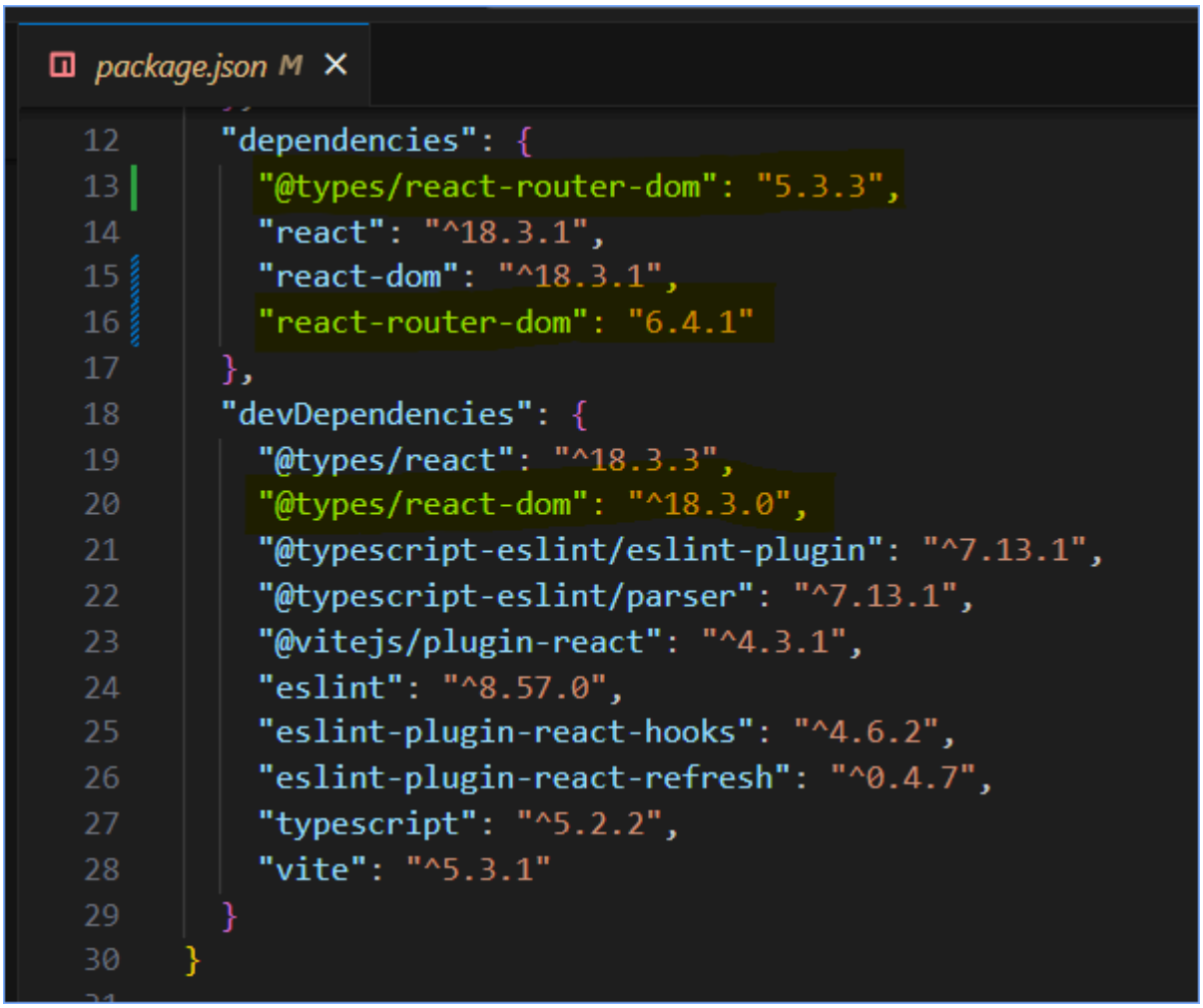
ROTAS

INSTALAR O REACT-ROUTER-DOM

- yarn add react-router-dom@6.4.1 @types/react-router-dom@5.3.3



VERIFICAR NO PROJETO



IMPLEMENTAR AS ROTAS NO APP.TSX

App.tsx M

```tsx
import './App.css'
import { BrowserRouter, Navigate, Route, Routes } from 'react-router-dom'
import Home from './pages/Home'
import Listing from './pages/Listing'
import NewForm from './pages/NewForm'

function App() {

  return (

    <BrowserRouter>
    <Routes>
      <Route path='/' element={<App />} />
      <Route index element={<Home />} />
      <Route path='listings' element={<Listing />}/>
      <Route path='Listings/:studentId' element={<NewForm />}/>
      <Route path='*' element={<Navigate to='/' />}/>
    </Routes>
  </BrowserRouter>

  )
}

export default App
```

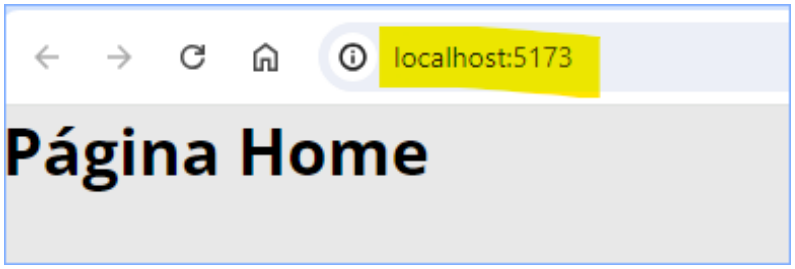
VERIFICAR O MAIN.TSX

main.tsx M

```tsx
import './index.css'
import App from './App.tsx'
import ReactDOM from 'react-dom/client'

ReactDOM.createRoot(document.getElementById('root')!).render(

  <App />

)
```

VISUALIZAÇÃO WEB

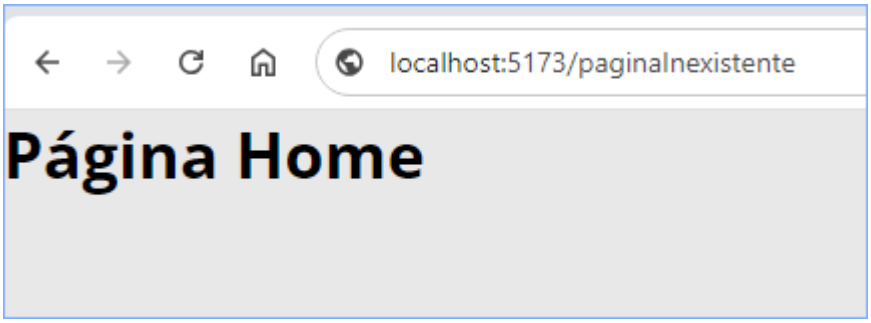
Home



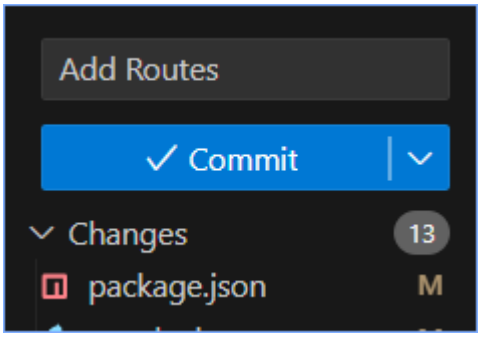
Listing

NewForm

***NOTA**: Ao digitar uma página inexistente, será redirecionada para a home.*



GitHub-5



CRIAR O COMPONENTE HEADER

IMPLEMENTAR O INDEX

index.tsx U

```tsx
import './styles.css';

export default function Header() {

    return (
        <header className='proj-header'>
            <nav className='proj-container'>
                <div className='proj-header-content'>

                    <h3>ProjCrudFront</h3>

                    <div>
                        <a href='#'>GitProject</a>
                    </div>
                </div>

            </nav>
        </header>
    );
}
```

ESTILIZAR

styles.css M

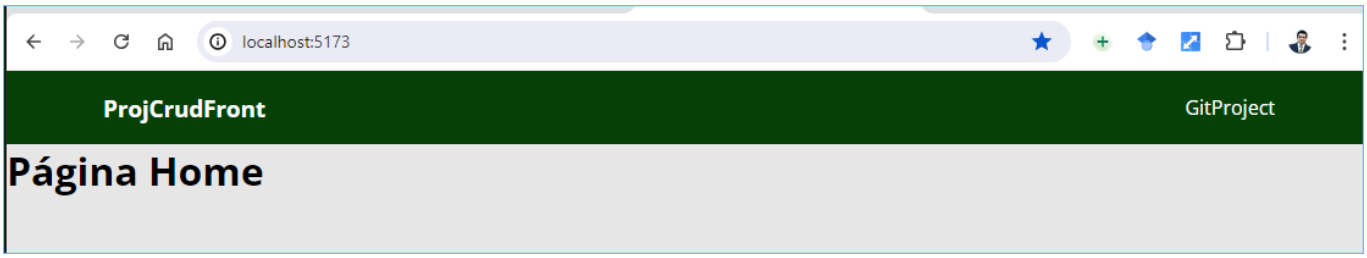
```css
.proj-header {
    background-color: var(--proj-color-bg-secondary);
    color: var(--proj-color-font-tertiary);
    height: 60px;
    display: flex;
    align-items: center;

}

.proj-header-content {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 30px;
}
```

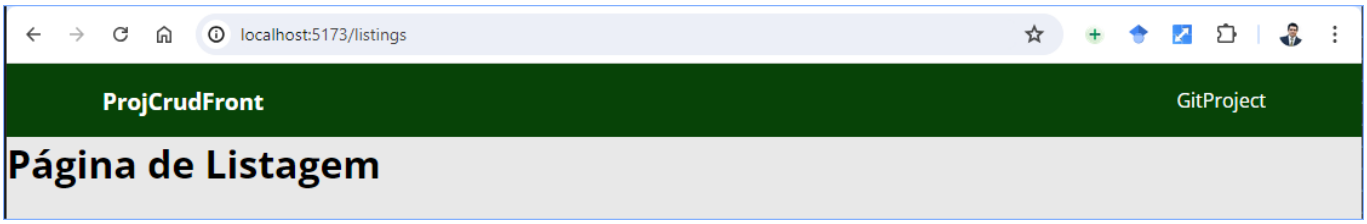
RENDERIZAR O HEADER NO APP.TSX

App.tsx M

```tsx
function App() {

  return (

    <BrowserRouter>

    <Header />

    <Routes>
      <Route path='/' element={<App />} />
      <Route index element={<Home />} />
      <Route path='listings' element={<Listing />}/>
      <Route path='Listings/:studentId' element={<NewForm />}/>
      <Route path='*' element={<Navigate to='/' />}/>
    </Routes>
  </BrowserRouter>

  )
}

export default App
```

VISUALIZAÇÃO WEB



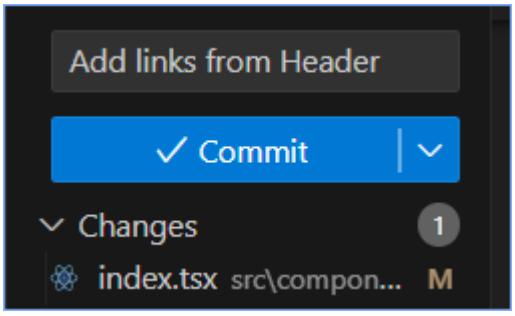


IMPLEMENTAR OS LINKS DO HEADER

```tsx
index.tsx U
    return (
        <header className='proj-header'>
            <nav className='proj-container'>
                <div className='proj-header-content'>
                    <Link to="/">
                        <h3>ProjCrudFront</h3>
                    </Link>

                    <div>
                        <a href='https://github.com/auriceliof/pessoal-ProjCrudBackFront'>GitProject</a>
                    </div>
                </div>
```

***NOTA**: Ao clicar no "ProjCrudFront" a página é redirecionada para o home. Ao clicar no "GitProject", será direcionado para a página do projeto no GitHub.*

GitHub-6



CRIAR O COMPONENTE BUTTON_PRIMARY

IMPLEMENTAR O INDEX

index.tsx U

```tsx
import './styles.css';

type Props = {
    name: string;
}

export default function ButtonPrimary({name}: Props) {

    return (
        <button className='proj-btn-primary'>
            {name}
        </button>
    );
}
```

ESTILIZAR

styles.css U

```css
.proj-btn-primary {
    display: flex;
    justify-content: center;
    font-size: 1rem;
    font-weight: bold;
    background-color: var(--proj-color-btn-primary);
    color: var(--proj-color-font-tertiary);
    width: 150px;
    padding: 5px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}
```

CRIAR O COMPONENTE BUTTON_SECONDARY

IMPLEMENTAR O INDEX

src > components > ButtonSecondary > index.tsx > ...

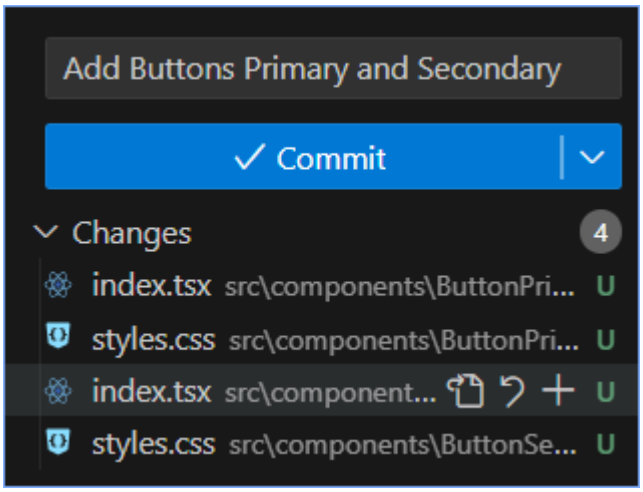
```tsx
import './styles.css';

type Props = {
    name: string;
}

export default function ButtonSecondary({name}: Props) {

    return (
        <button className='proj-btn-secondary'>
            {name}
        </button>
    );
}
```

ESTILIZAR

src > components > ButtonSecondary > styles.css > ...

```css
.proj-btn-secondary {
    display: flex;
    justify-content: center;
    font-size: 1rem;
    font-weight: bold;
    background-color: var(--proj-color-btn-secondary);
    color: var(--proj-color-font-tertiary);
    width: 150px;
    padding: 5px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}
```

GitHub-7


Add Buttons Primary and Secondary
✓ Commit
Changes
4
index.tsx src\components\ButtonPri... U
styles.css src\components\ButtonPri... U
index.tsx src\component... U
styles.css src\components\ButtonSe... U

PÁGINA HOME

IMPLEMENTAR O INDEX

src > pages > Home > index.tsx > ...

```tsx
import './styles.css';

export default function Home() {

    return (
        <main>
            <section className='proj-container'>
                <div className='proj-mt20 proj-card'>
                    <div className="proj-home-content">
                        <h1>Bem-vindo ao projeto CRUD no frontend!!!</h1>
                    </div>
                    <div className="proj-home-content">
                        <h5>Clique no botão abaixo para iniciar</h5>
                    </div>
                </div>
            </section>
        </main>
    )
}
```

ESTILIZAR

src > pages > Home > styles.css > ...

```css
.proj-home-content {
    display: flex;
    justify-content: center;
}

.proj-home-content h1 {
    margin-top: 70px;
}

.proj-home-content h5 {
    margin-top: 60px;
    color: var(--proj-color-font-primary);
}
```

VISUALIZAÇÃO WEB

CHAMADO O COMPONENTE BUTTON_PRIMARY

src > pages > Home > index.tsx > ...

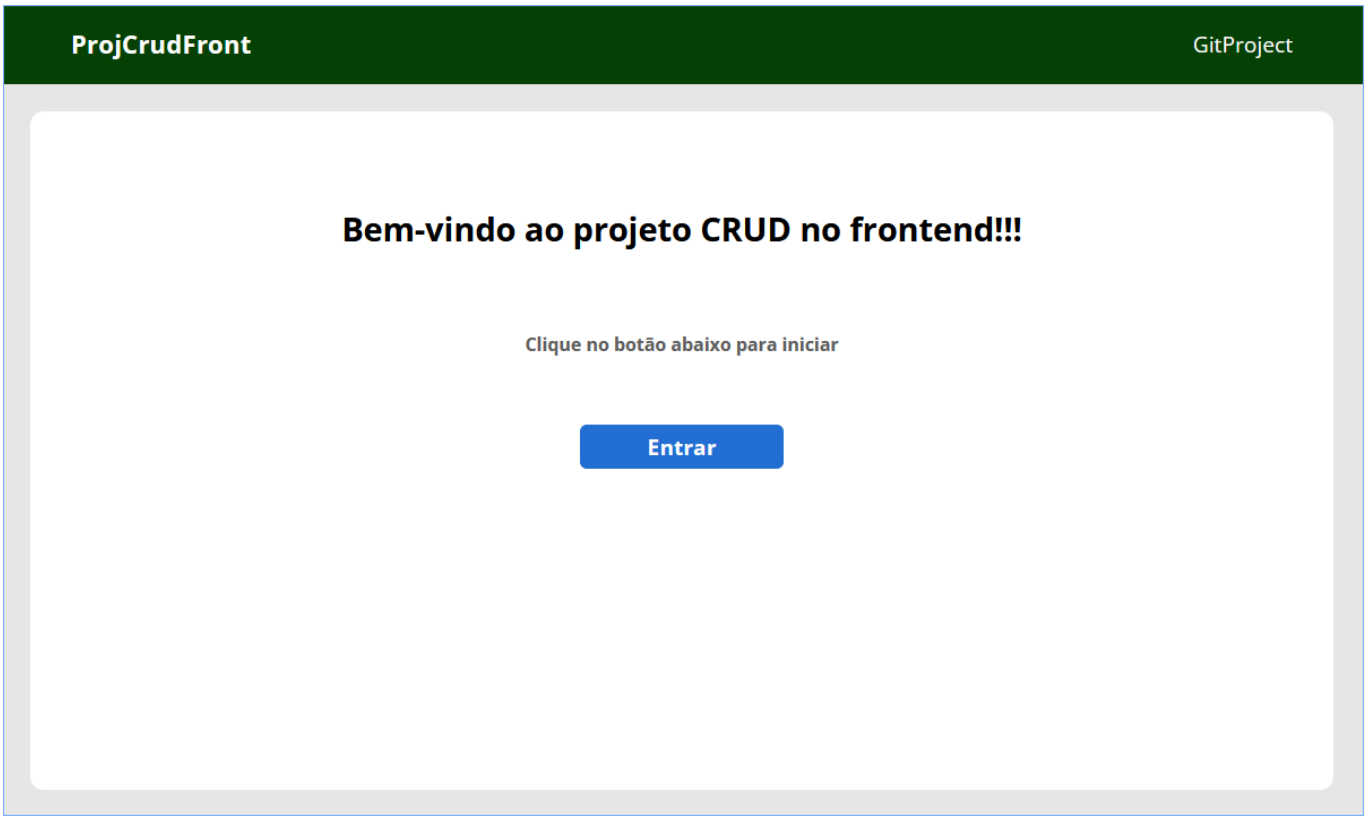
```tsx
    return (
        <main>
            <section className='proj-container'>
                <div className='proj-mt20 proj-card'>
                    <div className="proj-home-content">
                        <h1>Bem-vindo ao projeto CRUD no frontend!!!</h1>
                    </div>
                    <div className="proj-home-content">
                        <h5>Clique no botão abaixo para iniciar</h5>
                    </div>
                    <div className='proj-home-btn'>
                        <Link to="/listings">
                            <ButtonPrimary name='Entrar'/>
                        </Link>
                    </div>
                </div>
            </section>
        </main>
    )
}
```

ESTILIZAR

src > pages > Home > styles.css > ...

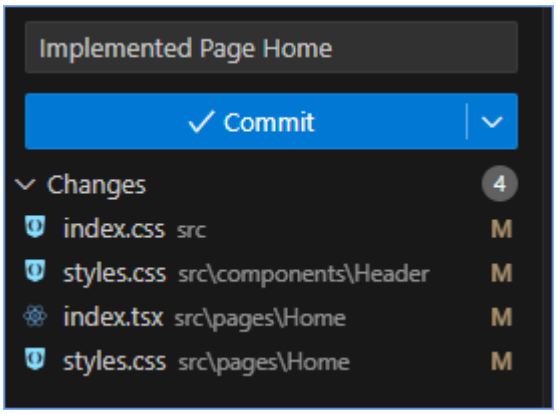
```css
.proj-home-btn {
    display: flex;
    justify-content: center;
    margin-top: 50px;
}
```

VISUALIZAÇÃO WEB



***NOTA**: Ao clicar no botão "Entrar", será direcionada para a "Página de Listagem".*

GitHub-8



PÁGINA LISTING

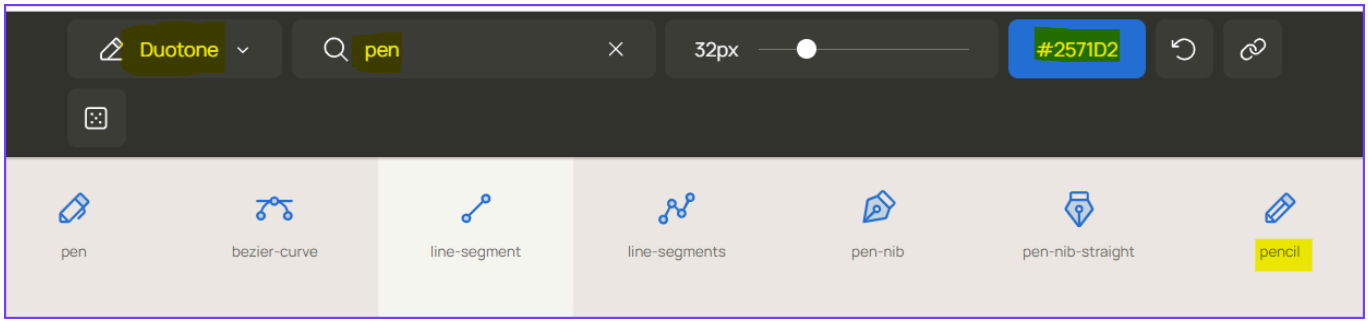
BAIXAR ICONES

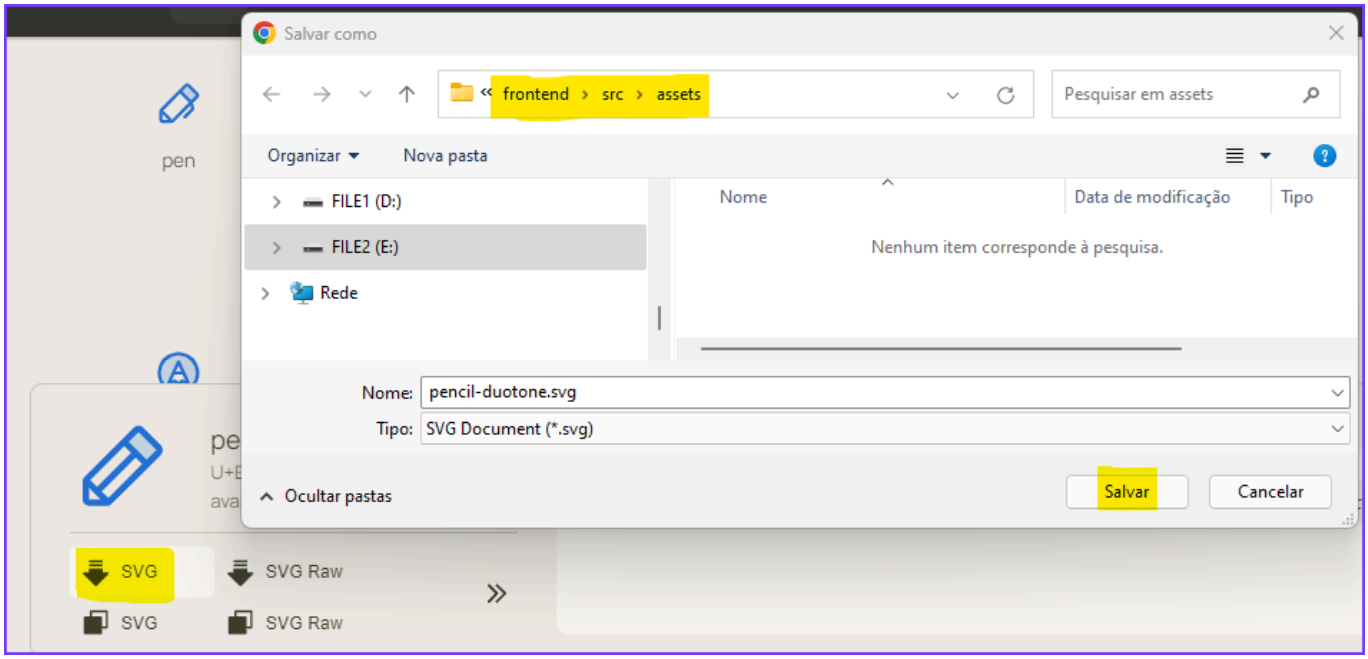
***NOTA**: Em nosso projeto, colocaremos dois ícones, um lápis e uma lixeira.*

- ***Link:*** *https://phosphoricons.com/*

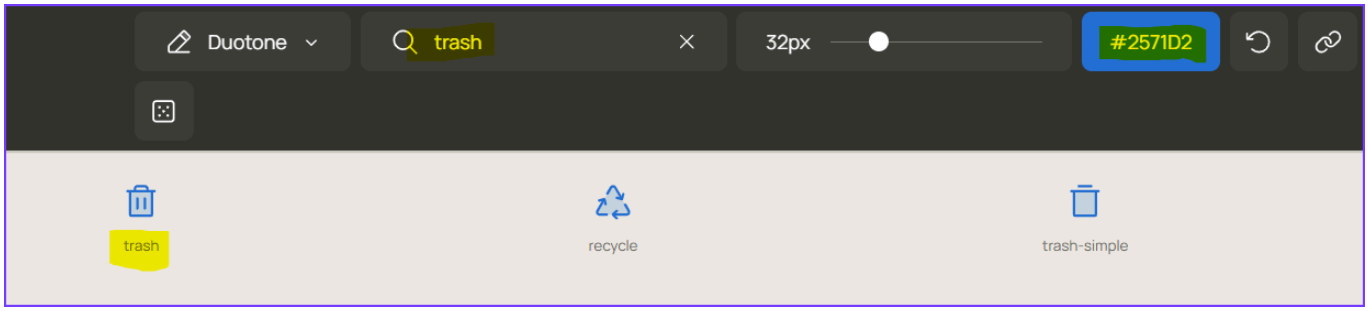
***NOTA**: Baixar os ícones em formato SVG e salvar no diretório do projeto, em assets.*

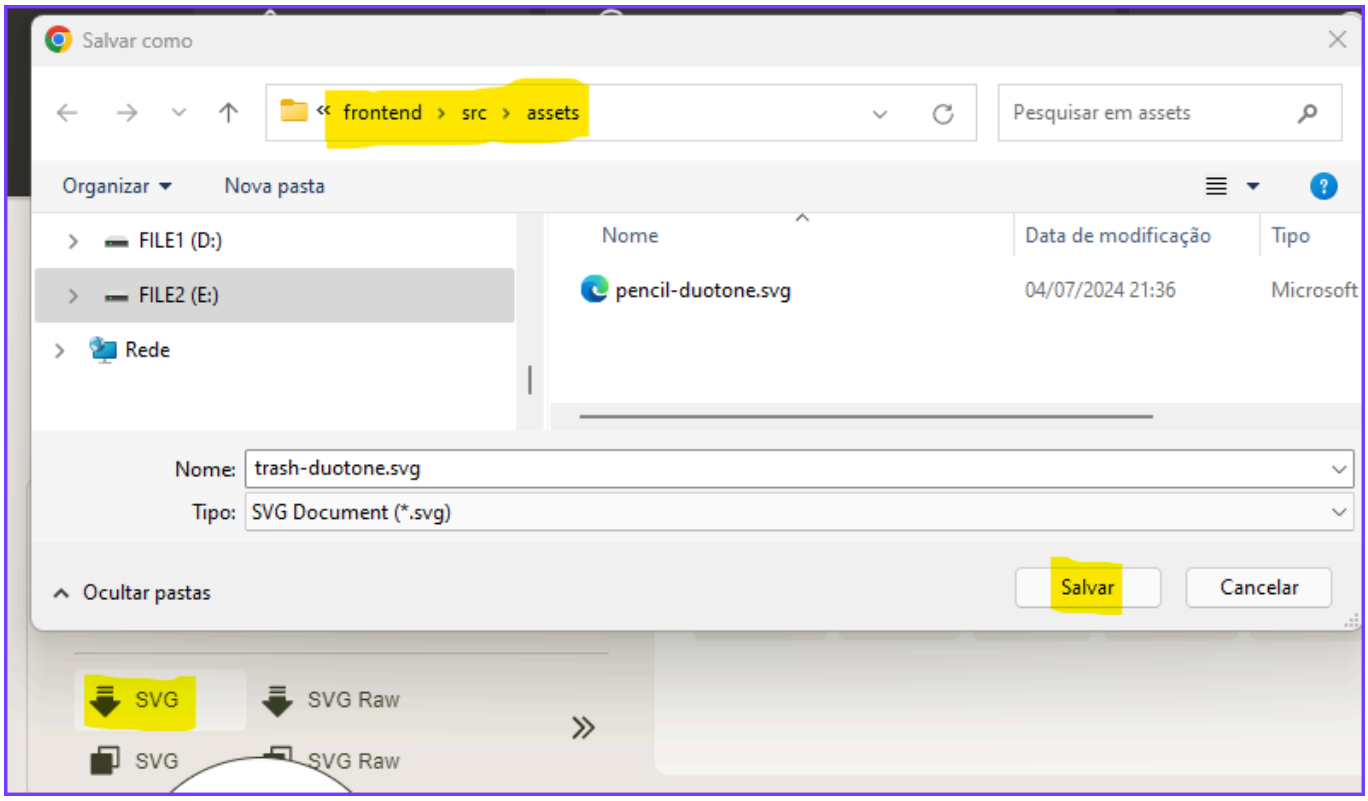
Lápis

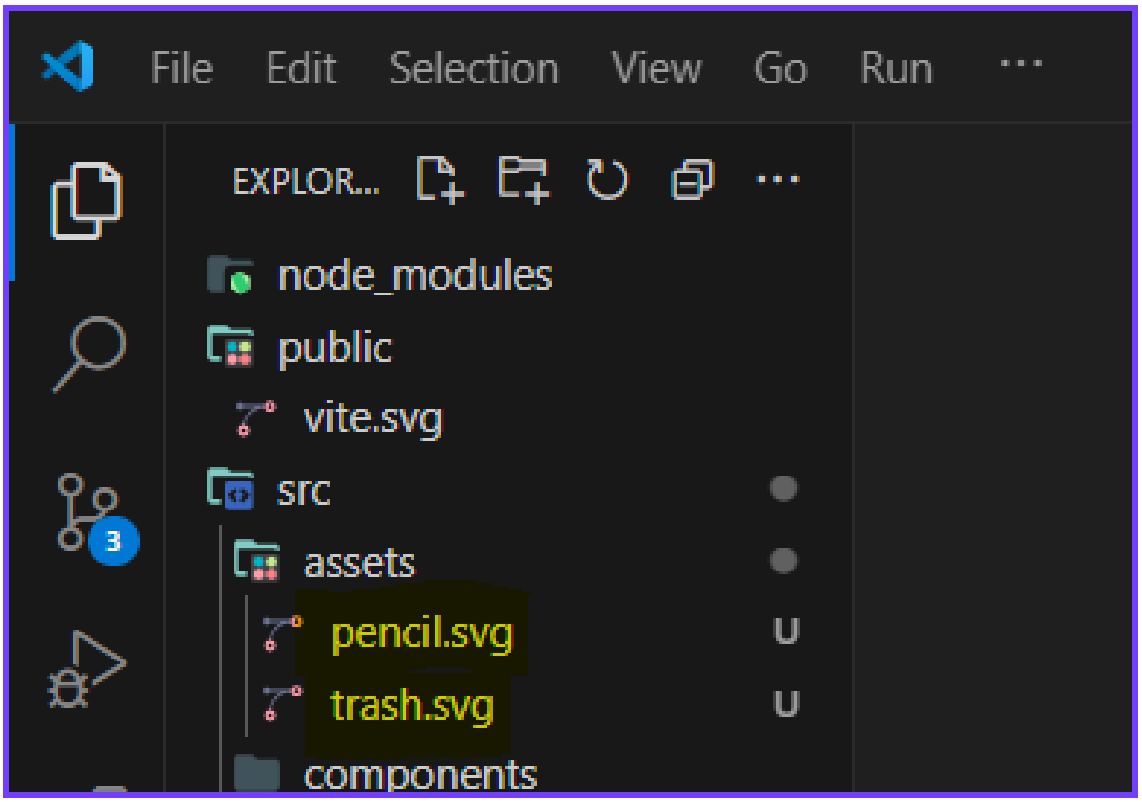




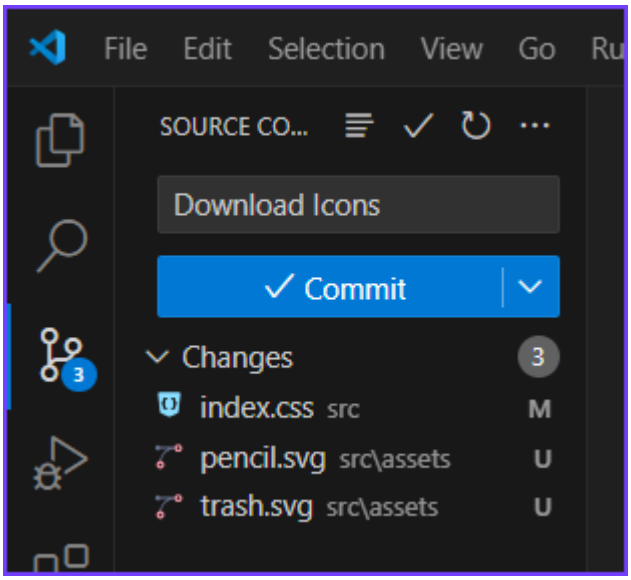
Lixeira


Duotone
trash
32px
#2571D2
trash
recycle
trash-simple


Salvar como
frontend › src › assets
Pesquisar em assets
Organizar
Nova pasta
FILE1 (D:)
FILE2 (E:)
Rede
Nome
Data de modificação
Tipo
pencil-duotone.svg
04/07/2024 21:36
Microsoft
Nome: trash-duotone.svg
Tipo: SVG Document (*.svg)
Ocultar pastas
Salvar
Cancelar
SVG
SVG Raw
SVG
SVG Raw


File Edit Selection View Go Run ···
EXPLOR...
node_modules
public
vite.svg
src
assets
pencil.svg
trash.svg
components
3
U
U

GitHub-9


File Edit Selection View Go Ru
SOURCE CO...
Download Icons
✓ Commit
Changes
3
index.css src
M
pencil.svg src\assets
U
trash.svg src\assets
U

LAYOUT DA LISTAGEM DE ALUNOS

Implementar o Index

src > pages > Listing > index.tsx > ...

```tsx
import './styles.css';
import editIcon from '../../assets/pencil.svg';
import deleteIcon from '../../assets/trash.svg';
import ButtonPrimary from '../../components/ButtonPrimary';

export default function Listing() {

    return (
        <main>
            <section id="proj-listing-section" className="proj-container">
                <div>
                    <h2 className="proj-mt20">Listagem de Alunos</h2>
                </div>

                <div className="proj-mt40 proj-mb20">
                    <div className="proj-listing-btn">
                        <ButtonPrimary name='Novo' />
                    </div>
                </div>

                <table className="proj-listing-table proj-mb20 proj-mt20">
                    <thead>
                        <tr>
                            <th className='proj-listing-table-id'>ID</th>
                            <th>NOME</th>
                            <th>CPF</th>
                            <th>NASCIMENTO</th>
                            <th>RENDA</th>
                            <th></th>
                            <th></th>
                        </tr>
                    </thead>
```

```
                    <tbody>
                        <tr>
                            <td className='proj-listing-table-id'>1</td>
                            <td>Auricelio Freitas</td>
                            <td>123.456.789-00</td>
                            <td>28/08/1982</td>
                            <td>R$ 10999.0</td>
                            <td><img src={editIcon} alt='Editar'/></td>
                            <td><img src={deleteIcon} alt='Deletar'/></td>
                        </tr>
                        <tr>
                            <td className='proj-listing-table-id'>2</td>
                            <td>Teste1 Sobrenome</td>
                            <td>123.456.789-00</td>
                            <td>08/06/1990</td>
                            <td>R$ 1999.0</td>
                            <td><img src={editIcon} alt='Editar'/></td>
                            <td><img src={deleteIcon} alt='Deletar'/></td>
                        </tr>
                        <tr>
                            <td className='proj-listing-table-id'>3</td>
                            <td>Teste2 Sobrenome</td>
                            <td>123.456.789-00</td>
                            <td>08/06/1990</td>
                            <td>R$ 12999.0</td>
                            <td><img src={editIcon} alt='Editar'/></td>
                            <td><img src={deleteIcon} alt='Deletar'/></td>
                        </tr>
                        <tr>
                            <td className='proj-listing-table-id'>4</td>
                            <td>Teste3 Sobrenome</td>
                            <td>123.456.789-00</td>
                            <td>08/06/1990</td>
                            <td>R$ 8999.0</td>
                            <td><img src={editIcon} alt='Editar'/></td>
                            <td><img src={deleteIcon} alt='Deletar'/></td>
                        </tr>
                        <tr>
                            <td className='proj-listing-table-id'>5</td>
                            <td>Teste4 Sobrenome</td>
                            <td>123.456.789-00</td>
                            <td>08/06/1990</td>
                            <td>R$ 15999.0</td>
                            <td><img src={editIcon} alt='Editar'/></td>
                            <td><img src={deleteIcon} alt='Deletar'/></td>
                        </tr>
                    </tbody>
                </table>
            </section>
        </main>
    )
}
```

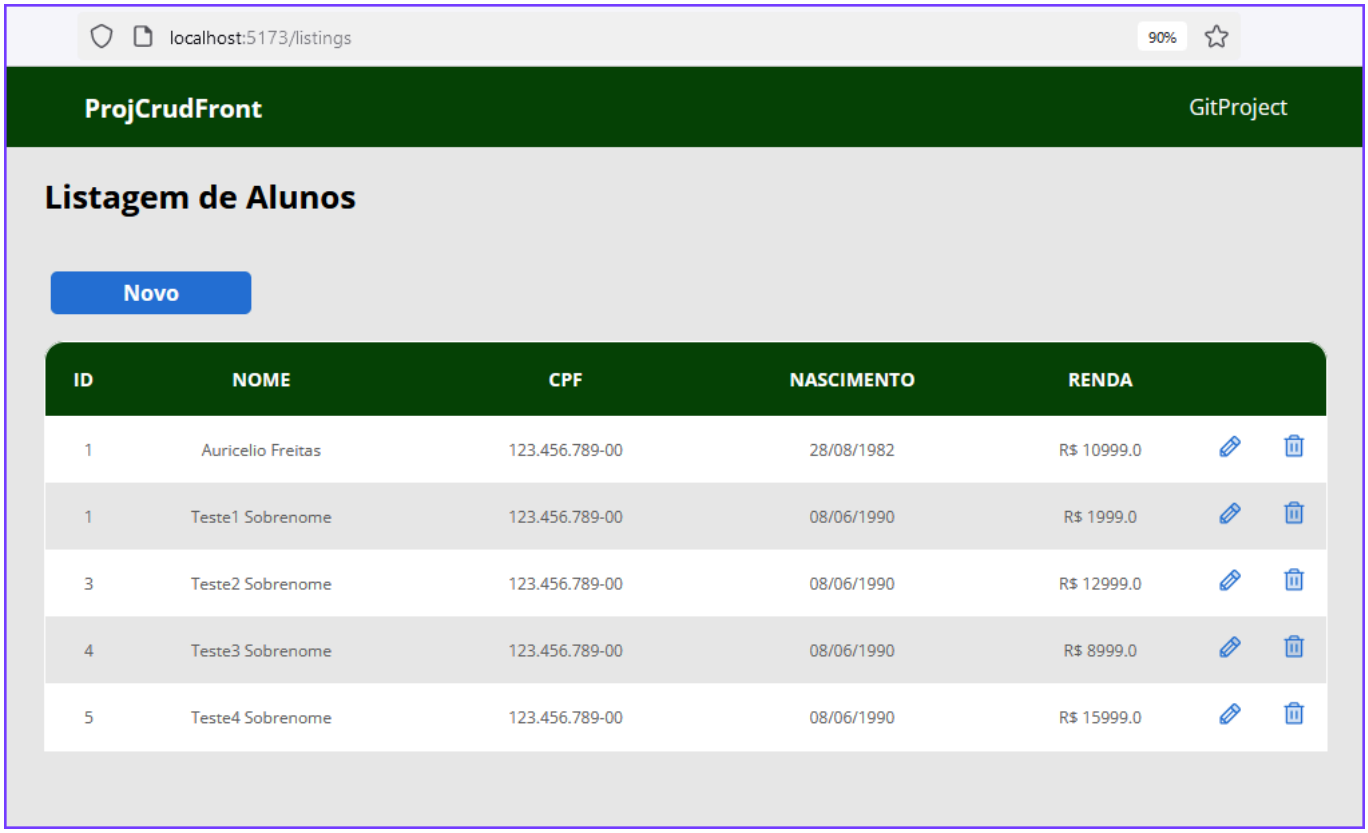
NOTA: *Neste momento, criaremos alunos estáticos para testes.*

Estilizar

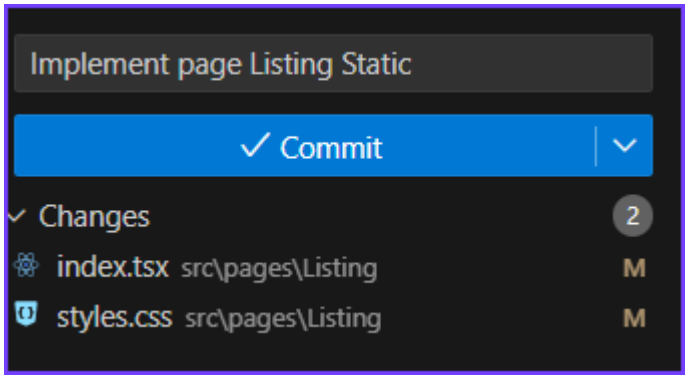
src > pages > Listing > styles.css > ...

```css
.proj-listing-btn {
    cursor: pointer;
    margin: 5px;
    width: 18%;
    font-size: 10px;
}

.proj-listing-table {
    width: 100%;
    background-color: var(--proj-color-table-primary);
    border-spacing: 0;
    border: 1px solid var(--proj-color-table-border);
    border-radius: 15px 15px 0 0;
}

.proj-listing-table-id {
    text-align: end;
}

.proj-listing-table thead {
    height: 55px;
    font-size: 14px;
    color: var(--proj-color-font-quaternary);
}

.proj-listing-table tbody {
    text-align: center;
    font-size: 12px;
    color: var(--proj-color-font-primary);
    background-color: var(--proj-color-card-bg);
}

.proj-listing-table tbody tr {
    height: 50px;
    border-top: 1px solid var(--proj-color-card-border);
}

.proj-listing-table tbody img {
    height: 20px;
    cursor: pointer;
}

.proj-listing-table tr:nth-of-type(even) {
    background-color: var(--proj-color-table-tertiary);
}
```

Visualização web



GitHub-10



LISTAGEM DINÂMICA

***NOTA:** Para que o frontend consiga realizar requisições no backend é necessário realizar a liberação do CORS no backend.*

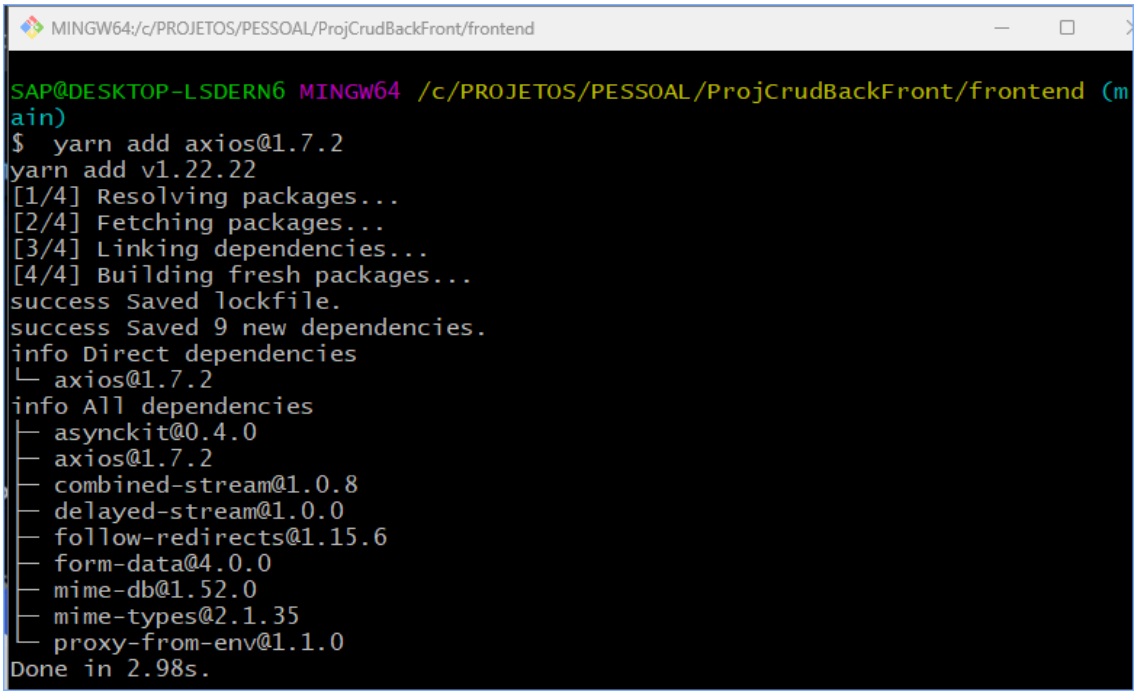
INSTALAR O AXIOS

Verificar versões

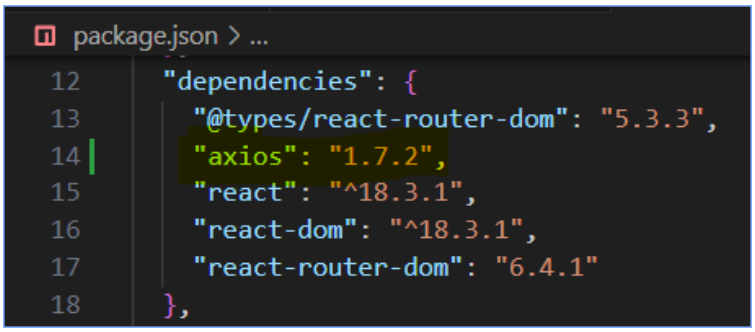
- https://www.npmjs.com/package/axios?activeTab=versions

Adicionar ao projeto

- yarn add axios@1.7.2



Verificar instalação



IMPLEMENTAR O UTILS

Criar o system

```
src > utils > system.ts > ...
1   export const BASE_URL = import.meta.env.VITE_BACKEND_URL ?? "http://localhost:8080";
2
```

NOTA: *Local onde armazenaremos as variáveis de URL.*

Criar o request

```
src > utils > requests.ts > ...
1   import axios, { AxiosRequestConfig } from "axios";
2   import { BASE_URL } from "./system";
3
4   export function requestBackend(config: AxiosRequestConfig) {
5
6       const headers = config.headers
7
8       ? {
9           ...config.headers?.Accept
10      }
11      : config.headers;
12
13      return axios ({ ...config, baseURL: BASE_URL, headers })
14  }
```

NOTA: *Local onde armazenaremos as variáveis de configuração do Axios*

Criar o format

src > utils > format.ts > ...

```ts
// Função para formatar data e hora no formato Brazil "dd/mm/yyyy"
export function formatDateBR(dataHora: any) {
    const data = new Date(dataHora);
    const ano = data.getFullYear();
    const mes = String(data.getMonth() + 1).padStart(2, '0');
    const dia = String(data.getDate() + 1).padStart(2, '0');

    const dataBR = `${dia}/${mes}/${ano}`

        if (dataBR === "31/12/1969") {
            return " "
        }

    return dataBR;
}

```

NOTA: *Local onde armazenaremos as formatações de data/hora no padrão do Brasil.*

IMPLEMENTAR O SERVICE

Criar o student-service

src > services > student-service.ts > ...

```ts
import { AxiosRequestConfig } from "axios";
import { BASE_URL } from "../utils/system";
import { requestBackend } from "../utils/requests";

export function findPageRequest(page: number, name: string, size = 12, sort = "id") {
    const config : AxiosRequestConfig = {
        method: "GET",
        baseURL: BASE_URL,
        url: "/students",
        params: {
            page,
            name,
            size,
            sort
        }
    }

    return requestBackend(config)
}
```

NOTA: *Local onde armazenaremos as variáveis de requisições REST.*

IMPLEMENTAR O MODELS

Criar o students.ts

src > models > students.ts > ...

```ts
export type StudentDTO = {
    id: number;
    name: string;
    cpf: string;
    birthDate: Date;
    income: number;
};
```

NOTA: *Local onde armazenaremos os modelos DTO.*

REALIZAR AS CHAMADAS NO INDEX DO LISTING

Implementar o tipo QueryParams para pegar a pagina atual

src > pages > Listing > index.tsx > ...

```tsx
import './styles.css';
import editIcon from '../../assets/pencil.svg';
import deleteIcon from '../../assets/trash.svg';
import ButtonPrimary from '../../components/ButtonPrimary';
import { useEffect, useState } from 'react';
import * as studentService from "../../services/student-service";
import { StudentDTO } from '../../models/students';
import { formatDateBR } from '../../utils/format';

type QueryParams = {
    page: number;
    name: string;
}
```

Implementar o useState, tipado com o QueryParams

```
src > pages > Listing > index.tsx > ...
14
15  export default function Listing() {
16
17      const [queryParams, setQueryParams] = useState<QueryParams>({
18          page: 0,
19          name: ""
20      });
21
```

Implementar o useState, tipado com o StudentDTO

```
src > pages > Listing > index.tsx > ...
21
22      const [students, setStudents] = useState<StudentDTO[]>([])
23
```

Implementar o useEffect, chamado o Axios

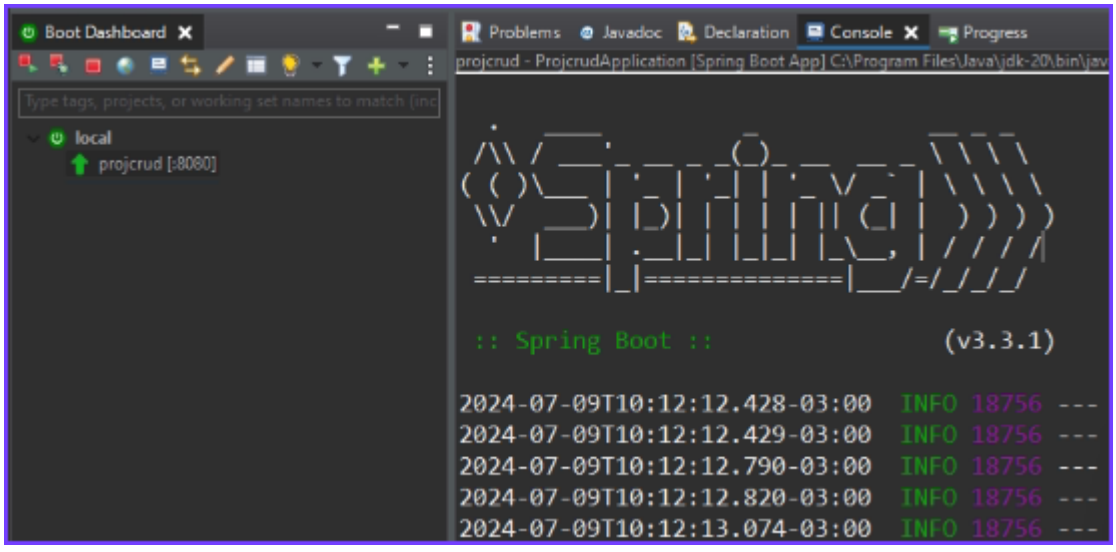
```
src > pages > Listing > index.tsx > ...
25      useEffect(() => {
26          studentService.findPageRequest(queryParams.page, queryParams.name)
27              .then(response => {
28                  setStudents(response.data.content)
29              })
30      }, [])
31
```

Implementar a renderização da tabela dinamicamente

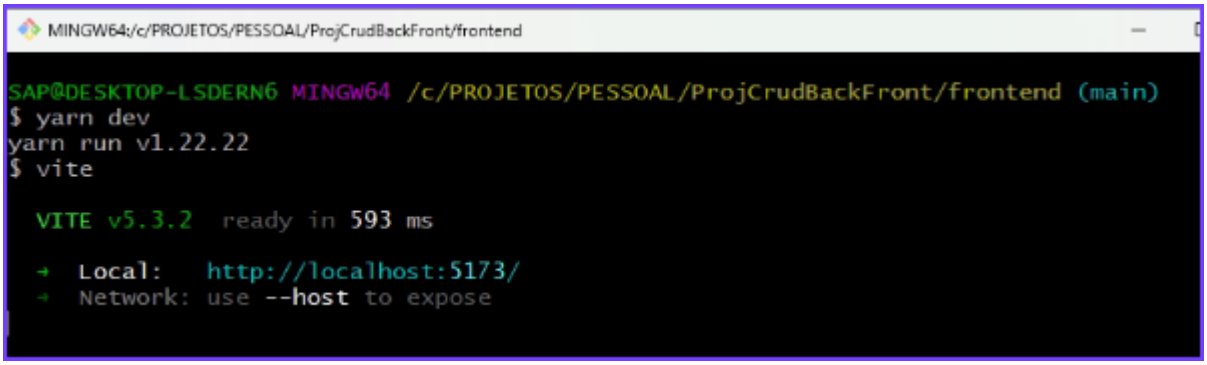
```
src > pages > Listing > index.tsx > ...
                    <tbody>
                        {
                            students.map(student => (
                                <tr key={student.id}>
                                    <td className='proj-listing-table-id'>{student.id}</td>
                                    <td>{student.name}</td>
                                    <td>{student.cpf}</td>
                                    <td>{formatDateBR(student.birthDate)}</td>
                                    <td>R$ {student.income}</td>
                                    <td><img src={editIcon} alt='Editar'/></td>
                                    <td><img src={deleteIcon} alt='Deletar'/></td>
                                </tr>
                            ))
                        }
                    </tbody>
                </table>
            </section>
        </main>
    )
}
```

VISUALIZAÇÃO WEB

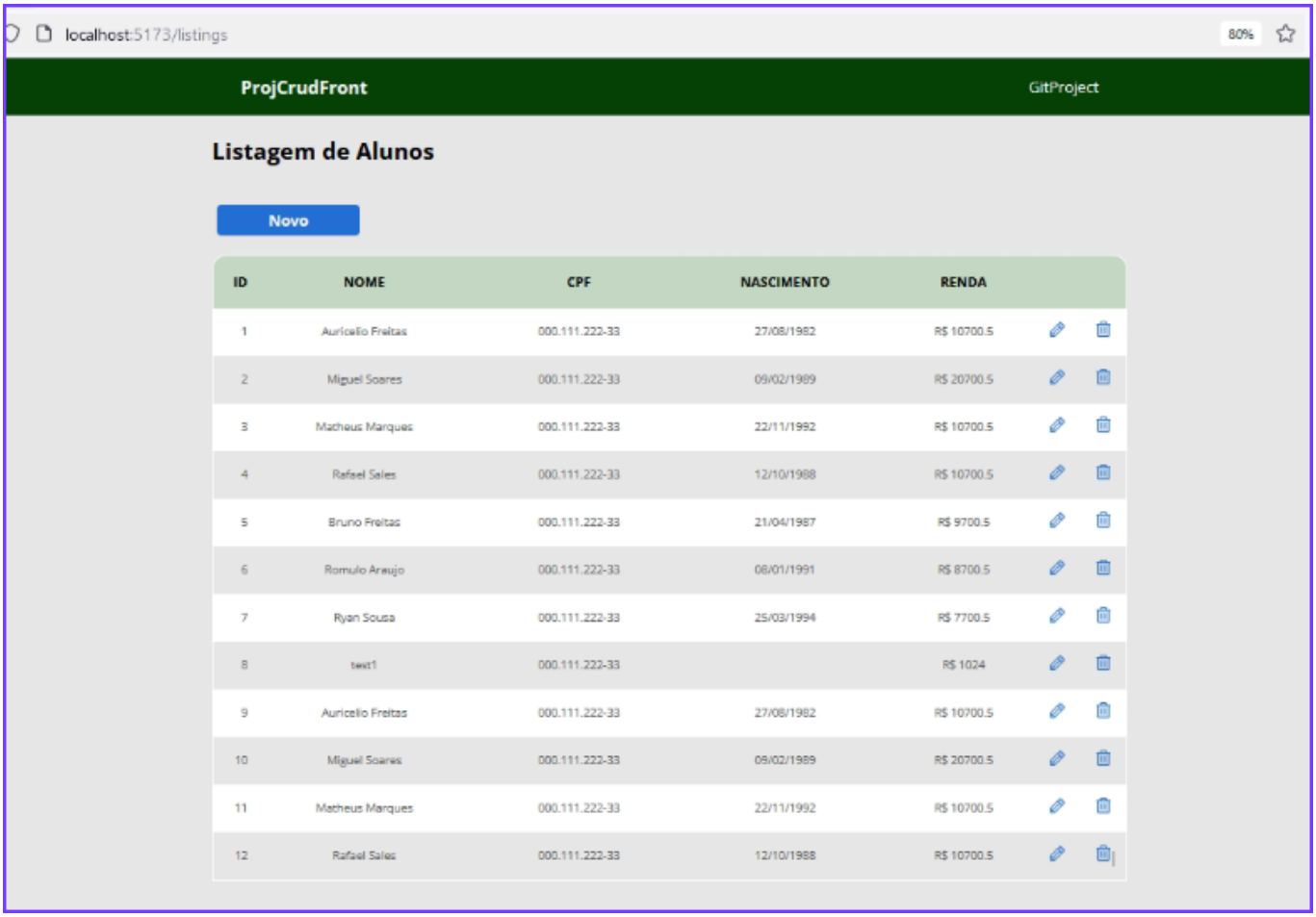
Iniciar o Backend



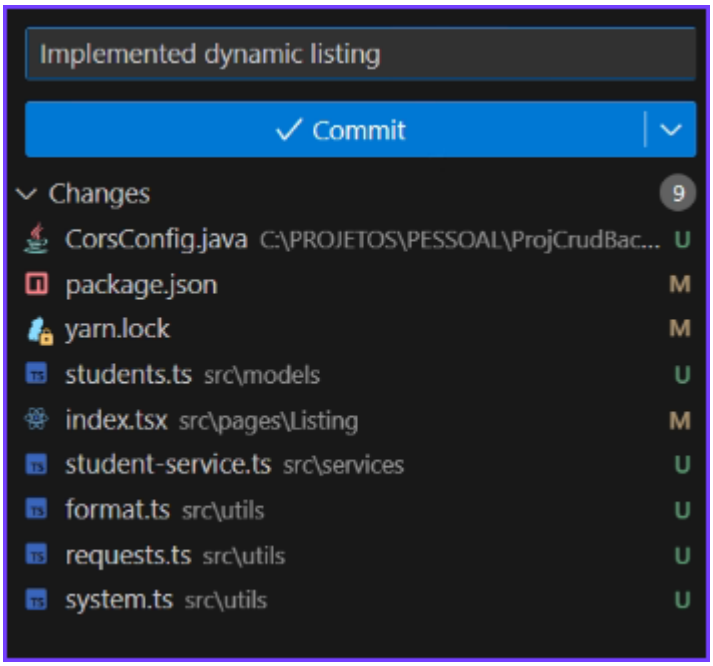
Iniciar o Frontend



Visualizar



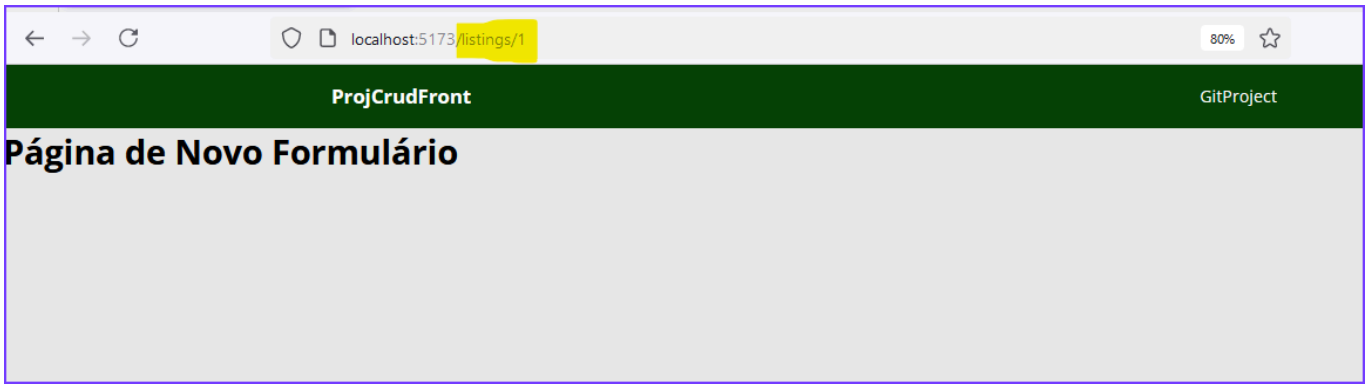
GitHub-11


Implemented dynamic listing
Commit
Changes 9
CorsConfig.java C:\PROJETOS\PESSOAL\ProjCrudBac... U
package.json M
yarn.lock M
students.ts src\models U
index.tsx src\pages\Listing M
student-service.ts src\services U
format.ts src\utils U
requests.ts src\utils U
system.ts src\utils U

PAGINA NEW_FORM

***NOTA**: Esta página será responsável pela INSERÇÃO de novos alunos.*

PROJETO ATÉ AQUI



IMPLEMENTAR O LAYOUT DA PÁGINA NEW_FORM

Index

src > pages > NewForm > index.tsx > ...

```tsx
import './styles.css';
import { Link } from 'react-router-dom';
import ButtonSecondary from '../../components/ButtonSecondary';
import ButtonPrimary from '../../components/ButtonPrimary';

export default function NewForm() {

    return (
        <main>
            <section id="proj-form-section" className="proj-container">
                <div className="proj-form-container">
                    <form className="proj-card proj-form">
                        <h2 className="proj-mb20">Dados do Aluno</h2>
                        <div className="proj-form-controls-container">
                            <div>
                                <h5>Nome</h5>
                                <input className="proj-form-control" type="text" placeholder='Nome'/>
                            </div>
                            <div>
                                <h5>Cpf</h5>
                                <input className="proj-form-control" type="text" placeholder='Cpf'/>
                            </div>
                            <div>
                                <h5>Data de Nascimento</h5>
                                <input className="proj-form-control" type="text" placeholder='Data de Nascimento'/>
                            </div>
                            <div>
                                <h5>Renda</h5>
                                <input className="proj-form-control" type="text" placeholder='Renda'/>
                            </div>

                        </div>

                        <div className="proj-form-buttons">
                            <Link to="/listings">
                                <ButtonSecondary name='Cancelar'/>
                            </Link>
                            <ButtonPrimary name='Salvar' />
                        </div>
                    </form>
                </div>
            </section>
        </main>
    );
}
```

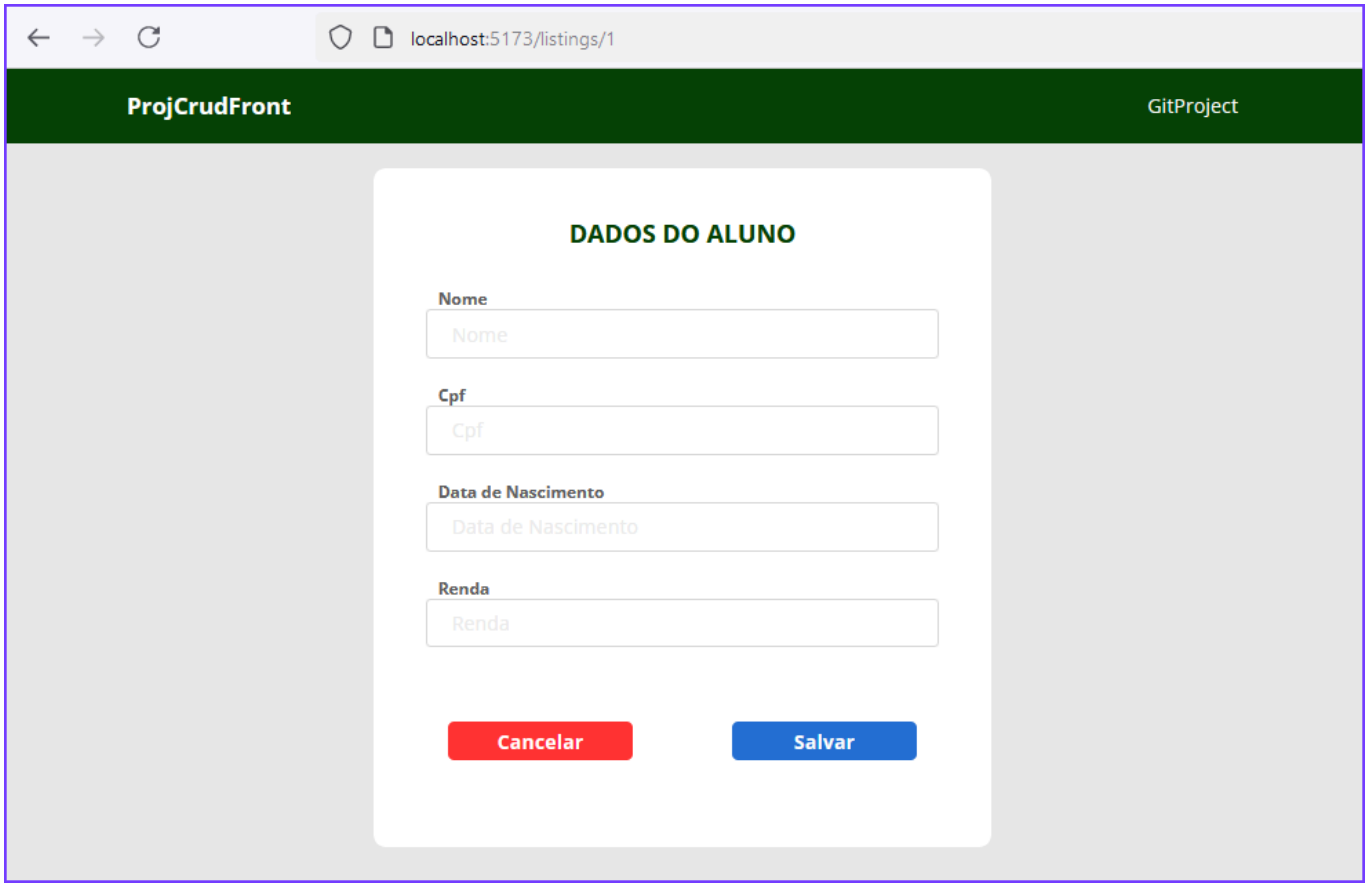
Styles

src > pages > NewForm > styles.css > ...

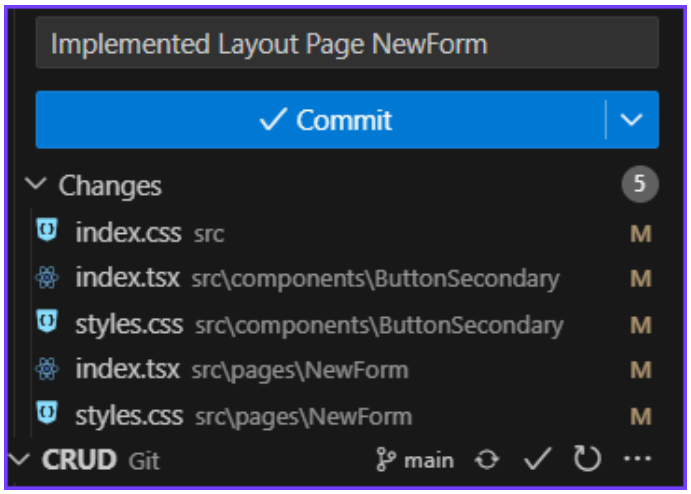
```css
#proj-form-section {
  padding: 20px 0;
}

.proj-form-container {
  width: 100%;
  max-width: 500px;
  margin: 0 auto;
}

.proj-form {
  display: flex;
  flex-direction: column;
  align-items: center;
  padding: 40px 20px;
}

.proj-form h2 {
  font-size: 20px;
  font-weight: bolder;
  color: var(--proj-color-font-quinternary);
  text-transform: uppercase;
  font-weight: bold;
  text-align: center;
}

.proj-form h5 {
  color: var(--proj-color-font-primary);
  margin-left: 10px;
}

.proj-form-controls-container {
  width: 100%;
  display: grid;
  grid-gap: 20px;
  margin: 10px 0;
  padding: 0 22px;
}
```

```
39
40  .proj-form-control {
41    width: 100%;
42    height: 40px;
43    font-size: 16px;
44    padding: 0 20px;
45    color: var(--proj-color-font-primary);
46    border-radius: 4px;
47    border: 1px solid var(--proj-color-card-border);
48  }
49
50  .proj-form-control:focus {
51    outline: none;
52  }
53
54  .proj-form-control::placeholder {
55    color: var(--proj-color-font-placeholder);
56  }
57
58  .proj-form-buttons {
59    width: 100%;
60    display: flex;
61    justify-content: space-around;
62    margin-top: 50px;
63  }
64
```

Visualização web



GitHub-12



ESTRATÉGIA PARA IMPLEMENTAÇÃO DE FORMULÁRIO

NOTA: *Neste projeto iremos realizar a implementação dos formulários manualmente para efeitos de aprendizagem, ou seja, não iremos utilizar biblioteca de terceiros.*

Segue sugestões para implementação de formulários usando biblioteca de terceiros:

- *React Hook Form*
- *Formik*

ESTRATÉGIA

- MANTER UM ESTADO
  MANTER UM ESTADO COM OS DADOS DO FORMULÁRIO, EM UM USE_STATE

- CONTROLES E FUNÇÕES
  IMPLEMENTAR O "FORMS.TS", QUE IRÁ CONTER OS CONTROLES E FUNÇÕES QUE TRANSFORMARÃO OS DADOS DO FORMULÁRIO QUE ESTÃO MANTIDOS NO ESTADO ACIMA.

- COMPONENTES CUSTOMIZADOS PARA OS INPUTS
  UTILIZAÇÃO DO FORM_INPUT PARA CUSTOMIZAÇÃO

- DATASET NOS INPUTS
  ACRESCENTARÁ NO DOM ACIMA, ALGUNS DADOS A MAIS, COMO O CAMPO INVÁLIDO E CAMPO SUJO, QUANDO APLICARMOS AS VALIDAÇÕES

- CSS PERSONALIZADO

  IMPLEMENTAREMOS OS ESTILOS PERSONALIZADOS PARA OS INPUTS E PARA OS ERROS DO FORMULÁRIO

IMPLEMENTAR O COMPONENTE FORM_INPUT

src > components > FormInput > index.tsx > ...

```tsx
export default function FormInput(props: any) {

    const { validation, invalid, ...inputProps } = props;

    return (
        <input { ...inputProps } data-invalid={invalid} />
    );
}
```

IMPLEMENTAR O UTILITÁRIO FORMS

- update

OBJETIVO: Gerar um novo objeto de formulário onde o campo de nome “”name” seja atualizado com o valor “newValue”

src > utils > forms.ts > ...

```ts
export function update(inputs: any, name: string, newValue: any) {
    return{ ...inputs, [name]: { ...inputs[name], value: newValue} };
}
```

IMPLEMENTAR A PAGE NEW_FORM

Implementar os Imports

src > pages > NewForm > index.tsx > ...

```tsx
import './styles.css';
import { Link } from 'react-router-dom';
import ButtonSecondary from '../../components/ButtonSecondary';
import ButtonPrimary from '../../components/ButtonPrimary';
import FormInput from '../../components/FormInput';
import { useState } from 'react';
import * as forms from '../../utils/forms';
```

Implementar o formData com useState

src > pages > NewForm > index.tsx > ...

```tsx
const [formData, setFormData] = useState<any>({
    name: {
        value: "",
        id: "name",
        name: "name",
        type: "text",
        placeholder: "Nome",
    },
    cpf: {
        value: "",
        id: "cpf",
        name: "cpf",
        type: "text",
        placeholder: "CPF",
    },
    birthDate: {
        value: "",
        id: "birthDate",
        name: "birthDate",
        type: "date",
        placeholder: "Data de Nascimento",
    },
    income: {
        value: [],
        id: "income",
        name: "income",
        type: "double",
        placeholder: "Salário",
    },
});
```

Implementar o FormInput

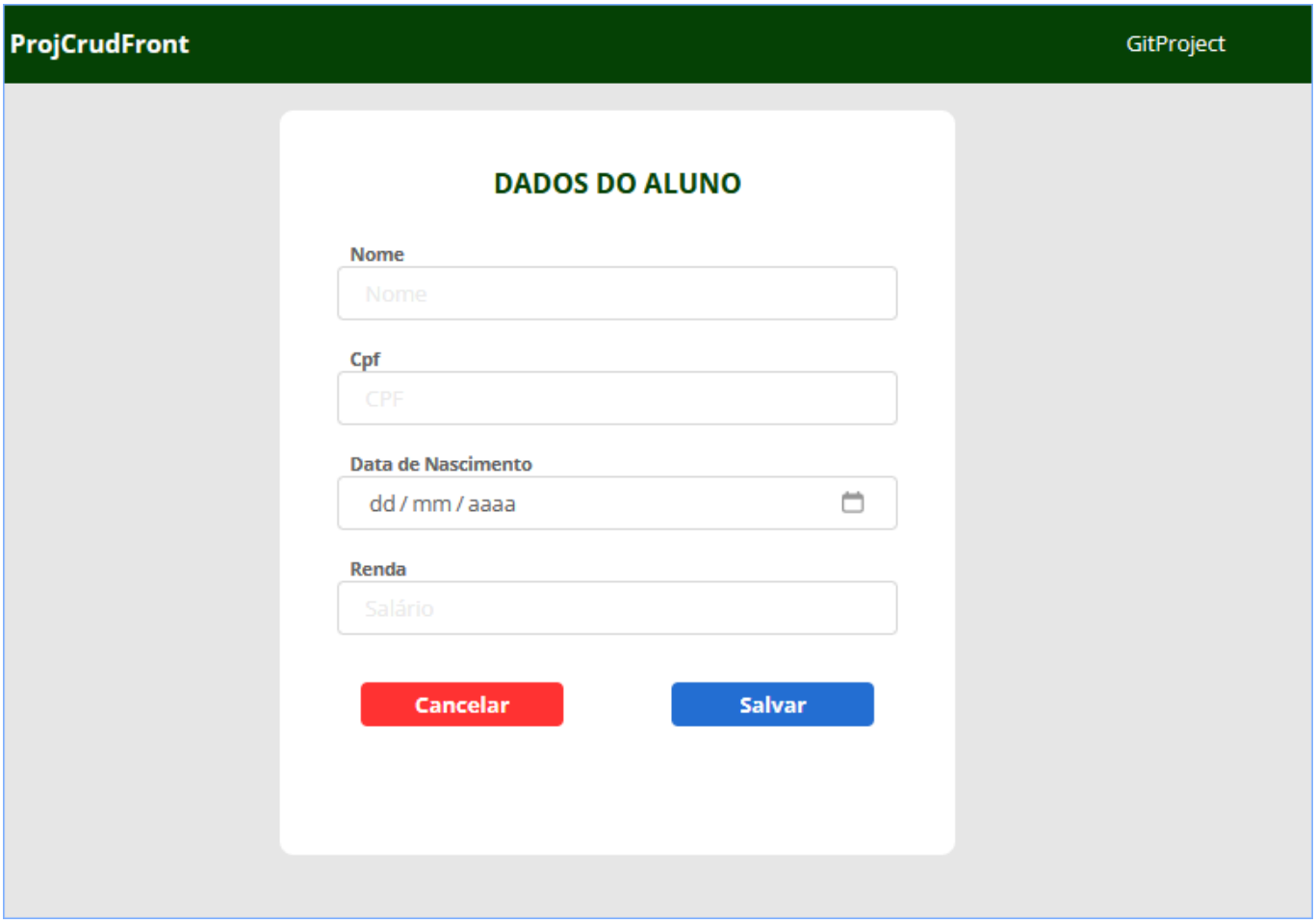
src > pages > NewForm > index.tsx > ...

```tsx
return (
    <main>
        <section id="proj-form-section" className="proj-container">
            <div className="proj-form-container">
                <form className="proj-card proj-form">
                    <h2 className="proj-mb20">Dados do Aluno</h2>
                    <div className="proj-form-controls-container">
                        <div>
                            <h5>Nome</h5>
                            <FormInput
                                { ...formData.name }
                                className="proj-form-control"
                                onChange={handleInputChange}
                            />
                        </div>
                        <div>
                            <h5>Cpf</h5>
                            <FormInput
                                { ...formData.cpf }
                                className="proj-form-control"
                                onChange={handleInputChange}
                            />
                        </div>
                        <div>
                            <h5>Data de Nascimento</h5>
                            <FormInput
                                { ...formData.birthDate }
                                className="proj-form-control"
                                onChange={handleInputChange}

                            />
                        </div>
                        <div>
                            <h5>Renda</h5>
                            <FormInput
                                { ...formData.income }
                                className="proj-form-control"
                                onChange={handleInputChange}
                            />
                        </div>
                    </div>
```

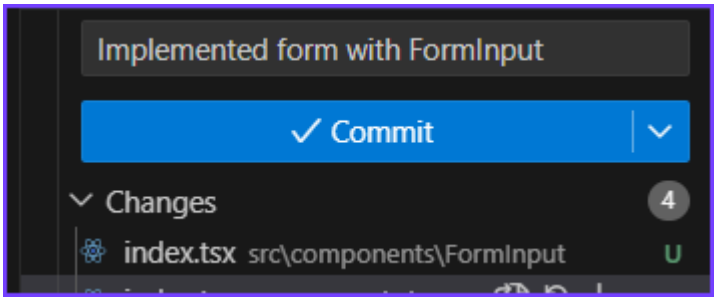
Implementar a função handleInputChange

```tsx

    function handleInputChange(event: any) {
        setFormData(forms.update(formData, event.target.name, event.target.value));
    }
```

Visualização web



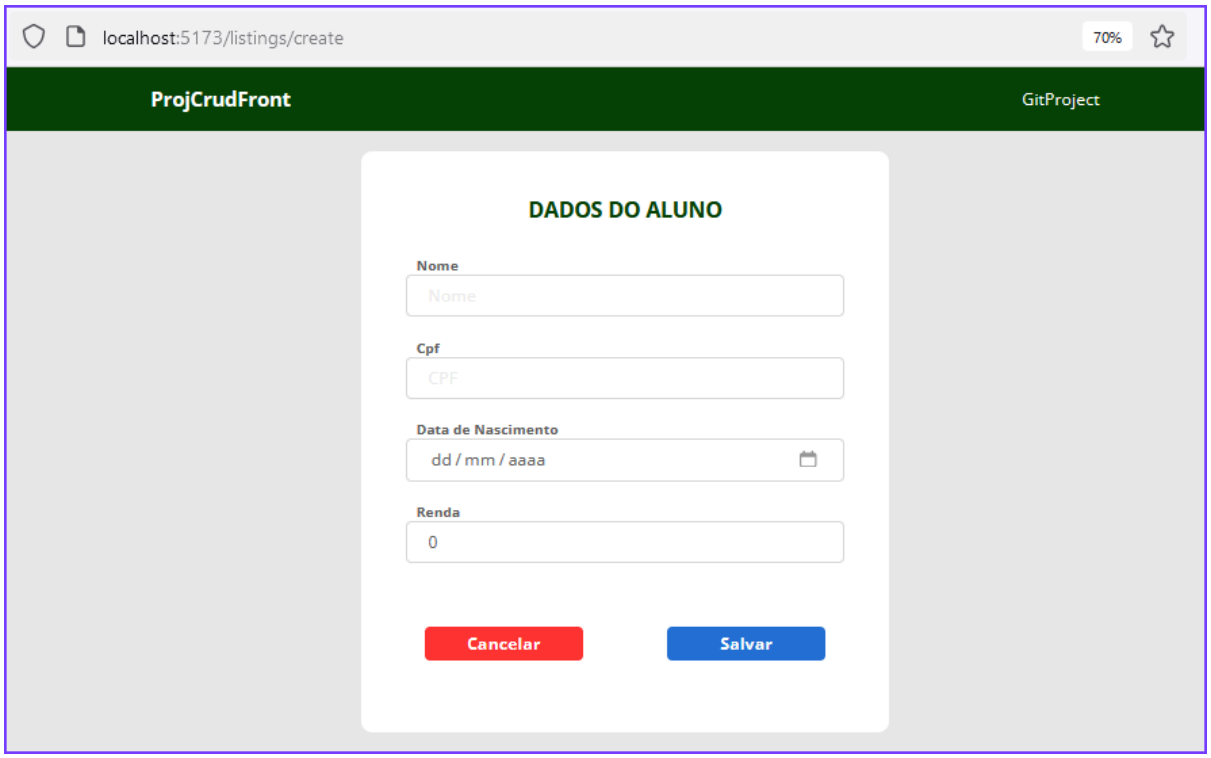
GitHub-13



DIFERENCIAR CRIAÇÃO E EDIÇÃO NO FORMULÁRIO

IMPLEMENTAR A ESTRATÉGIA PARA O FORMULÁRIO DE NOVO ALUNO

***NOTA**: Ao clicar no botão "Novo", da página Listing, deverá ser redirecionada para um formulário em branco, com a URL "/create".*



Rever a rota do studentId

src > App.tsx > ...

```
16      <Routes>
17        <Route path='/' element={<App />} />
18        <Route index element={<Home />} />
19        <Route path='listings' element={<Listing />}/>
20        <Route path='Listings/:studentId' element={<NewForm />}/>
21        <Route path='*' element={<Navigate to='/' />}/>
22      </Routes>
23    </BrowserRouter>
```

Implementar o botão Novo, na página Listing

src > pages > Listing > index.tsx > ...

```
38
39                  <div className="proj-mt40 proj-mb20">
40                      <div className="proj-listing-btn" onClick={handleNewProduct}>
41                          <ButtonPrimary name='Novo' />
42                      </div>
43                  </div>
```

Implementar o Navigate

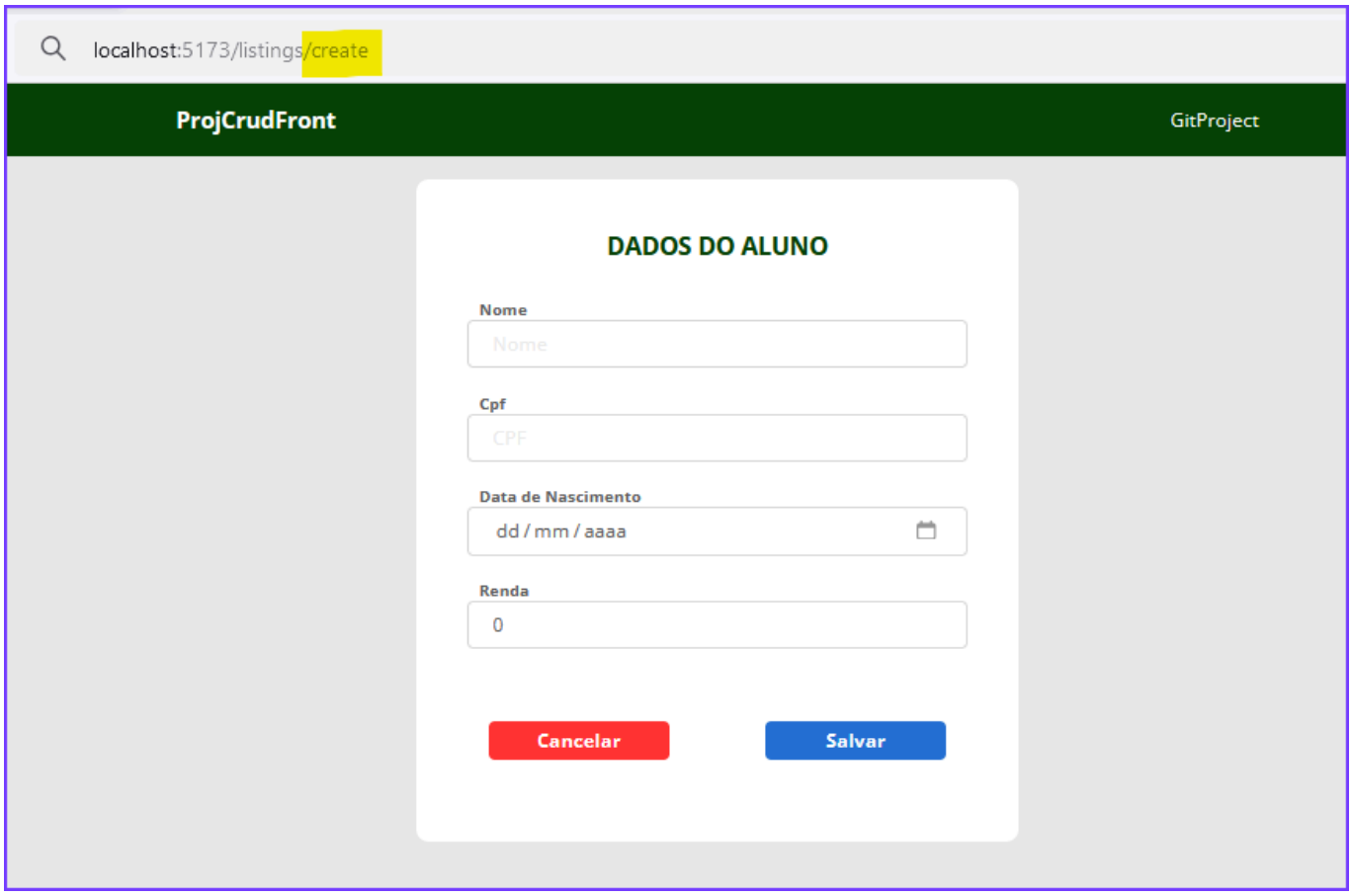
src > pages > Listing > index.tsx > ...

```
15
16  export default function Listing() {
17
18      const navigate = useNavigate();
19
```

Implementar a função handleNewProduct

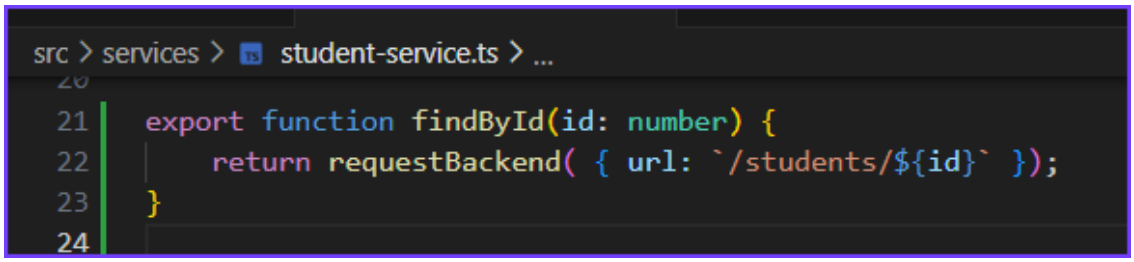
```
src > pages > Listing > index.tsx > ...
35      function handleNewProduct() {
36          navigate("/listings/create")
37      };
```

Visualização web



IMPLEMENTAR A ESTRATÉGIA PARA O FORMULÁRIO DE EDIÇÃO ALUNOS

Implementar o findbyId no student-service para requisição no backend



Implementar useParams no NewForm

src > pages > NewForm > index.tsx > ...

```tsx
export default function NewForm() {

    const params = useParams();

    const isEditing = params.studentId !== 'create';

```

***NOTA**: Caso a URL seja diferente de "create", significa que não é um aluno novo e sim uma edição de um aluno existente.*

Implementar o useEffect, para pegar os produtos no banco

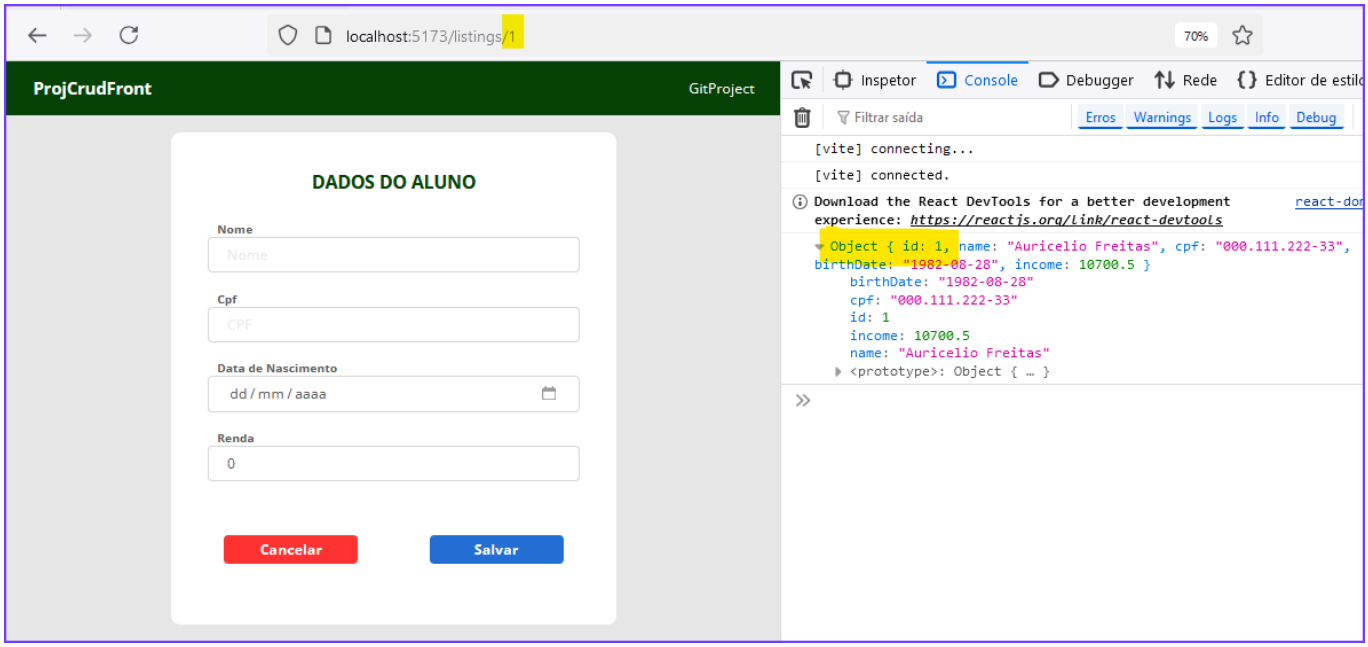
src > pages > NewForm > index.tsx > ...

```tsx
import './styles.css';
import { Link, useParams } from 'react-router-dom';
import ButtonSecondary from '../../components/ButtonSecondary';
import ButtonPrimary from '../../components/ButtonPrimary';
import * as studentService from '../../services/student-service';
```

src > pages > NewForm > index.tsx > ...

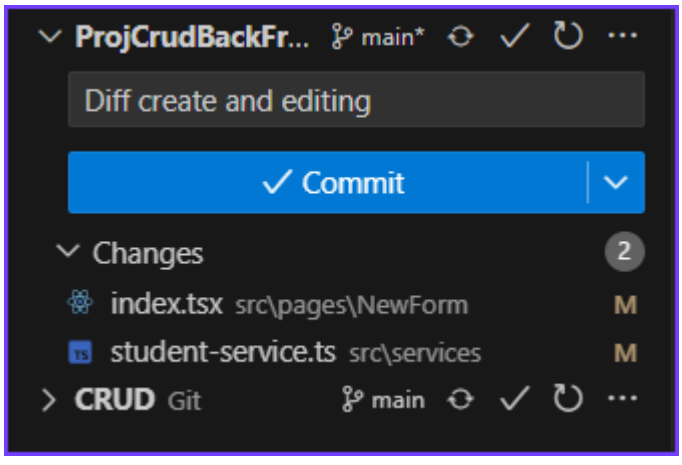
```tsx
    useEffect(() => {
        if (isEditing) {
            studentService.findById(Number(params.studentId))
            .then(response => {
                console.log(response.data);
            })
        }
    }, []);
```

Visualização web



***NOTA**: Neste momento, já estamos buscando no banco de dados os alunos por ID.*

GitHub-14



ADICIONAR OS VALORES DO BANCO NO FORMULÁRIO DE EDIÇÃO

IMPLEMENTAR A FUNÇÃO UPDATE_ALL, NO UTILITÁRIO FORMS

- **Objetivo:** gerar um novo objeto de formulário onde os campos sejam os valores contidos em "newValues".

src > utils > forms.ts > ...

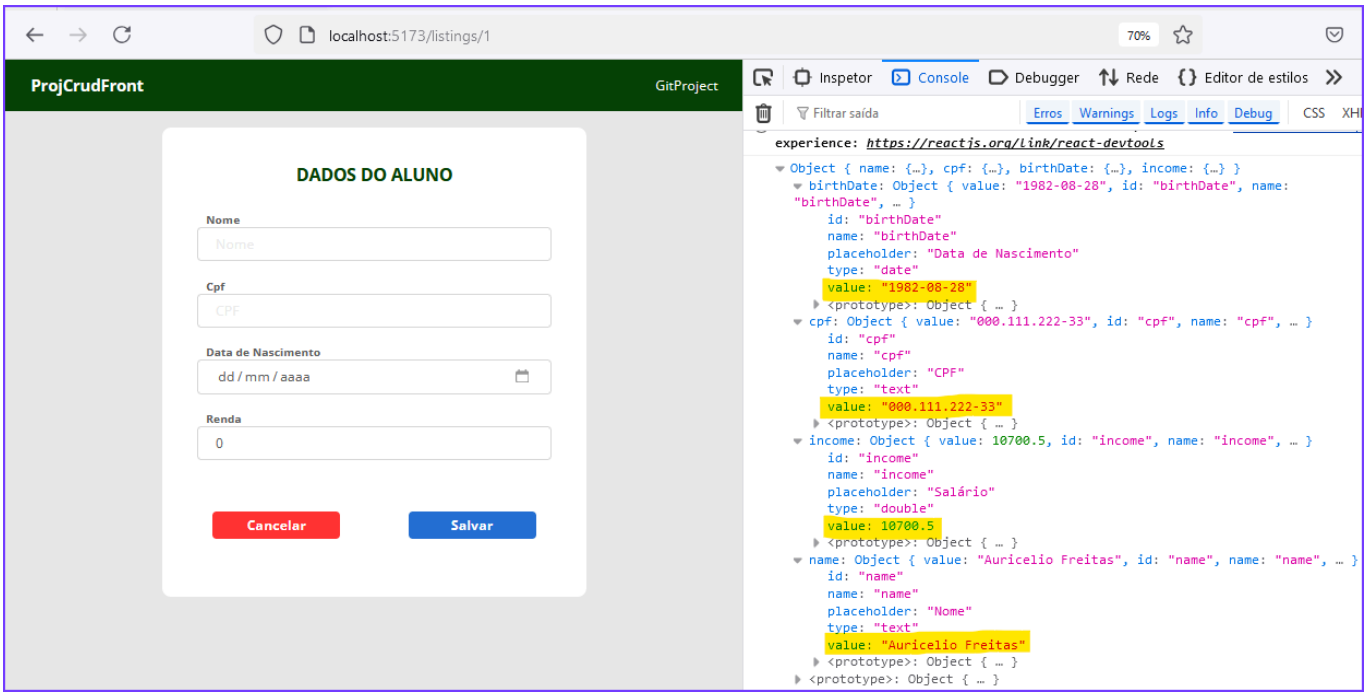
```ts
export function updateAll(inputs: any, newValue: any) {
    const newInputs: any = {};
        for (var name in inputs) {
            newInputs[name] = { ...inputs[name], value: newValue[name] };
        }

    return newInputs;
}
```

CHAMAR O UPDATE_ALL NO USE_EFFECT DO NEW_FORM

src > pages > NewForm > index.tsx > ...

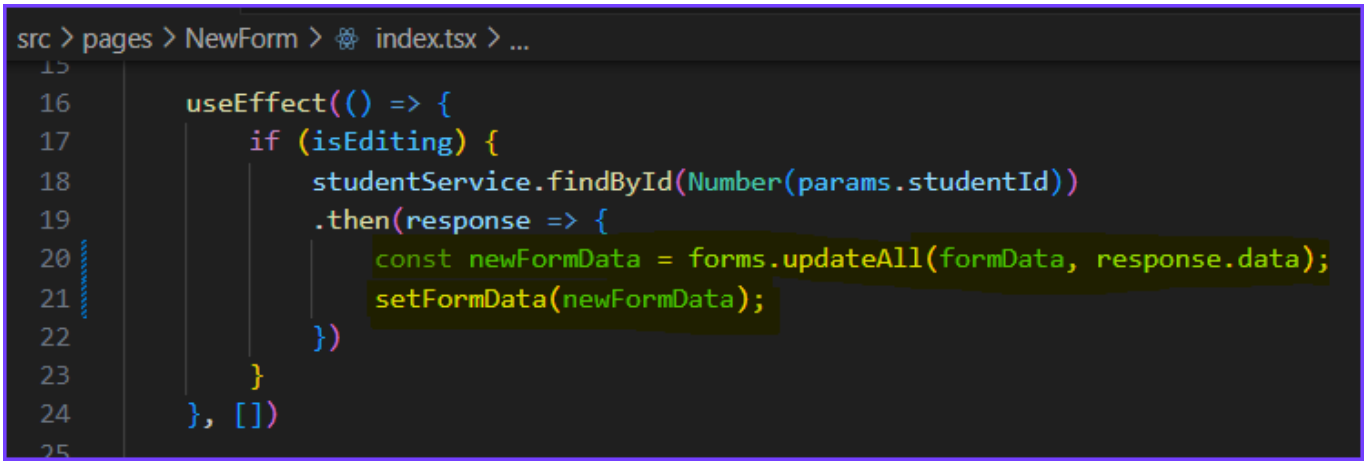
```tsx
    useEffect(() => {
        if (isEditing) {
            studentService.findById(Number(params.studentId))
            .then(response => {
                console.log(forms.updateAll(formData, response.data));
            })
        }
    }, [])
```

VISUALIZAÇÃO WEB

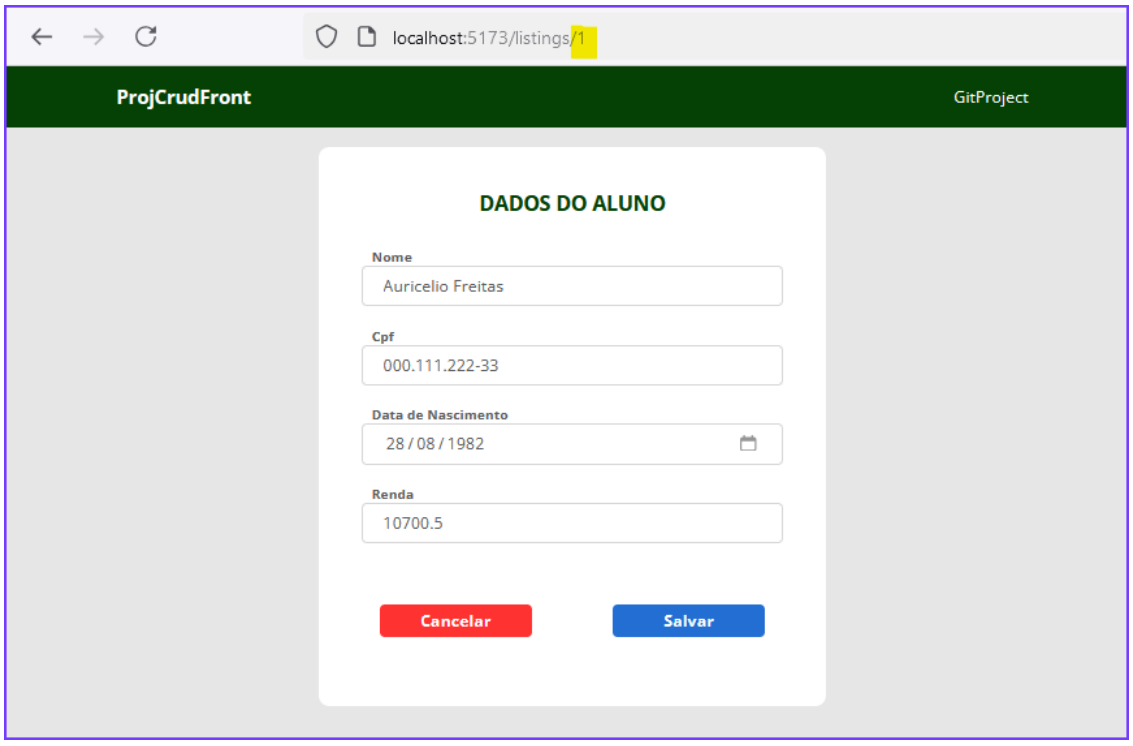


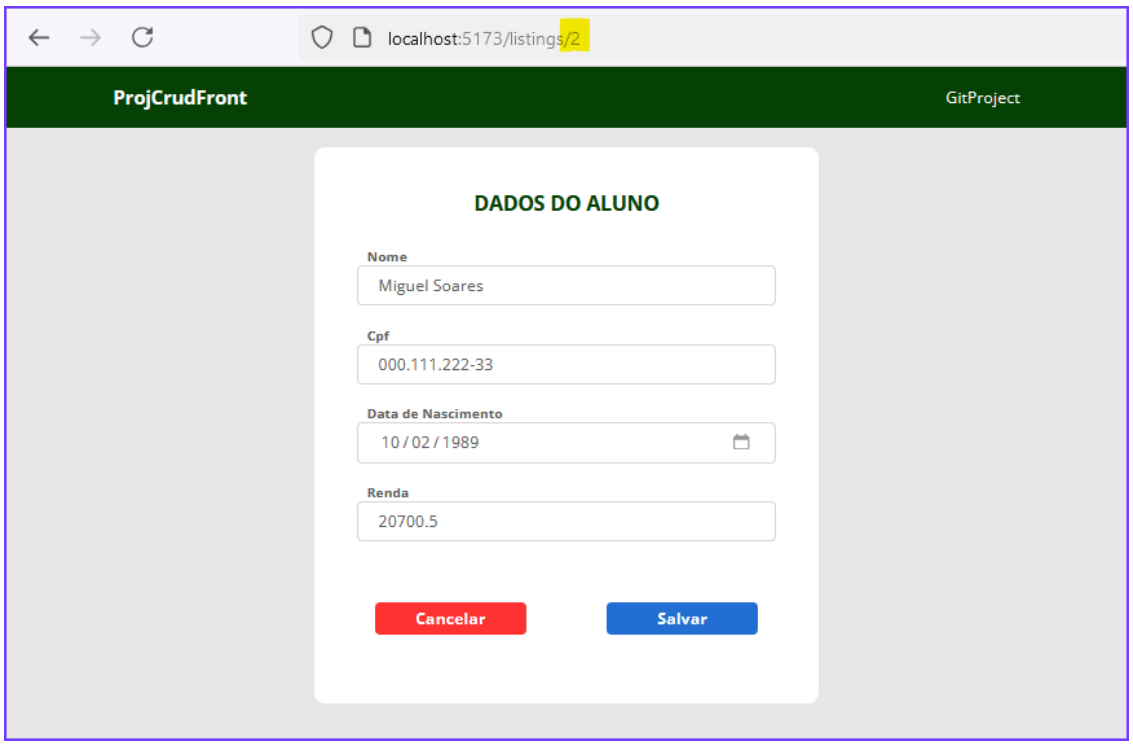
***NOTA**: Neste momento, o sistema já está pegando os "values" do banco de dados.*

IMPLEMENTAR O SET_FORM_DATA NO USE_EFFECT DO NEW_FORM

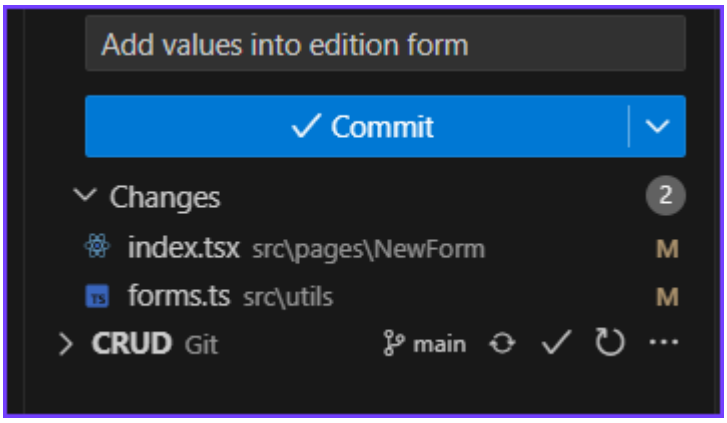


VISUALIZAÇÃO WEB



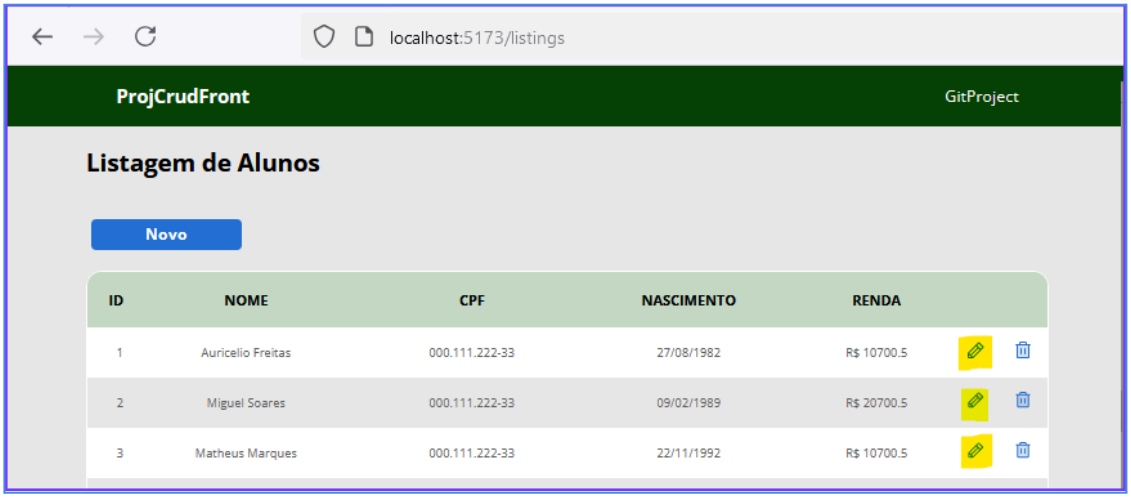


GitHub-15


Add values into edition form
Commit
Changes
2
index.tsx src\pages\NewForm
M
forms.ts src\utils
M
CRUD Git
main

ADICIONAR O NAVIGATE AO BOTÃO EDITAR

***NOTA**: Ao clicar na imagem "Lápis", da página Listing, deverá ser redirecionada para um formulário contendo os dados referentes ao ID selecionado.*



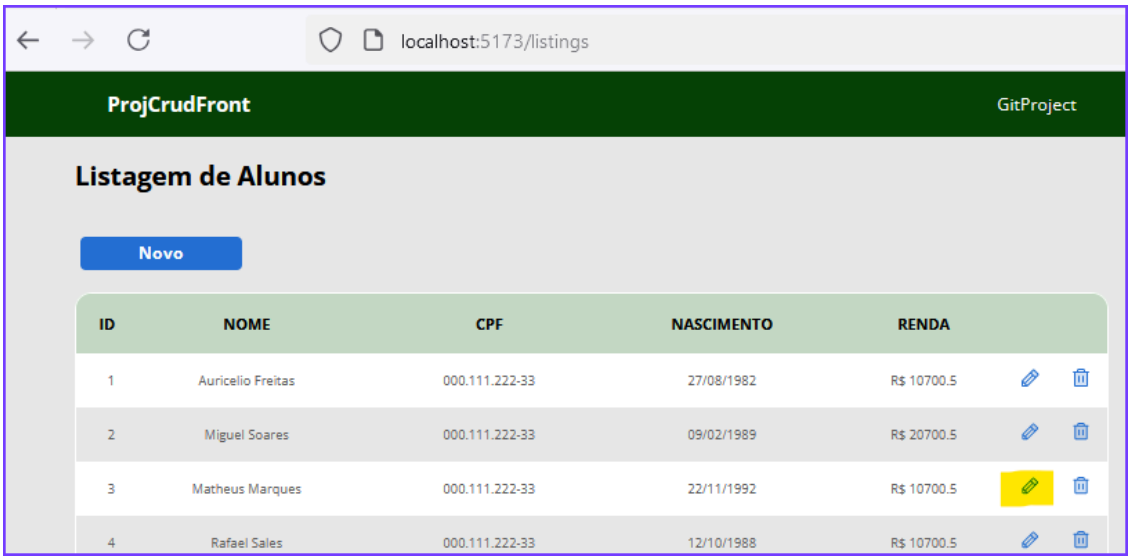
IMPLEMENTAR O ONCLICK DO BOTÃO EDITAR, NO LISTING

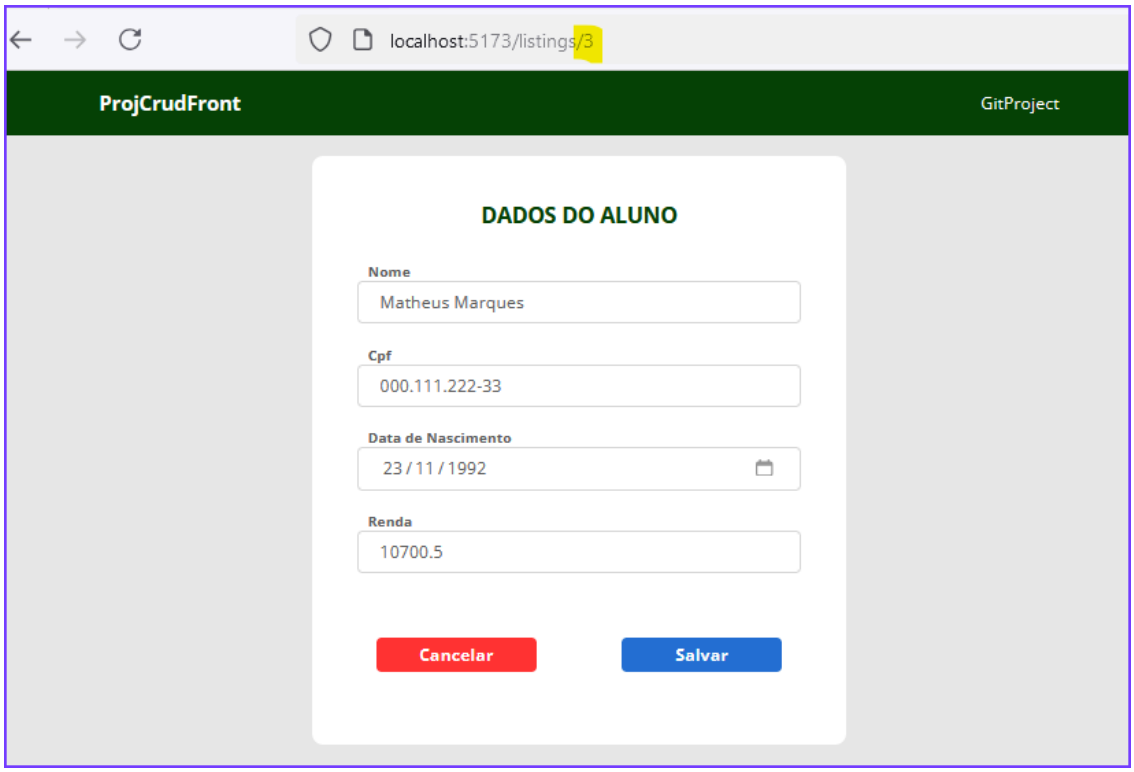
IMPLEMENTAR A FUNÇÃO PARA O HANDLEUPDATE

src > pages > Listing > index.tsx > ...

```
39      function handleUpdate(studentId: number) {
40          navigate(`/listings/${studentId}`);
41      }
42
```

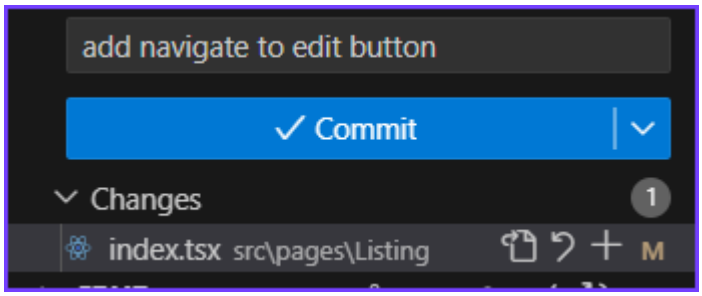
VISUALIZAÇÃO WEB





***NOTA**: Neste momento, ao clicar no botão de editar será aberto o formulário contendo os dados do ID selecionado.*

GitHub-16

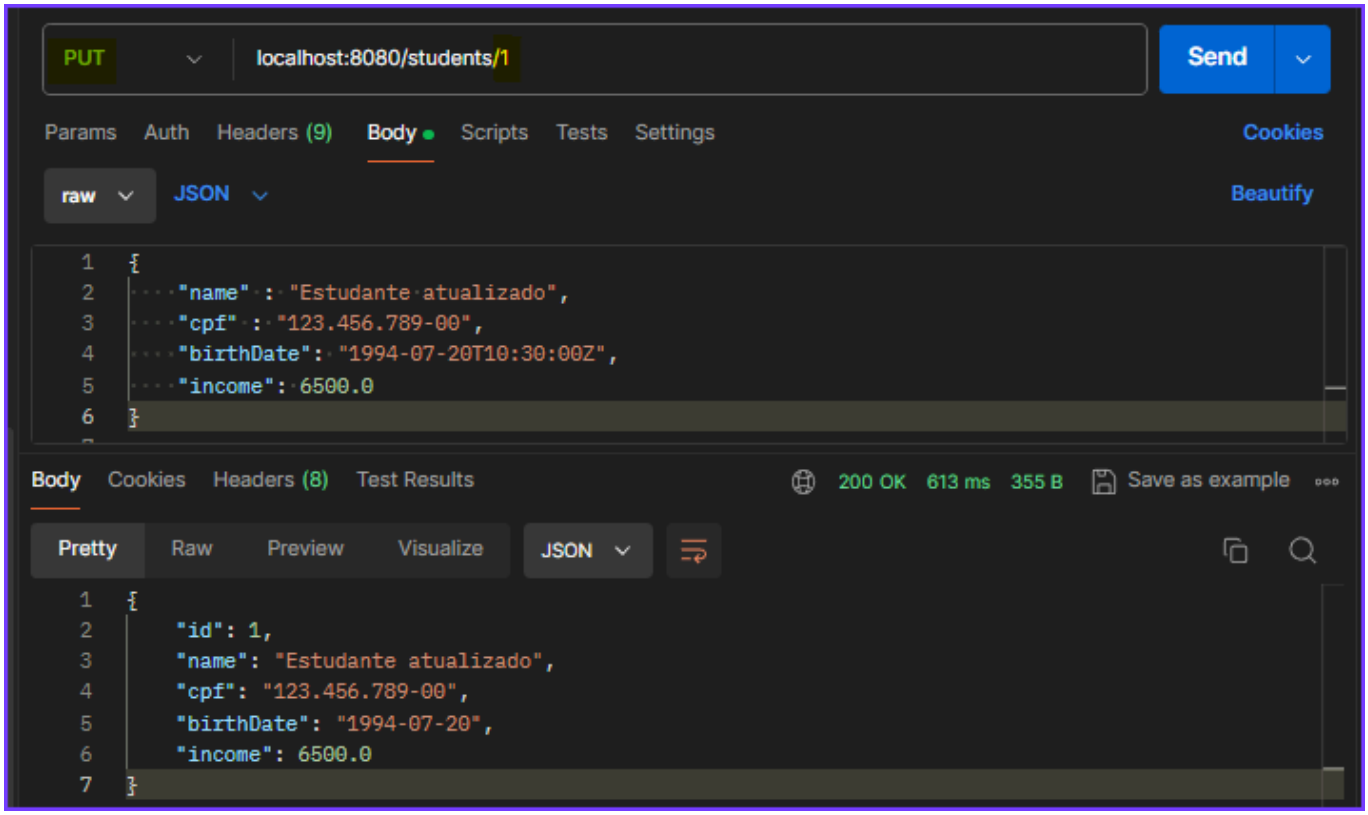


SUBMETER VALORES AO BANCO (CRUD)

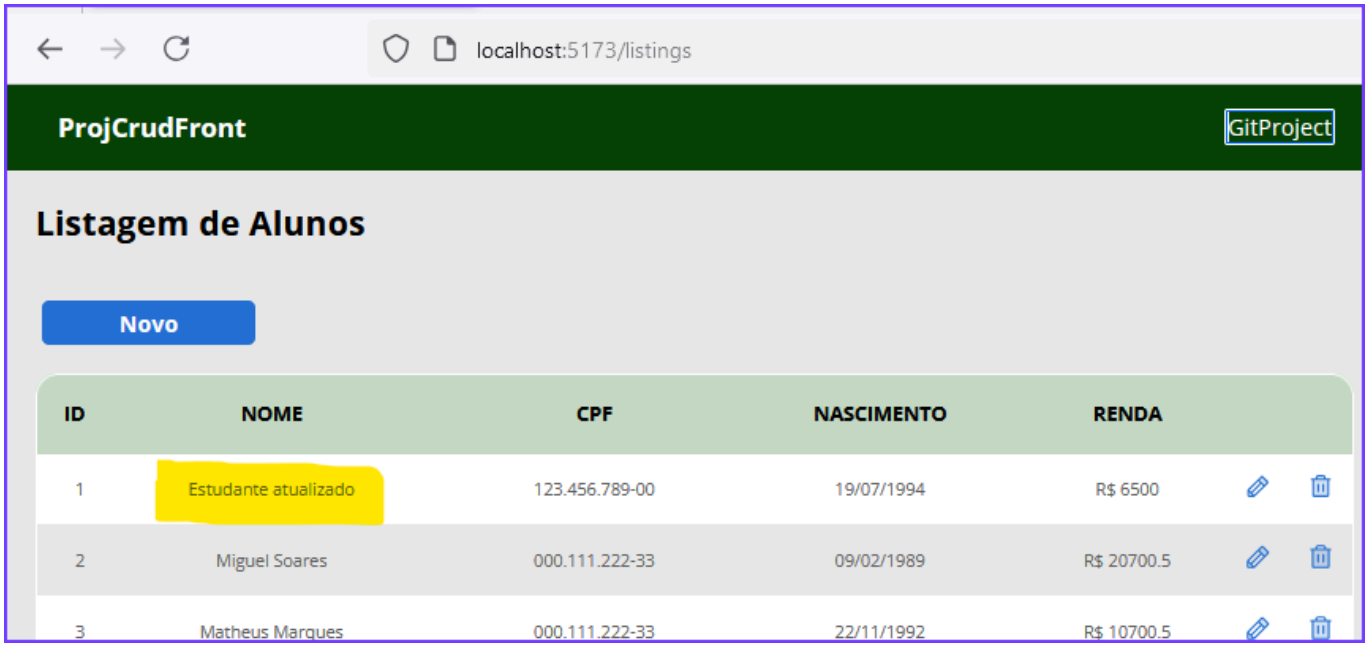
***NOTA**: O Projeto deve estar rodando, tanto no frontend, quanto no backend.*

EDITAR COM O PUT

Testar resposta no postman



Visualização web

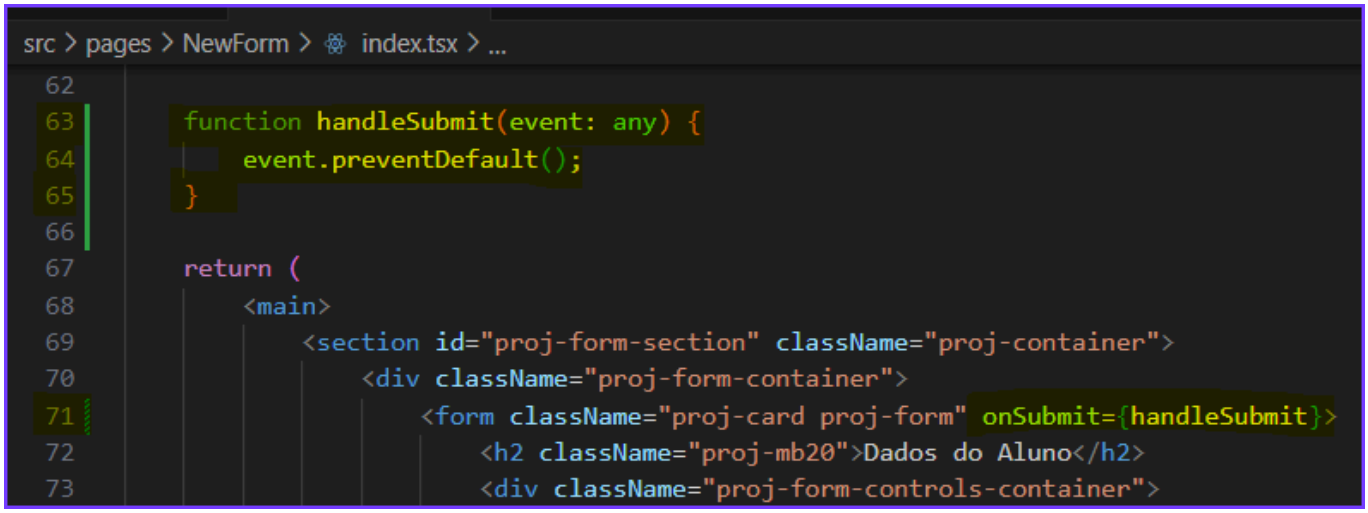
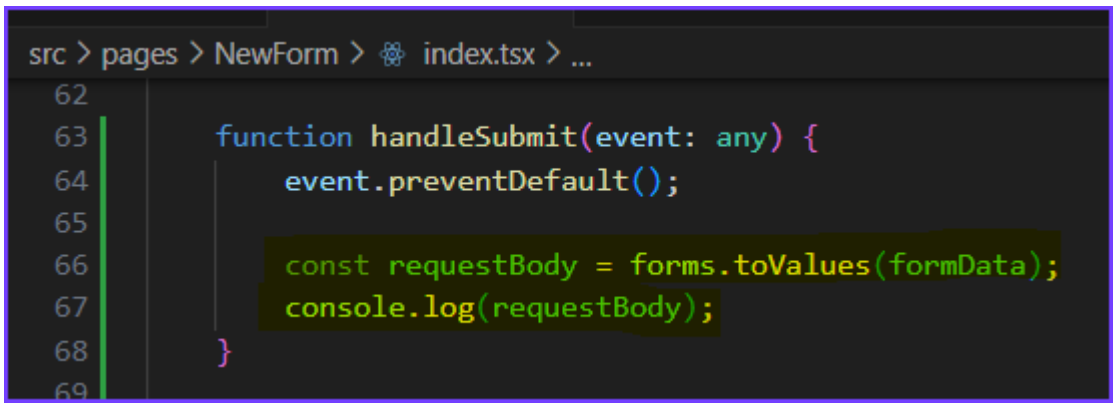


Implementar o toValues, no utilitario “forms”, para renderizar o objeto

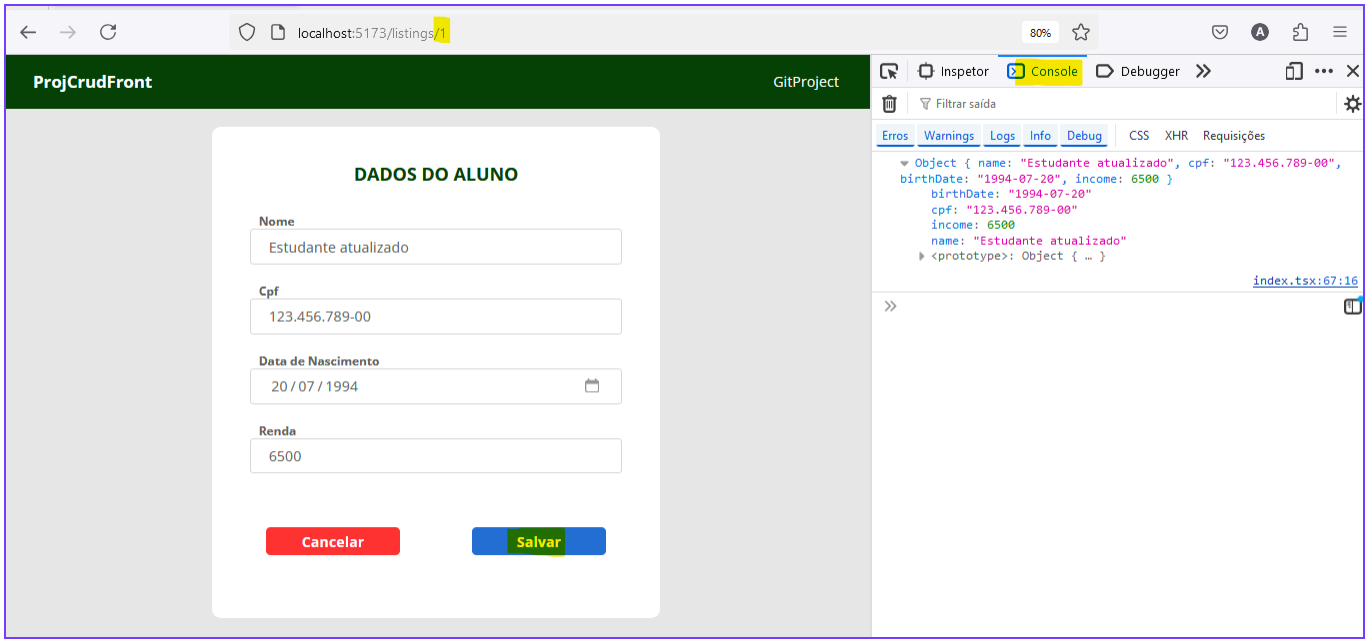
src > utils > forms.ts > ...

```
13
14  export function toValues(inputs: any) {
15
16      const data: any = {};
17      for (var name in inputs) {
18          data[name] = inputs[name].value;
19      }
20      return data;
21  }
```

Implementar o handleSubmit, no NewForm


src > pages > NewForm > index.tsx > ...
62
63 function handleSubmit(event: any) {
64 event.preventDefault();
65 }
66
67 return (
68 <main>
69 <section id="proj-form-section" className="proj-container">
70
71 <form className="proj-card proj-form" onSubmit={handleSubmit}>
72 <h2 className="proj-mb20">Dados do Aluno</h2>
73
```

Implementar um teste


```

Visualização web



Implementar uma requisição PUT, no student-service

src > services > student-service.ts > ...

```
25
26  export function updateRequest(obj: StudentDTO) {
27      const config : AxiosRequestConfig = {
28          method: "PUT",
29          baseURL: BASE_URL,
30          url: `/students/${obj.id}`,
31          data: obj
32      }
33
34      return requestBackend(config)
35  }
```

Implementar o navigate, no NewForm

src > pages > NewForm > index.tsx > ...

```tsx
import './styles.css';
import { Link, useNavigate, useParams } from 'react-router-dom';
import ButtonSecondary from '../../components/ButtonSecondary';
import ButtonPrimary from '../../components/ButtonPrimary';
import FormInput from '../../components/FormInput';
import { useEffect, useState } from 'react';
import * as forms from '../../utils/forms';
import * as studentService from '../../services/student-service';

export default function NewForm() {

    const params = useParams();

    const navigate = useNavigate();

    const isEditing = params.studentId !== 'create';
```

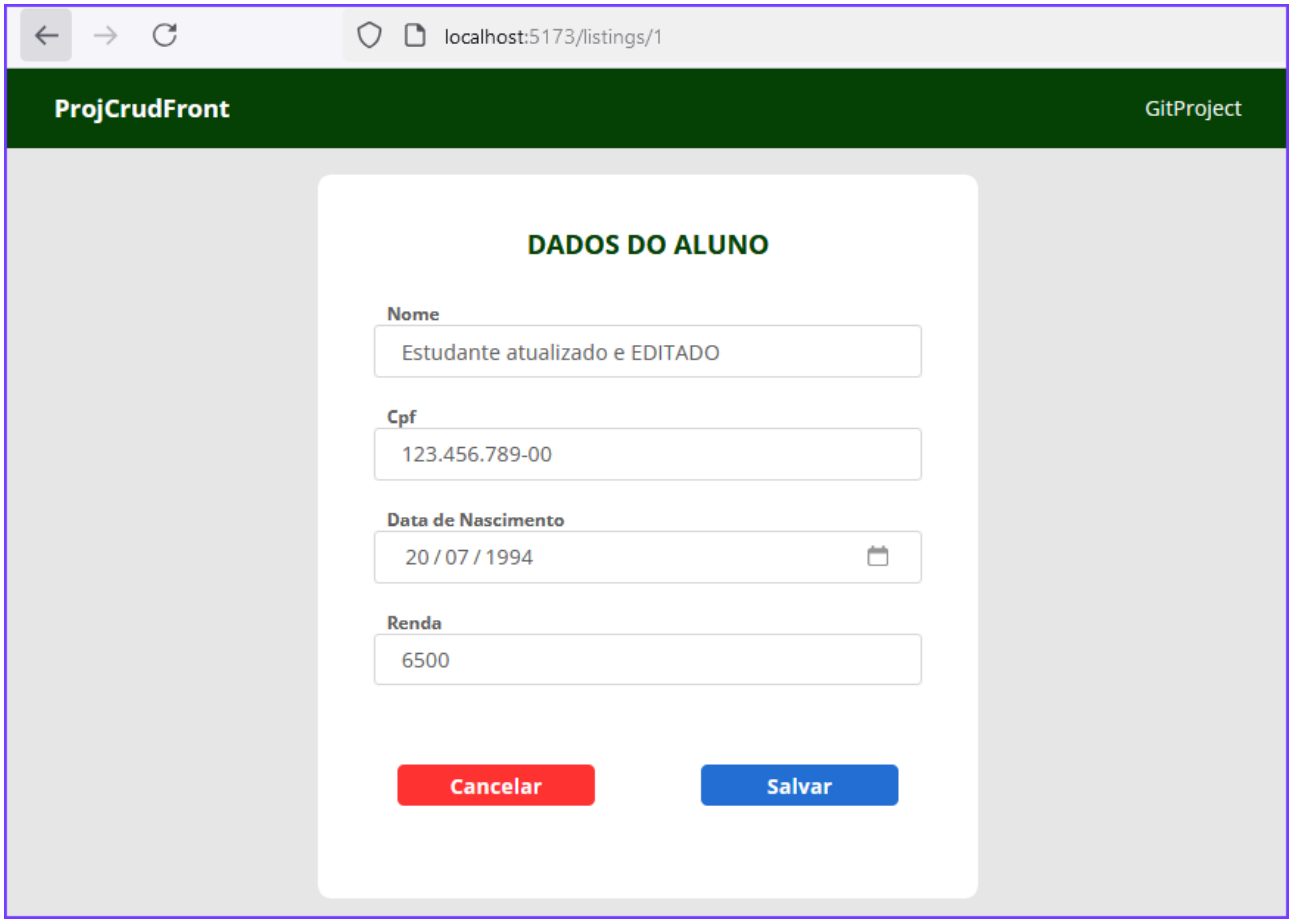
Implementar o updateRequest, no NewForm

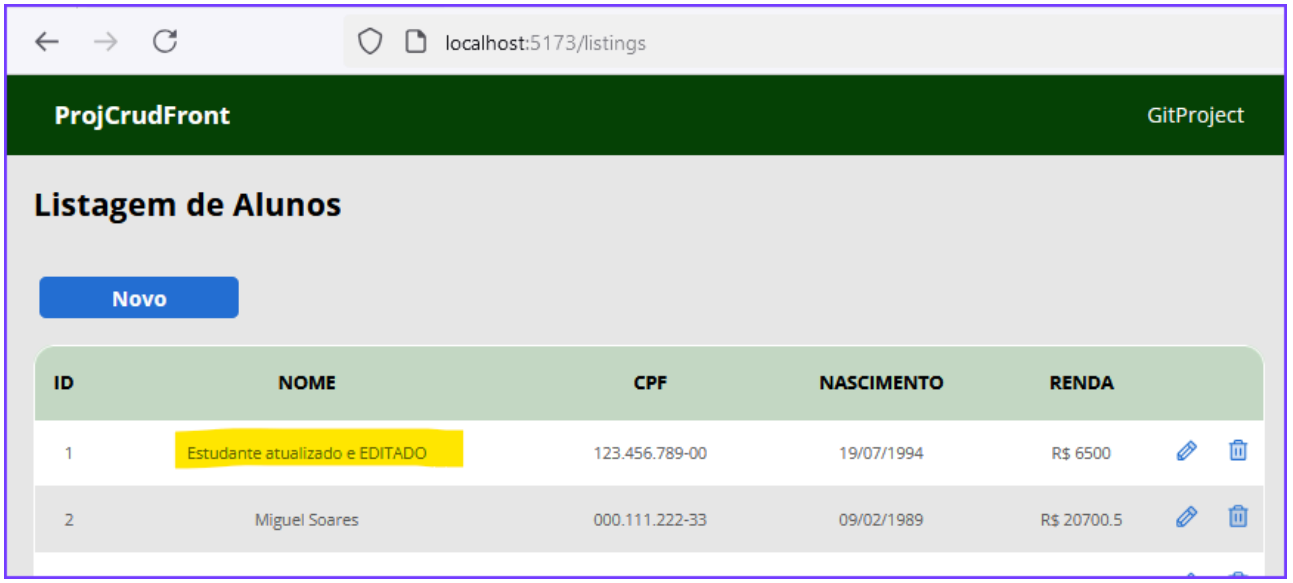
src > pages > NewForm > index.tsx > ...

```tsx
    function handleSubmit(event: any) {
        event.preventDefault();

        const requestBody = forms.toValues(formData);
        if (isEditing) {
            requestBody.id = params.studentId;
        }

        studentService.updateRequest(requestBody)
        .then(() => {
            navigate("/listings");
        });
    }
```

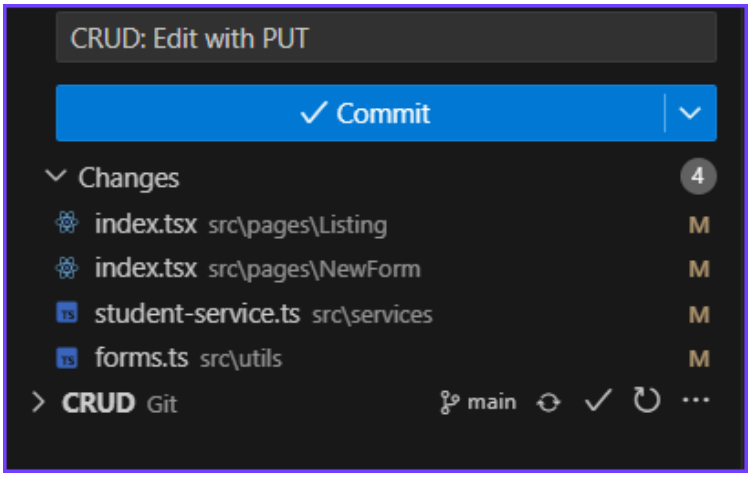
***NOTA**: Após Editar, será redirecionado para a página Listing com o novo valor atualizado.*

Visualização web



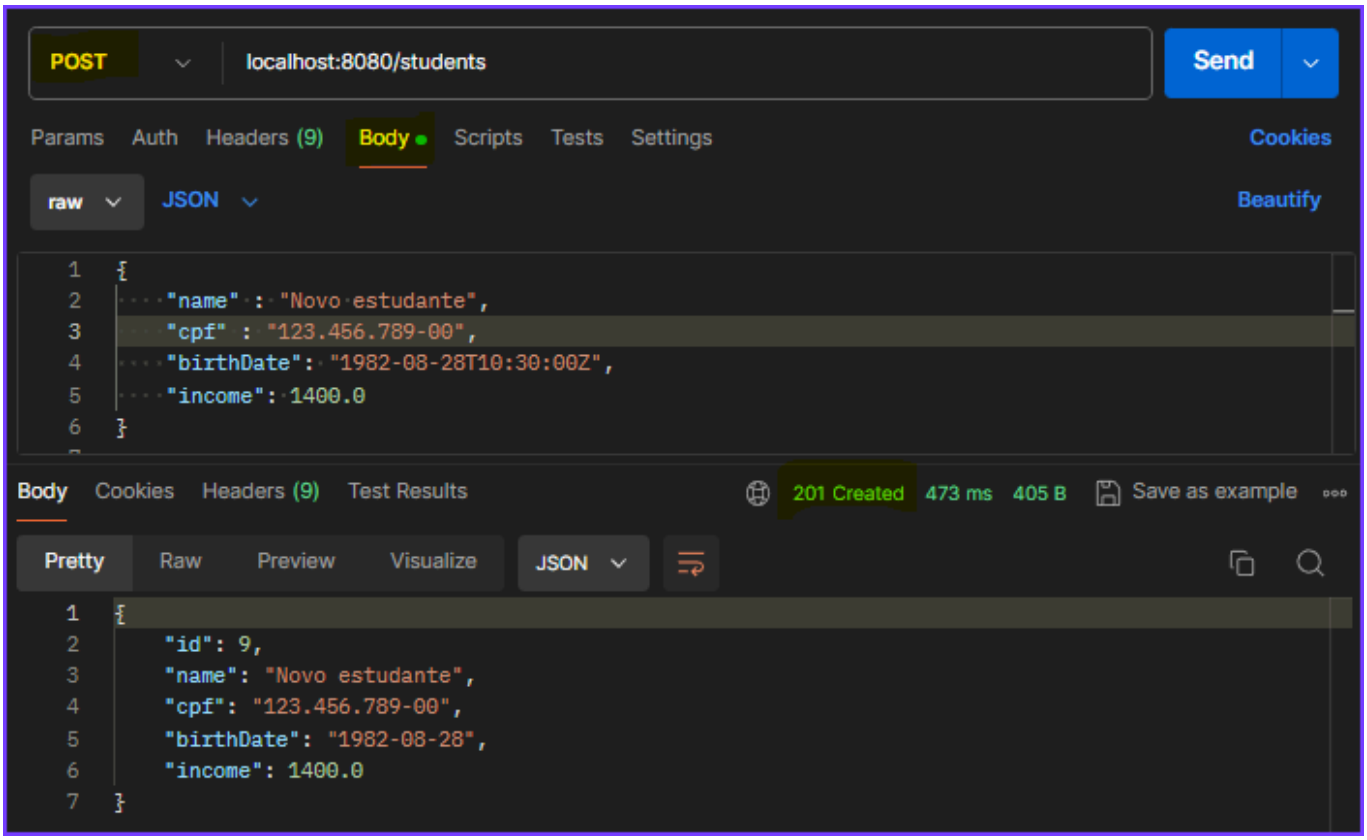


GitHub-17

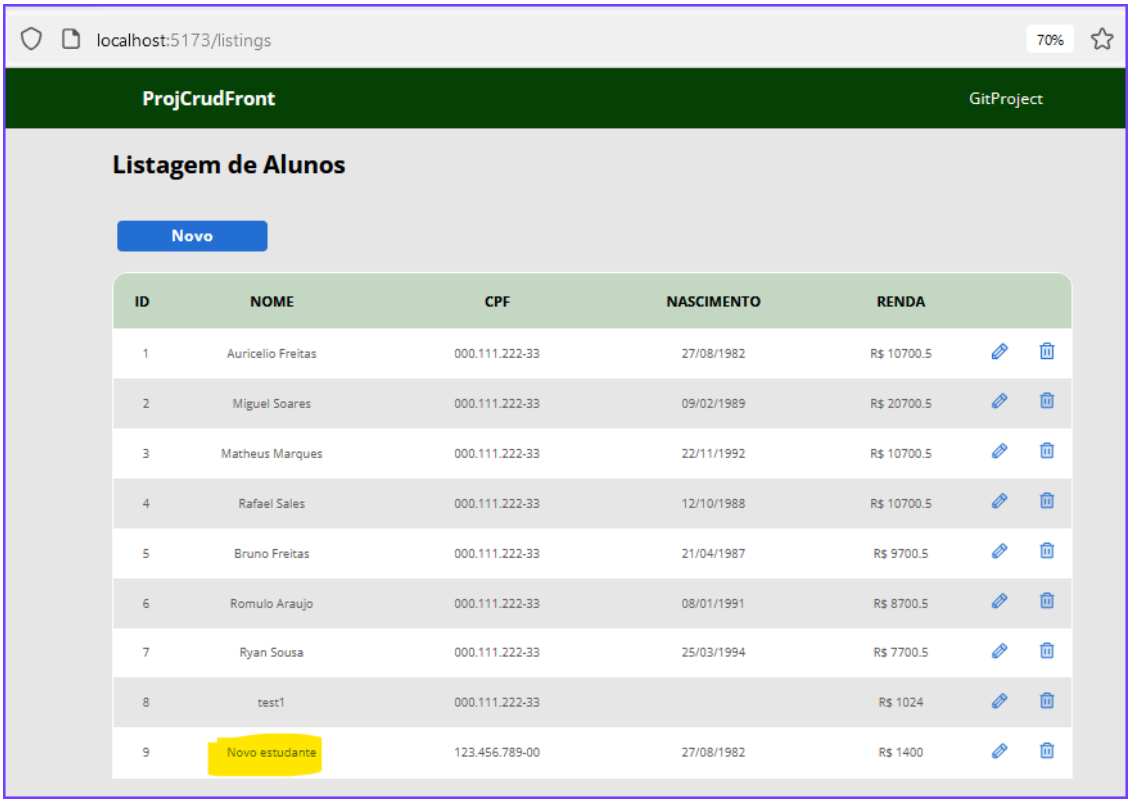


INSERIR COM O POST

Testar resposta no postman



Visualização web



Implementar uma requisição POST, no student-service

src > services > student-service.ts > ...

```
36
37  export function insertRequest(obj: StudentDTO) {
38      const config : AxiosRequestConfig = {
39          method: "POST",
40          baseURL: BASE_URL,
41          url: "/students",
42          data: obj
43      }
44
45      return requestBackend(config)
46  }
47
```

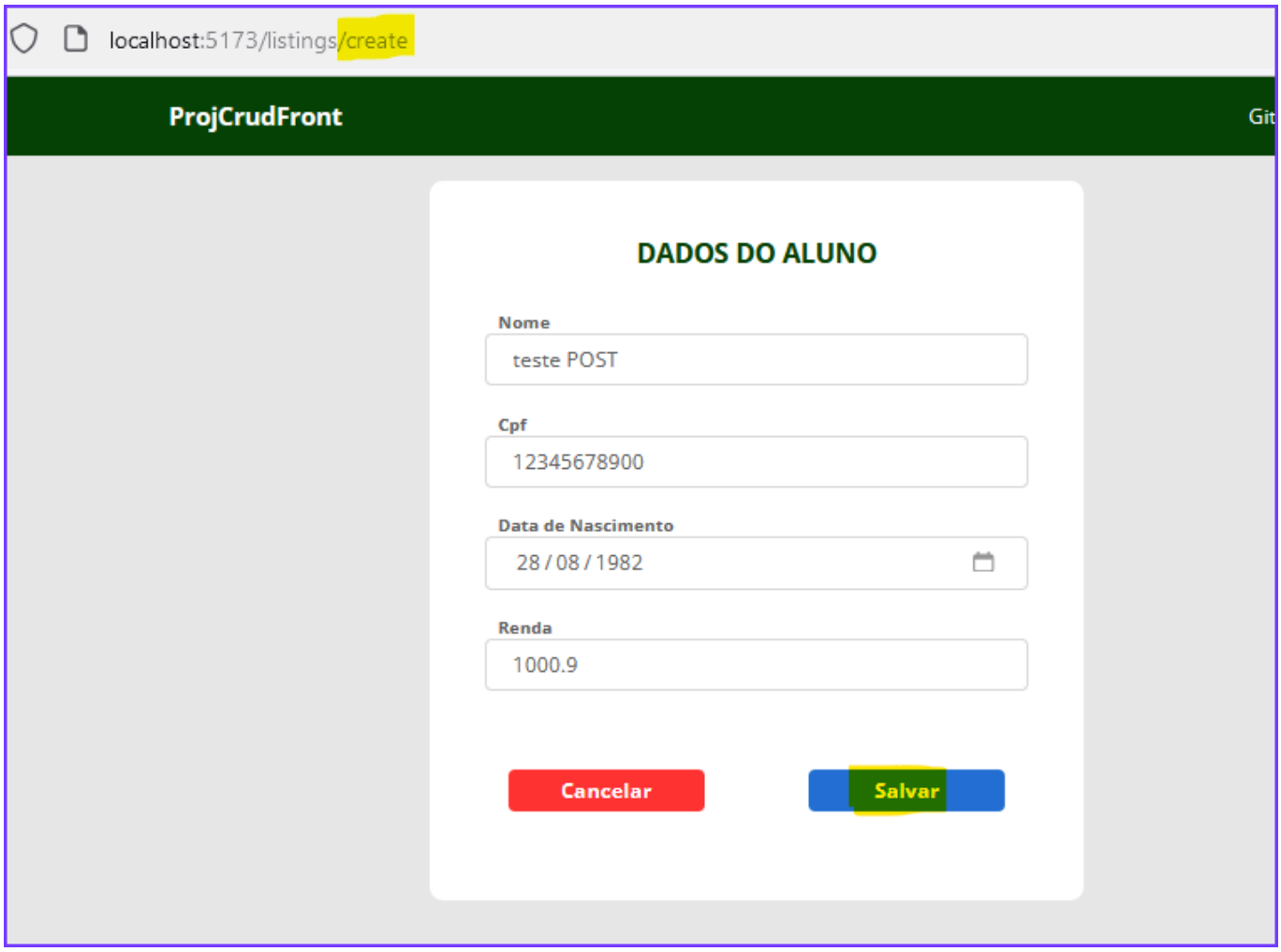
Implementar o insertRequest, no NewForm

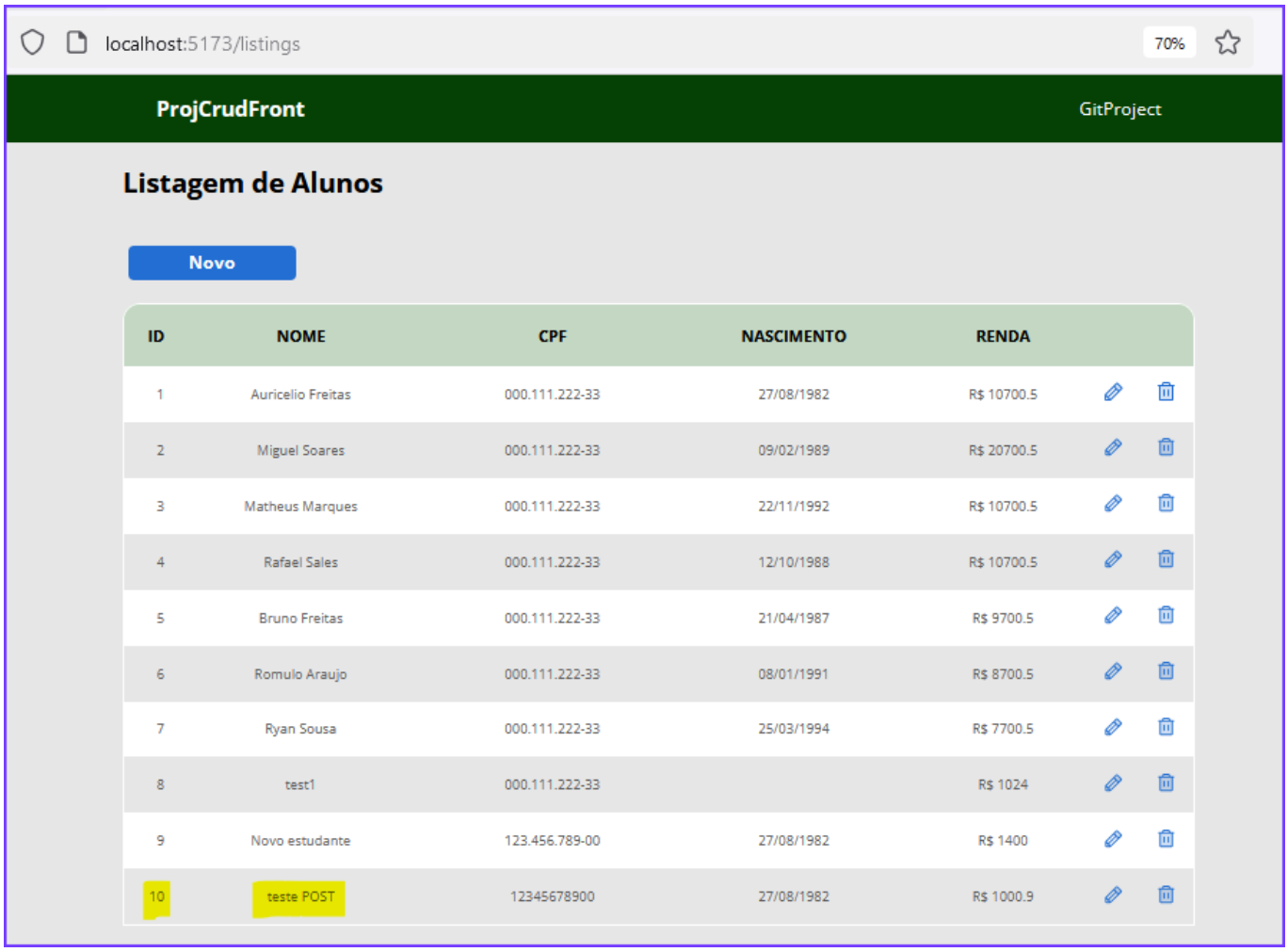
src > pages > NewForm > index.tsx > ...

```
63    function handleSubmit(event: any) {
64        event.preventDefault();
65
66        const requestBody = forms.toValues(formData);
67        if (isEditing) {
68            requestBody.id = params.studentId;
69        }
70
71        const request = isEditing
72            ? studentService.updateRequest(requestBody)
73            : studentService.insertRequest(requestBody)
74
75        request
76            .then(() => {
77                navigate("/listings");
78            });
79    }
```

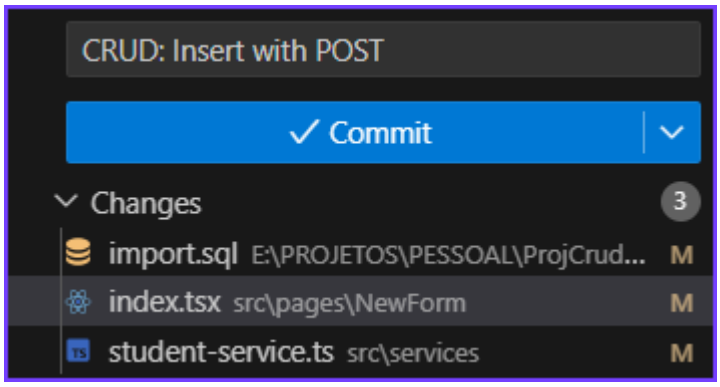
***NOTA**: Após Inserir, será redirecionado para a página Listing, com o novo valor atualizado.*

Visualização web


localhost:5173/listings/create
ProjCrudFront
Git
DADOS DO ALUNO
Nome
teste POST
Cpf
12345678900
Data de Nascimento
28 / 08 / 1982
Renda
1000.9
Cancelar
Salvar

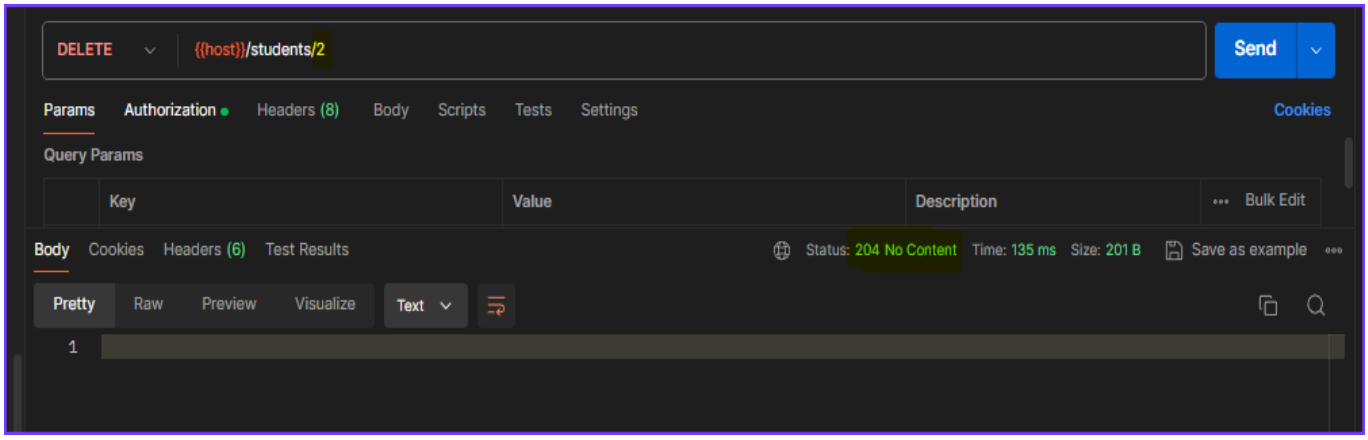

localhost:5173/listings
70%
ProjCrudFront
GitProject
Listagem de Alunos
Novo
ID
NOME
CPF
NASCIMENTO
RENDA
1 Auricelio Freitas 000.111.222-33 27/08/1982 R$ 10700.5
2 Miguel Soares 000.111.222-33 09/02/1989 R$ 20700.5
3 Matheus Marques 000.111.222-33 22/11/1992 R$ 10700.5
4 Rafael Sales 000.111.222-33 12/10/1988 R$ 10700.5
5 Bruno Freitas 000.111.222-33 21/04/1987 R$ 9700.5
6 Romulo Araujo 000.111.222-33 08/01/1991 R$ 8700.5
7 Ryan Sousa 000.111.222-33 25/03/1994 R$ 7700.5
8 test1 000.111.222-33 R$ 1024
9 Novo estudante 123.456.789-00 27/08/1982 R$ 1400
10 teste POST 12345678900 27/08/1982 R$ 1000.9

GitHub-18


CRUD: Insert with POST
Commit
Changes 3
import.sql E:\PROJETOS\PESSOAL\ProjCrud... M
index.tsx src\pages\NewForm M
student-service.ts src\services M

IMPLEMENTAR O DELETE COM CONFIRMAÇÃO

Testar resposta no postman



***NOTA**: No Delete, temos que passar o ID do produto, pois ao clicarmos no botão de apagar, temos que deletar o produto referente ao seu ID.*

Visualização web

Implementar o componente Modal

src > components > DialogConfirmation > index.tsx > ...

```tsx
import ButtonPrimary from '../ButtonPrimary';
import ButtonSecondary from '../ButtonSecondary';
import './styles.css';

type Props = {
    id: number;
    message: string;
    onDialogAnswer: Function;
}
export default function DialogConfirmation( {id, message, onDialogAnswer} : Props) {

    return (
        <div onClick={() => onDialogAnswer(false, id)}>
            <div onClick={(event) => event.stopPropagation}>
                <h2>{message}</h2>

                <div>
                    <div onClick={() => onDialogAnswer(false, id)}>
                        <ButtonSecondary name="Não"/>
                    </div>
                    <div onClick={() => onDialogAnswer(true, id)}>
                        <ButtonPrimary name="Sim"/>
                    </div>
                </div>
            </div>
        </div>
    );
}

```

Estilizar o Modal

src > components > DialogConfirmation > index.tsx > ...

```tsx
export default function DialogConfirmation({id, message, onDialogAnswer} : Props) {

    return (
        <div className="proj-dialog-background" onClick={() => onDialogAnswer(false, id)}>
            <div className="proj-dialog-box" onClick={(event) => event.stopPropagation}>
                <h2>{message}</h2>

                <div className="proj-dialog-btn">
                    <div onClick={() => onDialogAnswer(false, id)}>
                        <ButtonSecondary name="Não"/>
                    </div>
                    <div onClick={() => onDialogAnswer(true, id)}>
                        <ButtonPrimary name="Sim"/>
                    </div>
```

src > components > DialogConfirmation > styles.css > ...

```css
.proj-dialog-background {
  position: fixed;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  background-color: rgba(0, 0, 0, 0.5);
}

.proj-dialog-box {
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  border-radius: 5px;
  background-color: var(--proj-color-card-bg);
  padding: 20px;
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
}

.proj-dialog-box h2 {
  font-size: 20px;
  color: var(--proj-color-font-primary);
  margin-bottom: 20px;
}

.proj-dialog-btn {
  width: 100%;
  display: grid;
  grid-template-columns: 1fr 1fr;
  grid-gap: 20px;
}

```

Implementar a chamado do Dialog no Listing

src > pages > Listing > index.tsx > ...

```
                    </tbody>
                </table>
            </section>
            {
                dialogConfirmationData.visible &&
                <DialogConfirmation
                    id={dialogConfirmationData.id}
                    message={dialogConfirmationData.message}
                    onDialogAnswer={handleDialogConfirmationAnswer}
                />
            }
        </main>
    )
}
```

***NOTA**: Implementar no final, entre o </section> e o </main>.*

Implementar o “onClick” do botão Delete

src > pages > Listing > index.tsx > ...

```
                <tbody>
                    {
                        students.map(student => (
                            <tr key={student.id}>
                                <td className='proj-listing-table-id'>{student.id}</td>
                                <td>{student.name}</td>
                                <td>{student.cpf}</td>
                                <td>{formatDateBR(student.birthDate)}</td>
                                <td>R$ {student.income}</td>
                                <td><img src={editIcon} alt='Editar' onClick={() => {handleUpdate(student.id)}}/></td>
                                <td><img src={deleteIcon} alt='Deletar' onClick={() => handleDelete(student.id)}/></td>
                            </tr>
                        ))
```

Implementar a função handleDelete

src > pages > Listing > index.tsx > ...

```

    function handleDelete(studentId: number) {
        setDialogConfirmationData({ ...dialogConfirmationData, id: studentId, visible: true});
    }

```

Implementar o deleteRequest, no product-service

src > services > student-service.ts > ...

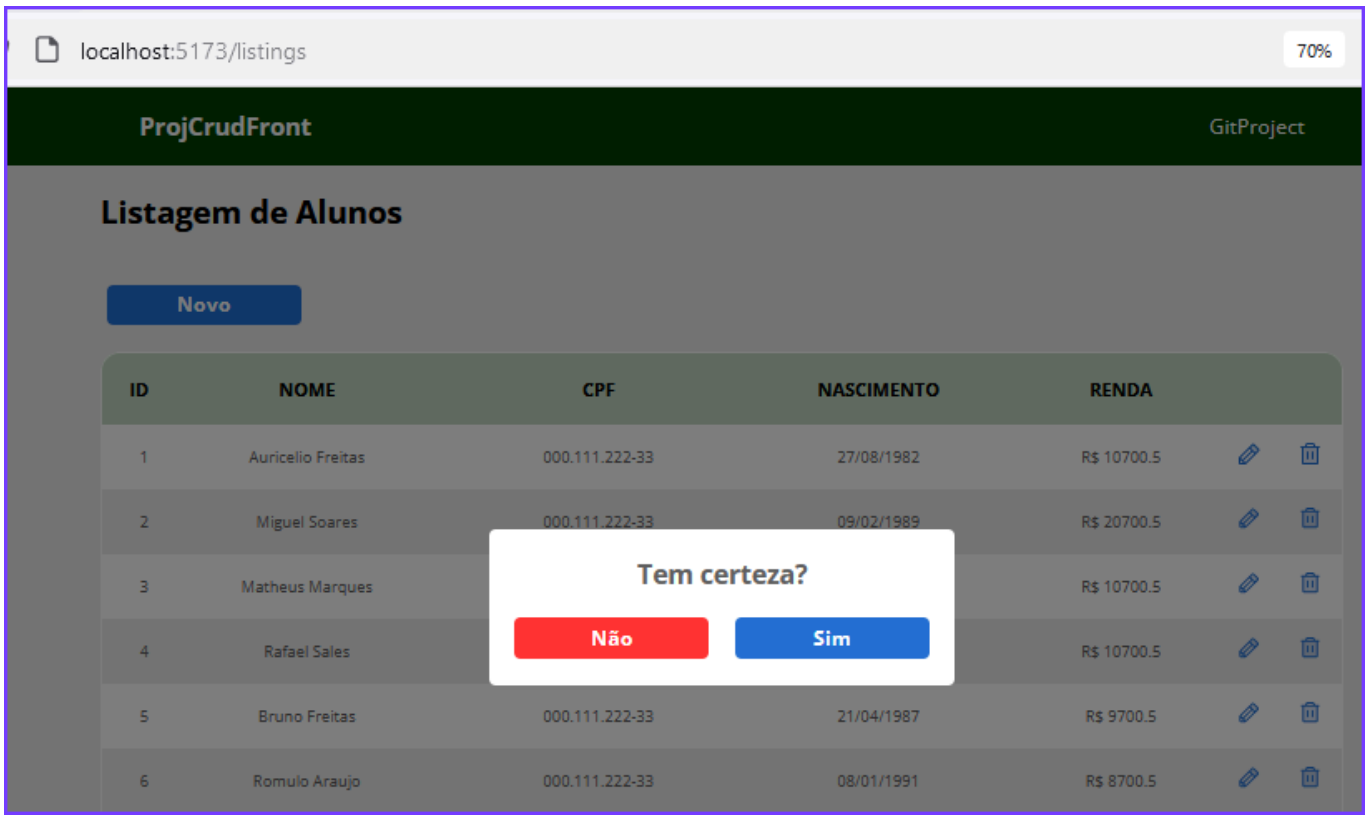
```
47
48  export function deleteRequest(id: number) {
49      const config : AxiosRequestConfig = {
50          method: "DELETE",
51          url: `/students/${id}`,
52      }
53
54      return requestBackend(config)
55  }
```

Implementar a função handleDialogConfirmationAnswer, no Listing

src > pages > Listing > index.tsx > ...

```
52
53      function handleDialogConfirmationAnswer(answer: boolean, studentId: number) {
54          if (answer) {
55              studentService.deleteRequest(studentId)
56                  .then(() => {
57                      setStudents([]);
58                      window.location.reload()
59                  })
60          }
61          setDialogConfirmationData({ ...dialogConfirmationData, visible: false});
62      }
```

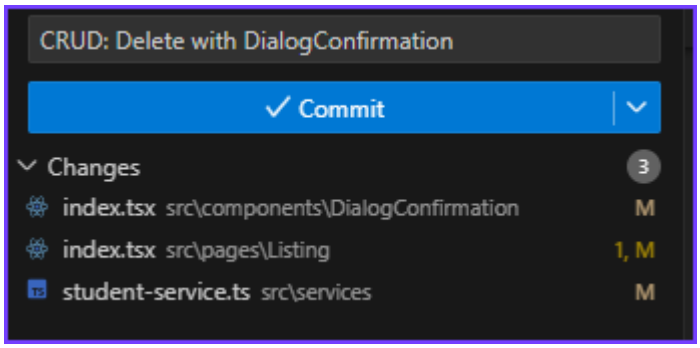
Visualização web



NOTA: *Ao clicar em "Não" a ação será cancelada.*

NOTA: *Ao clicar em "Sim" será deletado o aluno referente ao ID. Neste caso, foi deletado o aluno de ID = 2.*

GitHub-19



CONTROLE DE FORMULÁRIO

***NOTA**: O objetivo desta etapa é implementar os controles do formulário para inserção e edição de aluno, onde os campos devem ser preenchidos, observando os critérios de validação.*

O QUE SERÁ ABORDADO?

- VALIDAÇÃO DIRETAMENTE NO FRONTEND, SEM REALIZAR REQUISIÇÕES NO BACKEND
- REGRAS DE VALIDAÇÕES E EXPRESSÕES REGULARES
- EXIBIÇÃO DE MENSAGEM PARA CADA CAMPO
- COMPORTAMENTO SUJO DE UM CAMPO, OU SEJA, AO SAIR DE UM CAMPO SEM REALIZAR A VALIDAÇÃO NECESSÁRIA.

IMPLEMENTAR A FUNÇÃO VALIDATE

- **Objetivo:** gerar um novo objeto de formulário contendo o campo "invalid" (que pode valer "true" ou "false") para o campo de formulário cujo nome é "name".

Criar a função validate, no forms do utils

src > utils > forms.ts > ...

```
23  export function validate(inputs: any, name: string) {
24
25      const isInvalid = !inputs[name].validation(inputs[name].value);
26      return { ...inputs, [name]: { ...inputs[name], invalid: isInvalid.toString() }};
27  }
```

Implementar um teste com valor “não positivo”, no campo “income”, do NewForm

src > pages > NewForm > index.tsx > ...

```
54          income: {
55              value: -1,
56              id: "income",
57              name: "income",
58              type: "double",
59              placeholder: "Salário",
60              validation: function(value: any) {
61                  return Number(value) > 0;
62              },
63              message: "Favor informar um valor positivo"
64          },
65      });
```

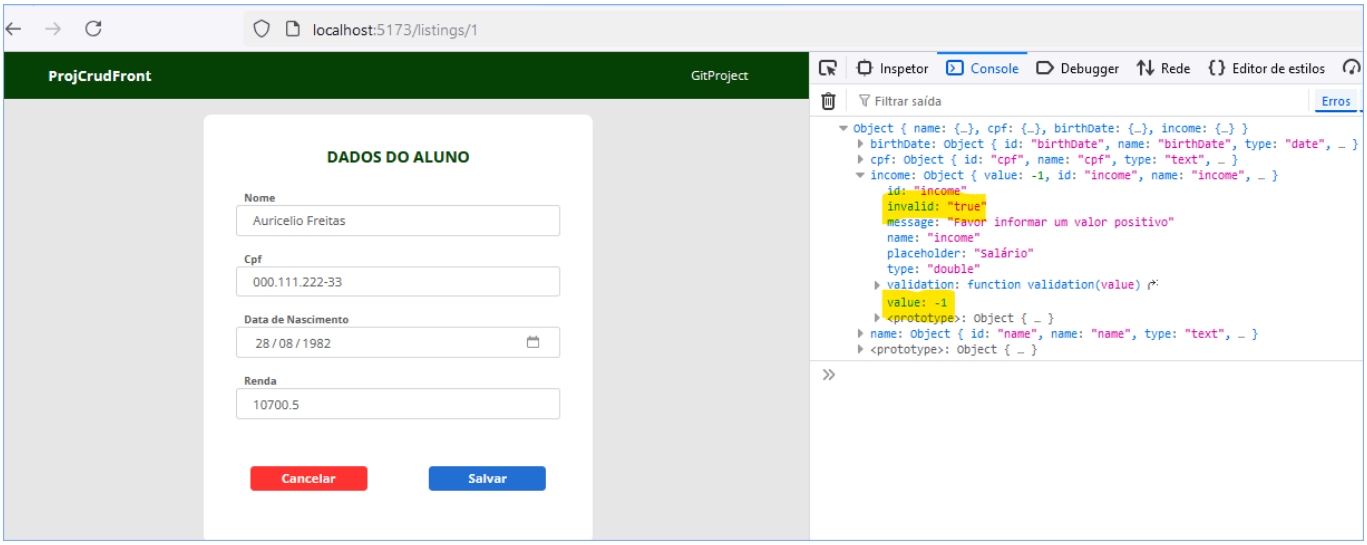
***NOTA**: O “value” do validation pode ser qualquer nome… não tem relação com o campo value.*

Implementar uma chamada para o teste

src > pages > NewForm > index.tsx > ...

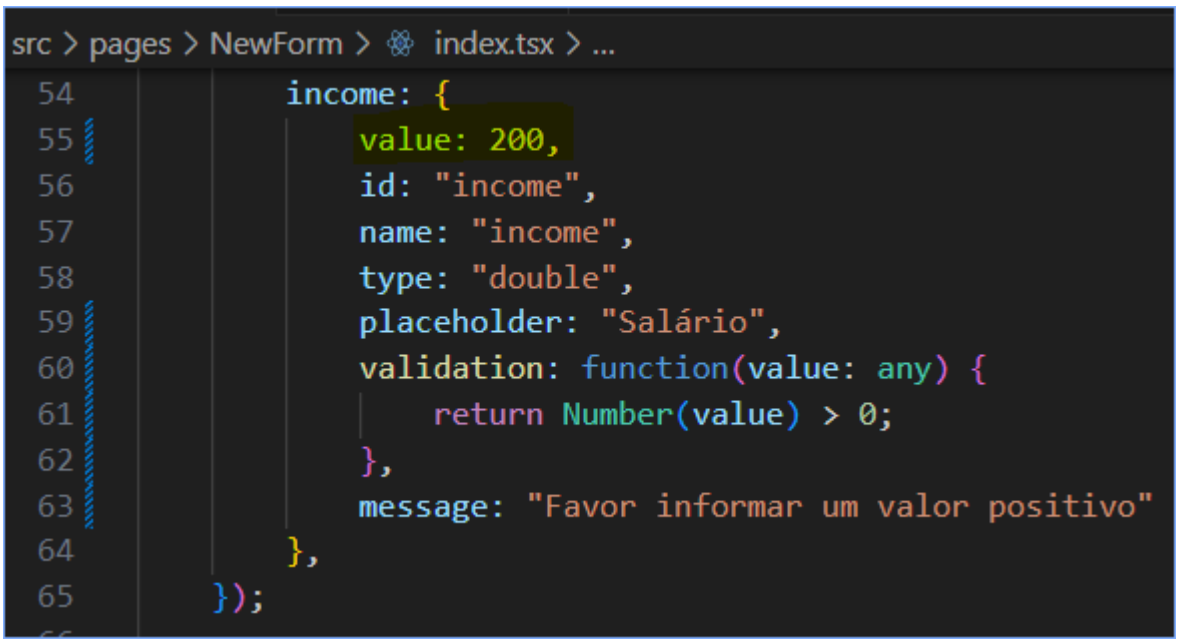
```
18      useEffect(() => {
19
20          const obj = forms.validate(formData, "income");
21          console.log(obj);
22
23          if (isEditing) {
24              studentService.findById(Number(params.studentId))
25              .then(response => {
26                  const newFormData = forms.updateAll(formData, response.data);
27                  setFormData(newFormData);
28              })
29          }
30      }, [])
```

Visualização web



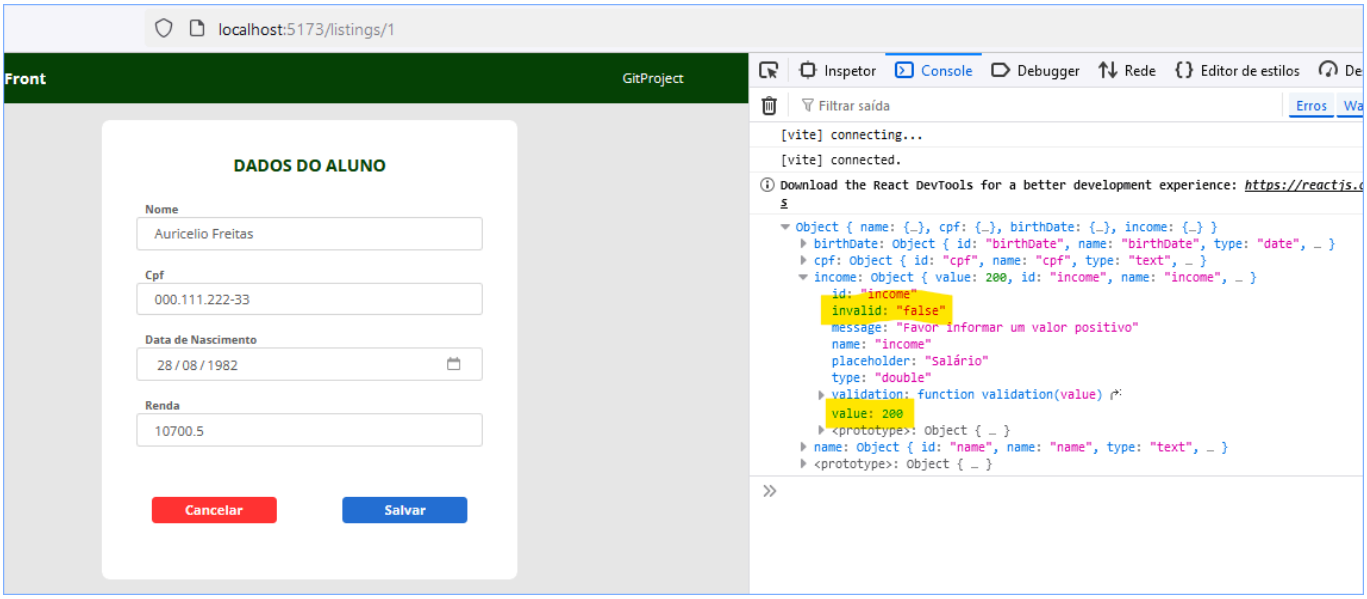
***NOTA**: Pode clicar para editar qualquer aluno. Basta abrir o "Inspecionar" e ir em "console".*

Realizar teste com valor "positivo", no ProductForm



***NOTA**: Este teste é apenas para visualizar a renderização.*

Visualização web



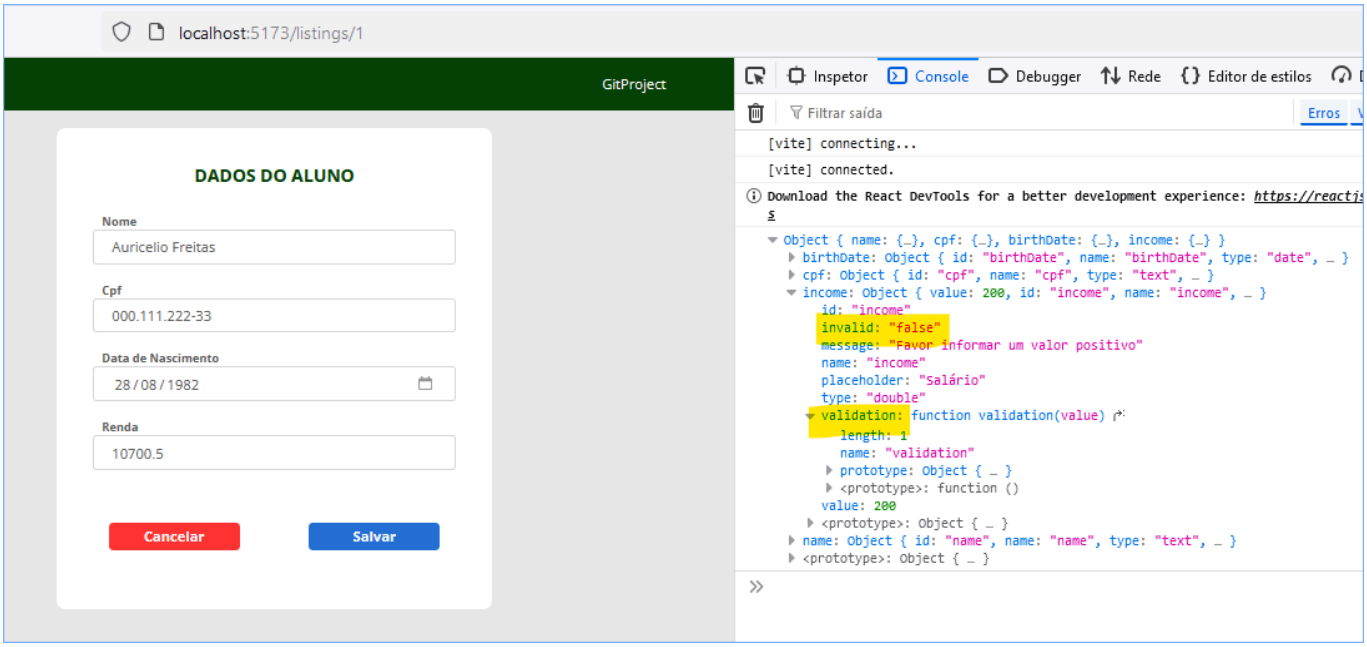
Implementar condicional para função validate, no forms

src > utils > forms.ts > ...

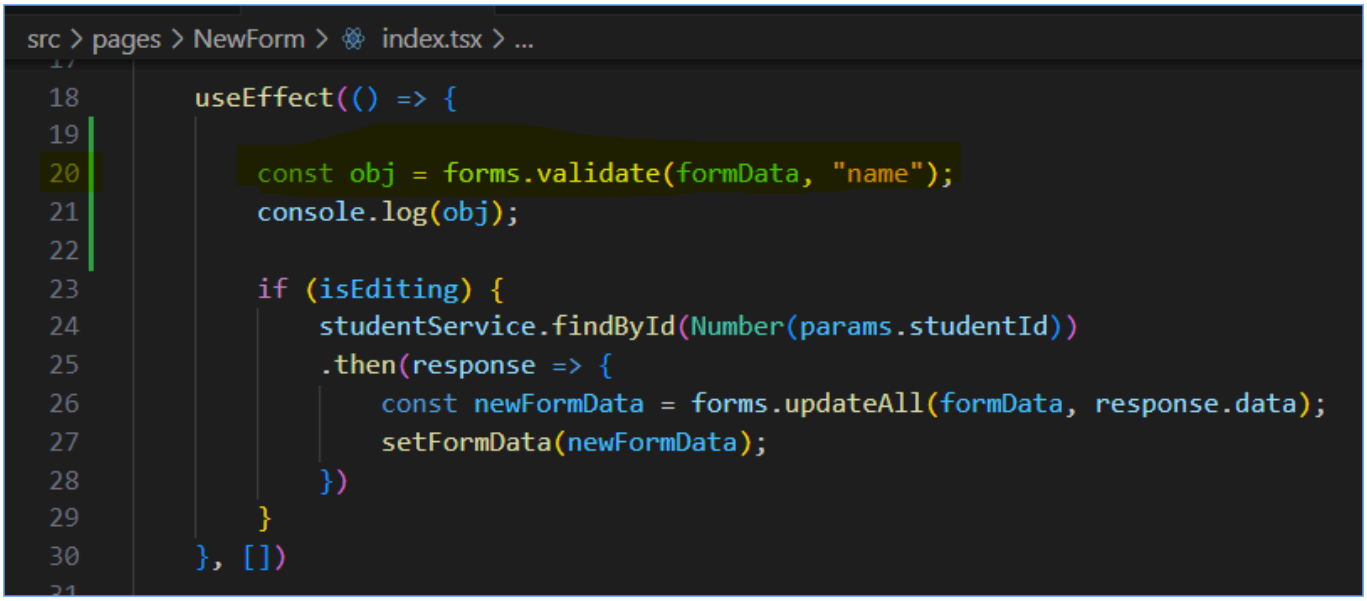
```
23  export function validate(inputs: any, name: string) {
24
25      if (!inputs[name].validation) {
26          return inputs
27      }
28
29      const isInvalid = !inputs[name].validation(inputs[name].value);
30      return { ...inputs, [name]: { ...inputs[name], invalid: isInvalid.toString() }};
31  }
```

NOTA: *A renderização só será realizada se houver uma implementação com o validation.*

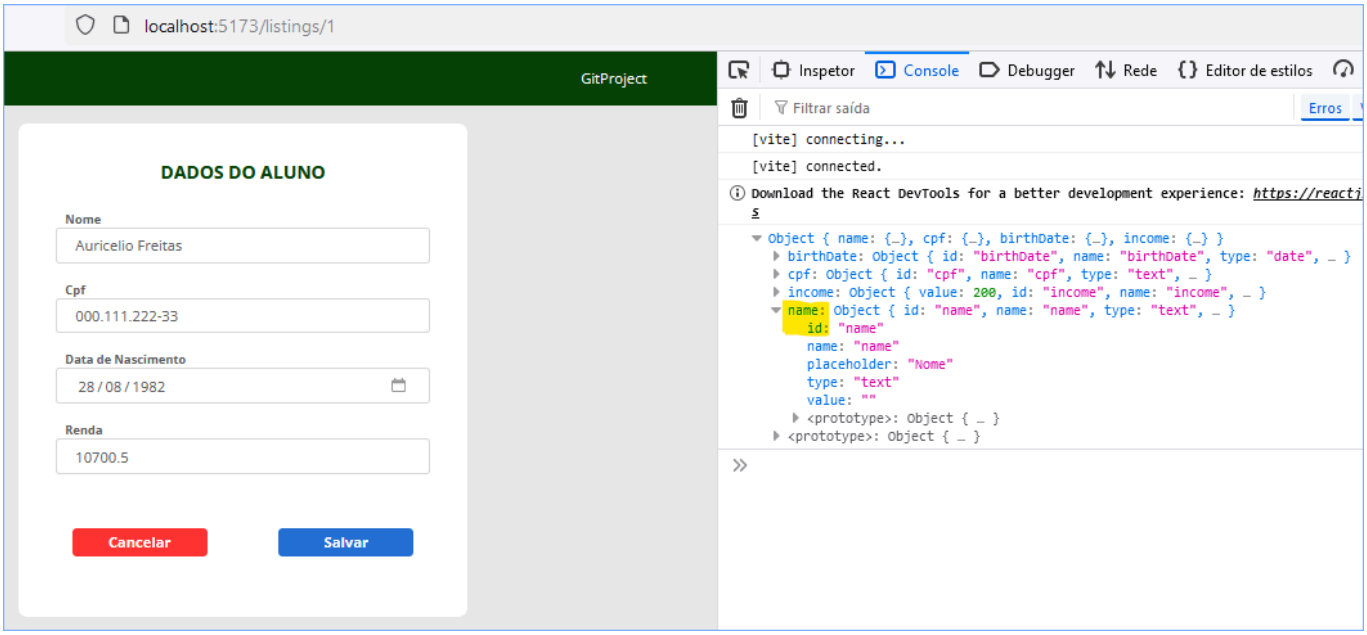
Visualização web



Testar com um campo sem o validation



Visualização web

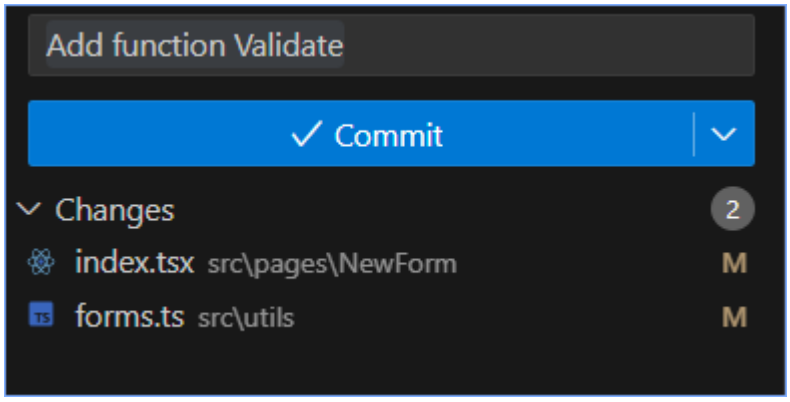


NOTA: Só irá renderizar, caso a função "validation" exista no campo.

Remover os testes anteriores

```
src > pages > NewForm > index.tsx > ...
54          income: {
55              value: 0,
56              id: "income",
57              name: "income",
58              type: "double",
59              placeholder: "Salário",
60              validation: function(value: any) {
61                  return Number(value) > 0;
62              },
63              message: "Favor informar um valor positivo"
64          },
65      });
66
```

GitHub-20



IMPLEMENTAR A VALIDAÇÃO

Verificar implementação do Componente FormInput

src > components > FormInput > index.tsx > ...

```tsx
export default function FormInput(props: any) {

    const { validation, invalid, ...inputProps } = props;

    return (
        <input { ...inputProps } data-invalid={invalid} />
    );
}
```

NOTA: *Já havíamos deixada implementada em momentos anteriores.*

Refatorar a função handleInputChange, no NewForm

src > pages > NewForm > index.tsx > ...

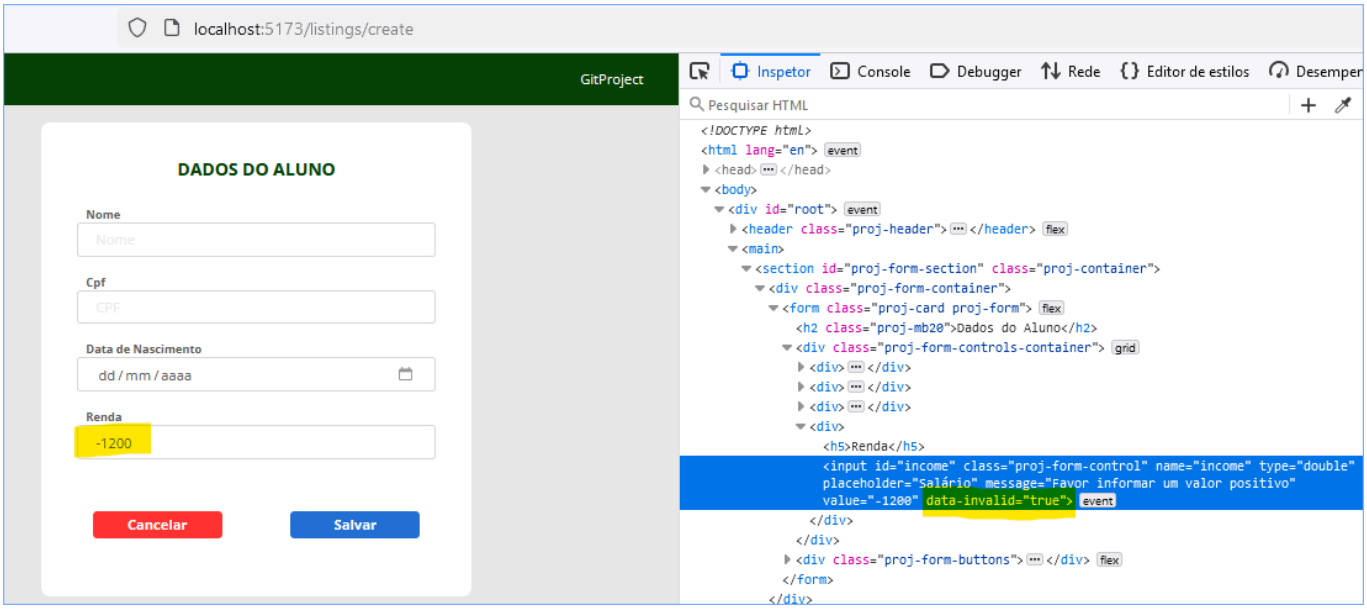
```tsx
function handleInputChange(event: any) {
    const dataUpdated = forms.update(formData, event.target.name, event.target.value);
    setFormData(dataUpdated);
}
```

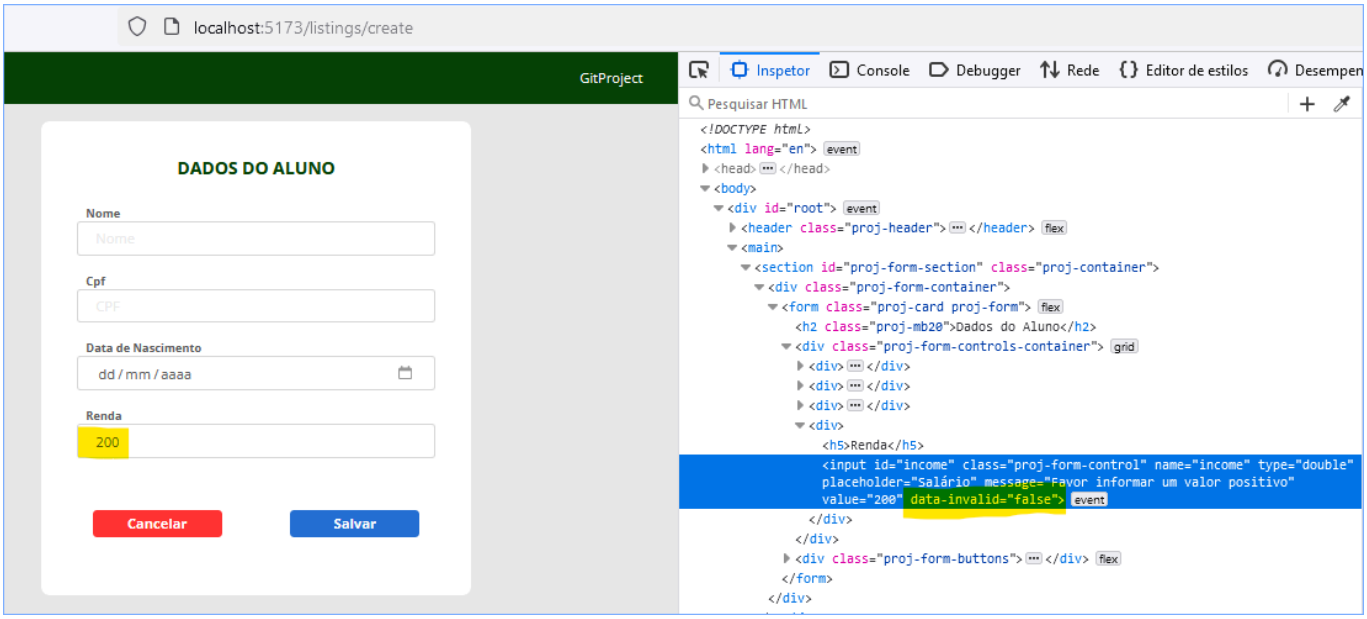
Validar os dados atualizados

src > pages > NewForm > index.tsx > ...

```tsx
function handleInputChange(event: any) {
    const dataUpdated = forms.update(formData, event.target.name, event.target.value);
    const dataValidated = forms.validate(dataUpdated, event.target.name);
    setFormData(dataValidated);
}
```

Visualização web





***NOTA**: Neste momento, o campo Renda, já está sendo verificado se o valor é válido ou não, em tempo real.*

Implementar o estilo CSS da borda

```
src > pages > NewForm > styles.css > ...
58  .proj-form-control[data-invalid="true"] {
59    border: 1px solid var(--proj-color-error);
60  }
```

Visualização web

***NOTA**: Neste momento a borda já fica vermelha, caso seja digitado um valor invalido.*

Implementar o validation em todos os campos

src > pages > NewForm > index.tsx > ...

```tsx
const [formData, setFormData] = useState<any>({
    name: {
        value: "",
        id: "name",
        name: "name",
        type: "text",
        placeholder: "Nome",
        validation: function(value: any) {
            return value.length >= 3 && value.length <= 50;
        },
        message: "Favor informar um nome entre 3 e 50 caracteres"
    },
    cpf: {
        value: "",
        id: "cpf",
        name: "cpf",
        type: "text",
        placeholder: "CPF",
        validation: function(value: any) {
            return /^(([0-9]{3}.?[0-9]{3}.?[0-9]{3}-?[0-9]{2}))$/.test(value);
        },
        message: "Favor informar um CPF válido"

    },
    birthDate: {
        value: "",
        id: "birthDate",
        name: "birthDate",
        type: "date",
        placeholder: "Data de Nascimento",
        validation: function(value: any) {
            return /^(([0-9]{4}.?[0-9]{2}.?[0-9]{2}))$/.test(value);
        },
        message: "Favor informar uma data válida"
    },
    income: {
        value: 0,
        id: "income",
        name: "income",
        type: "double",
        placeholder: "Salário",
        validation: function(value: any) {
            return Number(value) >= 0;
        },
        message: "Favor informar um valor positivo"
    },
});
```

Implementar a mensagem de erro, no FormInpu

src > pages > NewForm > index.tsx > ...

```
111                         <FormInput
112                             { ...formData.name }
113                             className="proj-form-control"
114                             onChange={handleInputChange}
115                         />
116                         <div className="proj-form-error">{formData.name.message}</div>
117                     </div>
118                     <div>
119                         <h5>Cpf</h5>
120                         <FormInput
121                             { ...formData.cpf }
122                             className="proj-form-control"
123                             onChange={handleInputChange}
124                         />
125                         <div className="proj-form-error">{formData.cpf.message}</div>
126                     </div>
127                     <div>
128                         <h5>Data de Nascimento</h5>
129                         <FormInput
130                             { ...formData.birthDate }
131                             className="proj-form-control"
132                             onChange={handleInputChange}
133                         />
134                         <div className="proj-form-error">{formData.birthDate.message}</div>
135                     </div>
136                     <div>
137                         <h5>Renda</h5>
138                         <FormInput
139                             { ...formData.income }
140                             className="proj-form-control"
141                             onChange={handleInputChange}
142                         />
143                         <div className="proj-form-error">{formData.income.message}</div>
144                     </div>
145                 </div>
```

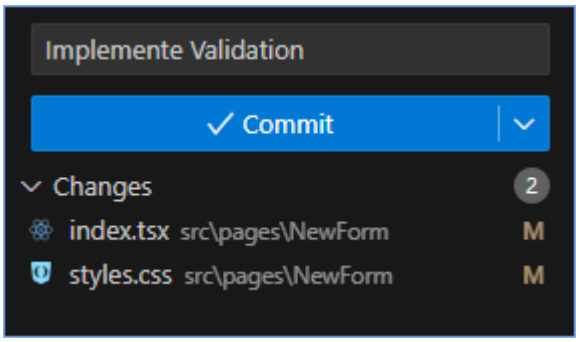
Implementar o estilo

src > pages > NewForm > styles.css > ...

```
57
58  .proj-form-control[data-invalid="true"] {
59    border: 1px solid var(--proj-color-error);
60  }
61
62  .proj-form-control[data-invalid="true"] ~ div {
63    display: unset;
64  }
65
66  .proj-form-error {
67    color: var(--proj-color-error);
68    font-size: 12px;
69    padding-left: 4px;
70    display: none;
71  }
```

Visualização web

GitHub-21


Implemente Validation
Commit
Changes
2
index.tsx src\pages\NewForm
M
styles.css src\pages\NewForm
M

IMPLEMENTAR O COMPORTAMENTO SUJO (DIRTY)

***NOTA**: Nesta etapa iremos implementar o conceito de comportamento sujo (DIRTY). A referida implementação é para os erros de validação nos campos, ou seja, apenas aparecer os referidos erros quando sair do campo, sem realizar o que lhe é solicitado na validação.*

Simulação

***NOTA**: Neste exemplo, a validação é para digitar entre 3 e 50 caracteres, porém, ainda na digitação, o erro já aparece. Pensando em UX, para uma melhor experiência de usuário, o erro só deve aparecer quando sair do campo, sem realizar os critérios de validação.*

Implementar o onTurnDirty, no componente FormInput

src > components > FormInput > index.tsx > ...

```tsx
export default function FormInput(props: any) {

    const { validation, invalid, onTurnDirty, ...inputProps } = props;

    return (
        <input { ...inputProps } data-invalid={invalid} />
    );
}
```

***NOTA**: Apenas mostrar a mensagem de erro quando sair do campo e este esteja inválido.*

Implementar a função toDirty, no utilitário forms

***OBJETIVO**: Gerar um novo objeto de formulário contendo o campo "dirty" (vando "true" ou "false") para o campo de formulário cujo nome é "name".*

src > utils > forms.ts > ...

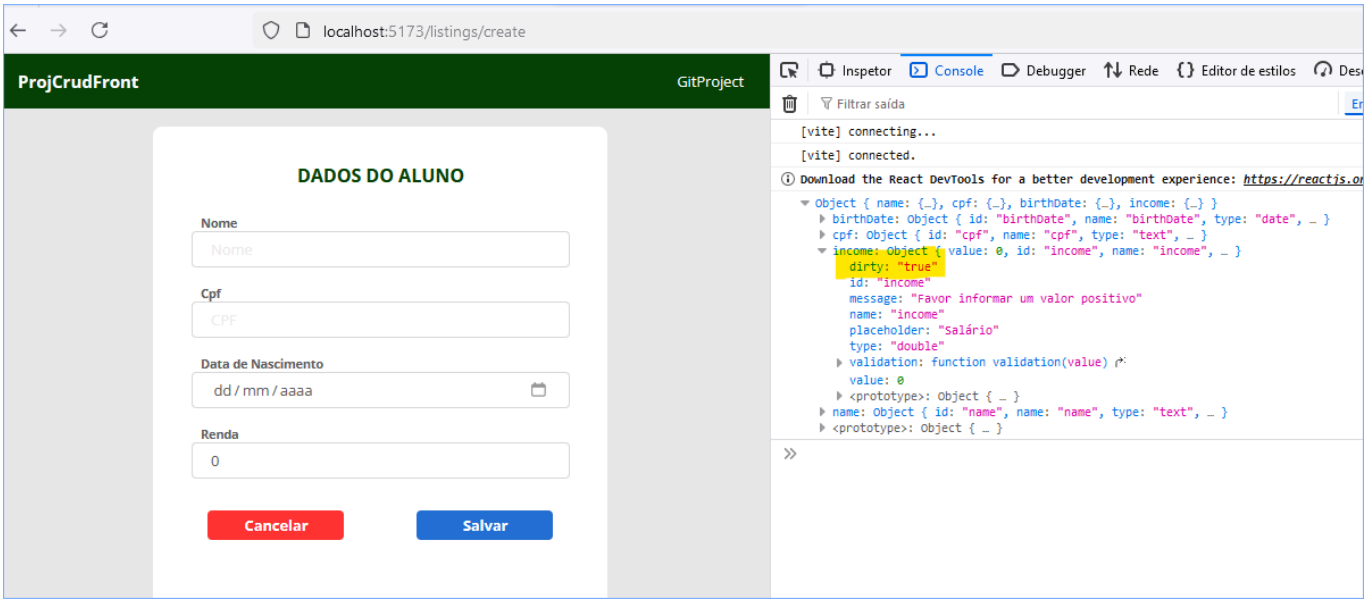
```ts

export function toDirty(inputs: any, name: string) {
    return { ...inputs, [name]: {...inputs[name], dirty: "true"} }
}

```

Implementar um teste na página NewForm

src > pages > NewForm > index.tsx > ...

```
18      useEffect(() => {
19
20          const result = forms.toDirty(formData, "income");
21          console.log(result);
22
23          if (isEditing) {
24              studentService.findById(Number(params.studentId))
25              .then(response => {
26                  const newFormData = forms.updateAll(formData, response.data);
27                  setFormData(newFormData);
28              })
29          }
30      }, [])
```

Visualização web



Implementar o evento onBlur, no componente FormInput

src > components > FormInput > index.tsx > ...

```tsx
export default function FormInput(props: any) {

    const {
        validation,
        invalid = "false",
        dirty = "false",
        onTurnDirty,
        ...inputProps
    } = props;

    function handleBlur() {
        onTurnDirty(props.name);
    }

    return (
        <input
            { ...inputProps }
            onBlur={handleBlur}
            data-invalid={invalid}
            data-dirty={dirty}
        />
    );
}
```

Implementar o onTurnDirty no NewForm

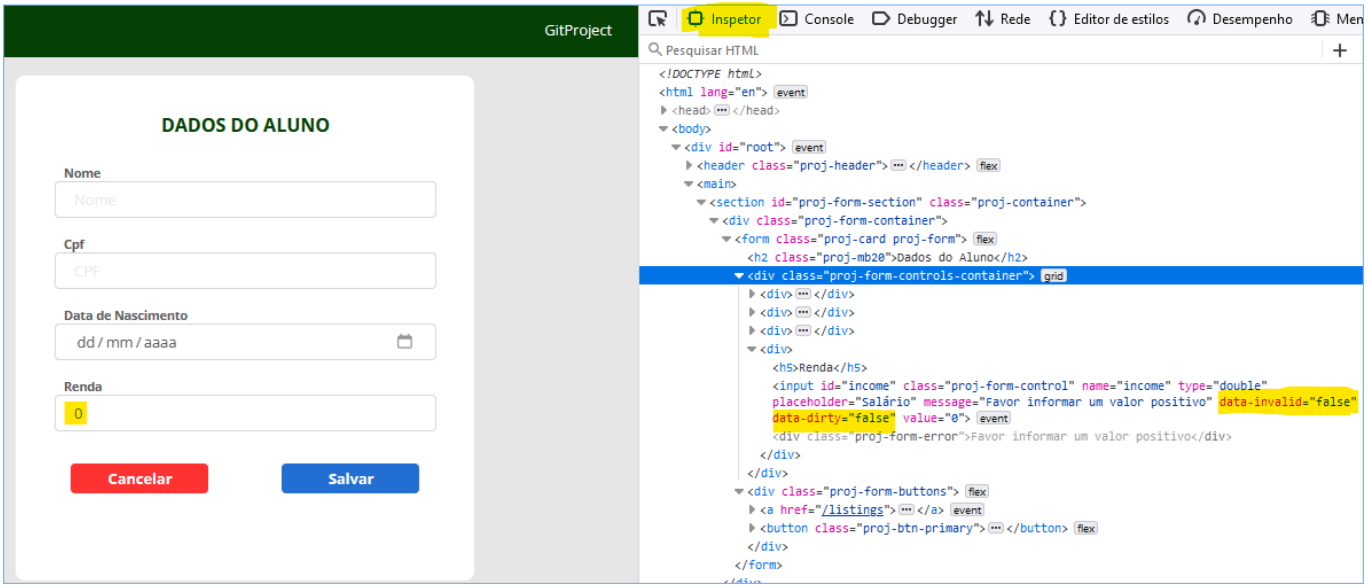
src > pages > NewForm > index.tsx > NewForm

```
110     return (
111         <main>
112             <section id="proj-form-section" className="proj-container">
113                 <div className="proj-form-container">
114                     <form className="proj-card proj-form" onSubmit={handleSubmit}>
115                         <h2 className="proj-mb20">Dados do Aluno</h2>
116                         <div className="proj-form-controls-container">
117                             <div>
118                                 <h5>Nome</h5>
119                                 <FormInput
120                                     { ...formData.name }
121                                     className="proj-form-control"
122                                     onTurnDirty={handleTurnDirty}
123                                     onChange={handleInputChange}
124                                 />
125                                 <div className="proj-form-error">{formData.name.message}</div>
126                             </div>
127                             <div>
128                                 <h5>Cpf</h5>
129                                 <FormInput
130                                     { ...formData.cpf }
131                                     className="proj-form-control"
132                                     onTurnDirty={handleTurnDirty}
133                                     onChange={handleInputChange}
134                                 />
135                                 <div className="proj-form-error">{formData.cpf.message}</div>
136                             </div>
137                             <div>
138                                 <h5>Data de Nascimento</h5>
139                                 <FormInput
140                                     { ...formData.birthDate }
141                                     className="proj-form-control"
142                                     onTurnDirty={handleTurnDirty}
143                                     onChange={handleInputChange}
144                                 />
145                                 <div className="proj-form-error">{formData.birthDate.message}</div>
146                             </div>
147                             <div>
148                                 <h5>Renda</h5>
149                                 <FormInput
150                                     { ...formData.income }
151                                     className="proj-form-control"
152                                     onTurnDirty={handleTurnDirty}
153                                     onChange={handleInputChange}
154                                 />
155                                 <div className="proj-form-error">{formData.income.message}</div>
156                             </div>
157                         </div>
```

Implementar a função para o handleTurnDirty

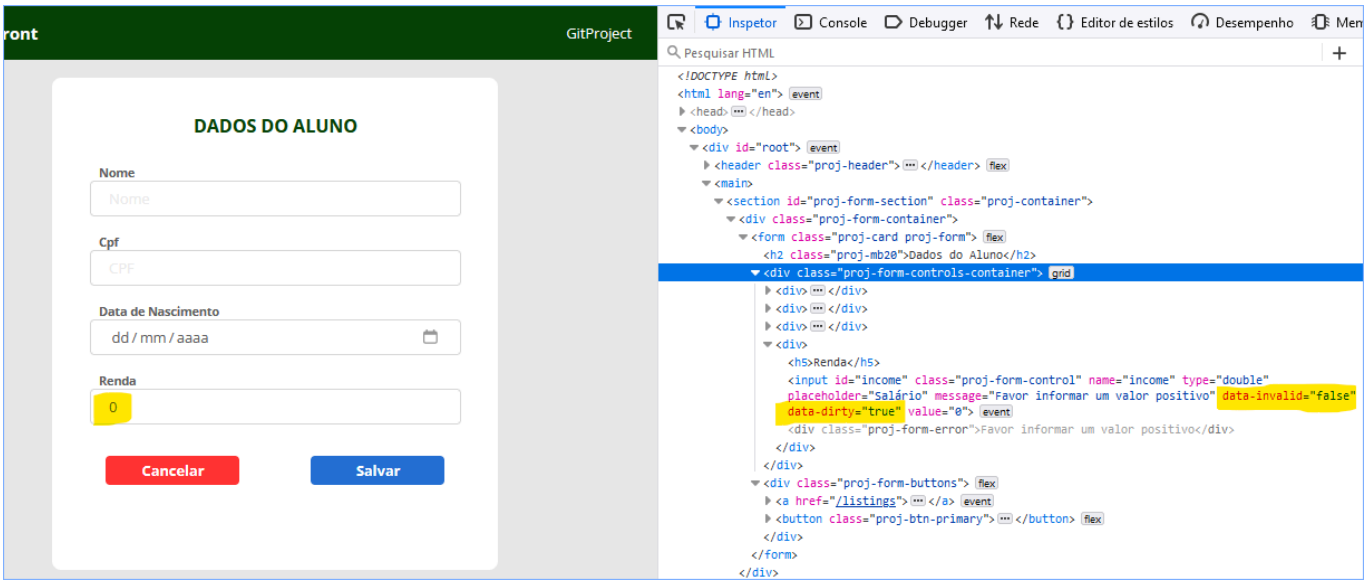
```
src > pages > NewForm > index.tsx > ...
105     function handleTurnDirty(name: string) {
106         const newFormData = forms.toDirty(formData, name);
107         setFormData(newFormData);
108     }
109
110     return (
```

Visualização web



NOTA: *Por padrão o "data-invalid" e o "data-dirty" são falsos.*

Simulação



NOTA: *Ao entrar e sair do campo renda, o "data-dirty" passou a ser "true", ou seja, neste momento já conseguimos saber se o campo está sujo ou não*

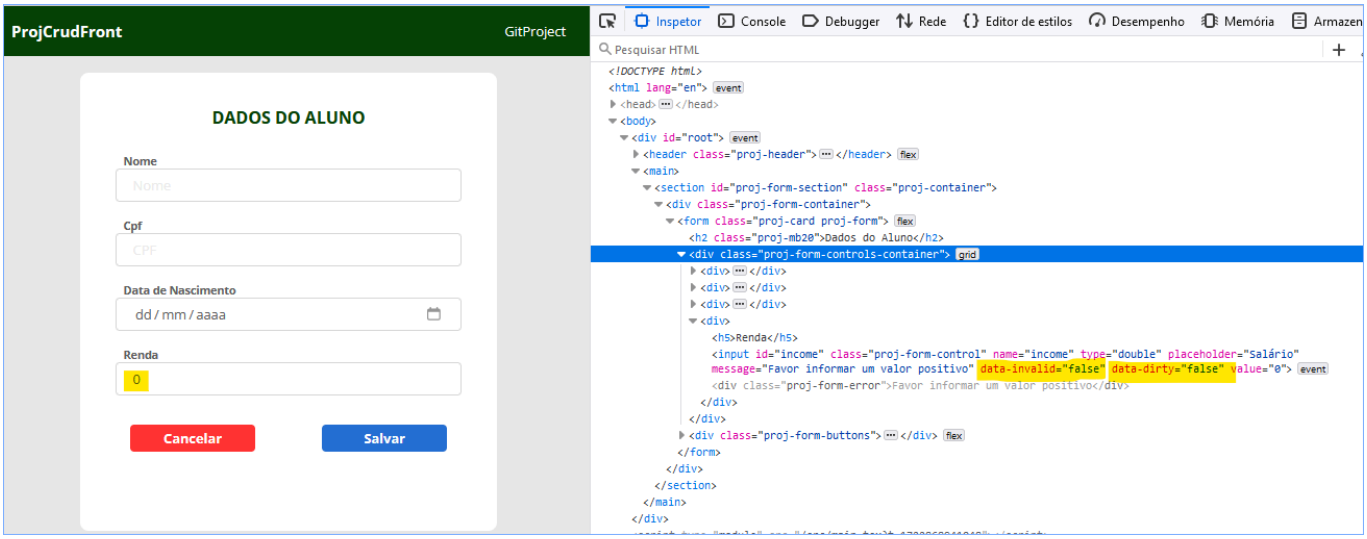
Implementar o CSS para o data-dirty

src > pages > NewForm > styles.css > ...

```
57
58  .proj-form-control[data-dirty="true"][data-invalid="true"] {
59    border: 1px solid var(--proj-color-error);
60  }
61
62  .proj-form-control[data-dirty="true"][data-invalid="true"] ~ div {
63    display: unset;
64  }
```

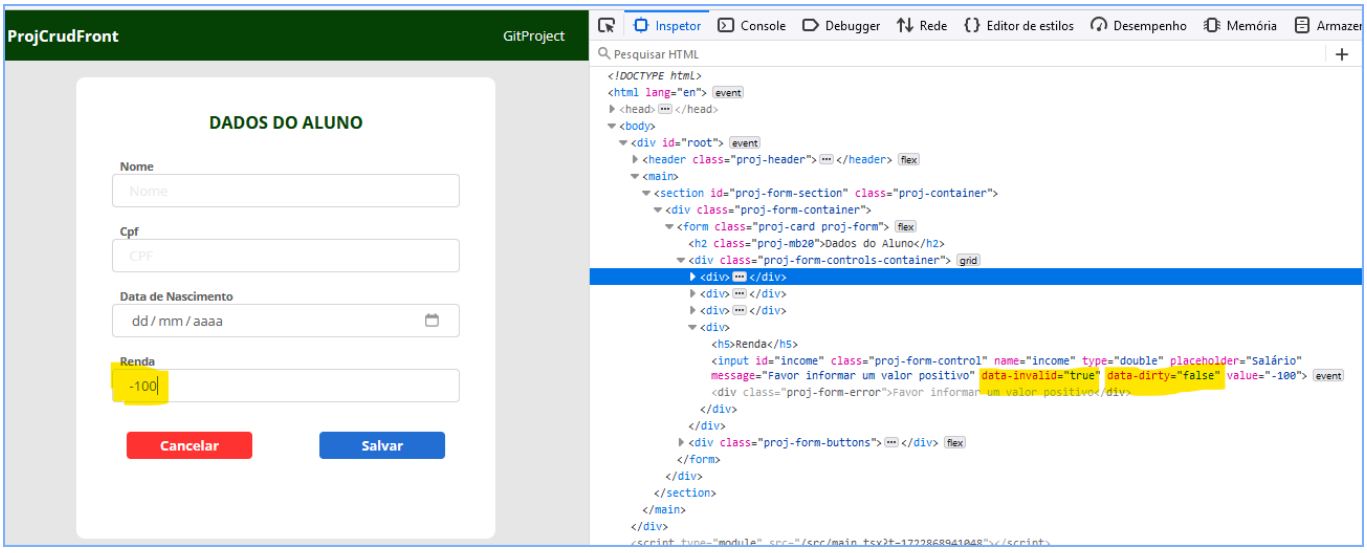
Simulação

Default



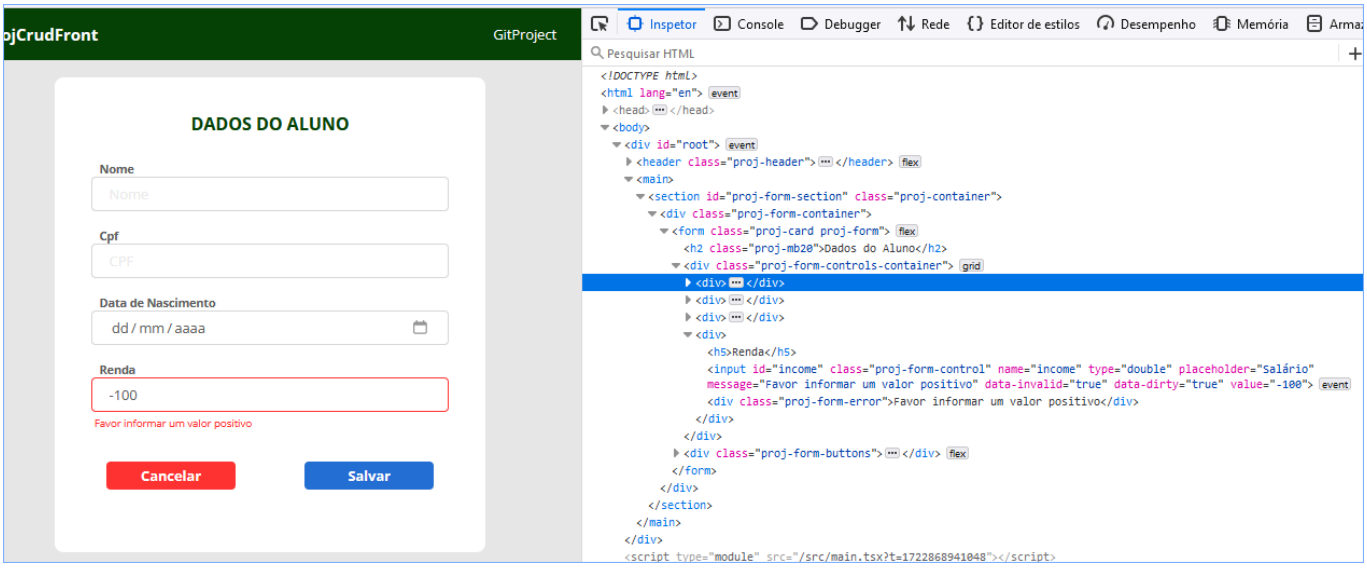
NOTA: *O valor default é o data-invalid=false e o data-dirty=false.*

Digitar um valor iválido no campo



NOTA: *Ao digitar um valor inválido no campo, o "data-invalid" passa a valer "true", porém não aparece a mensagem de erro, pois o "data-dirty" continua "false".*

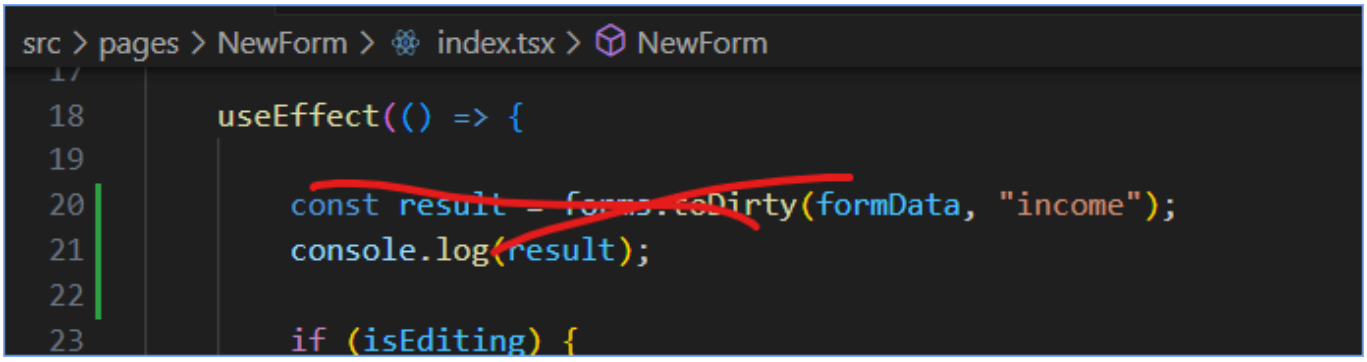
Digitar um valor iválido e sair do campo



NOTA: *Ao digitar um valor inválido no campo, o "data-invalid" passa a valer "true", ao sair do campo, deixando o valor inválido, o "data-dirty" passa a valer "true" e a mensagem de erro aparece.*

Melhorias no código

Remover testes, no NewForm



Criar uma função auxiliar para o Update e Validate, no forms

src > utils > forms.ts > ...

```
37  export function updateAndValidate(inputs: any, name: string, newValue: any) {
38      const dataUpdated = update(inputs, name, newValue);
39      return validate(dataUpdated, name);
40  }
```

Criar uma função auxiliar para o Dirty e Validate, no forms

src > utils > forms.ts > ...

```
42  export function dirtyAndValidate(inputs: any, name: string) {
43      const dataDirty = toDirty(inputs, name);
44      return validate(dataDirty, name);
45  }
```

Refatorar o handleInputChange, do NewForm

src > pages > NewForm > index.tsx > ...

```
96      function handleInputChange(event: any) {
97          setFormData(forms.updateAndValidate(formData, event.target.name, event.target.value));
98      }
```

Refatorar o handleTurnDirty, do NewForm

src > pages > NewForm > index.tsx > ...

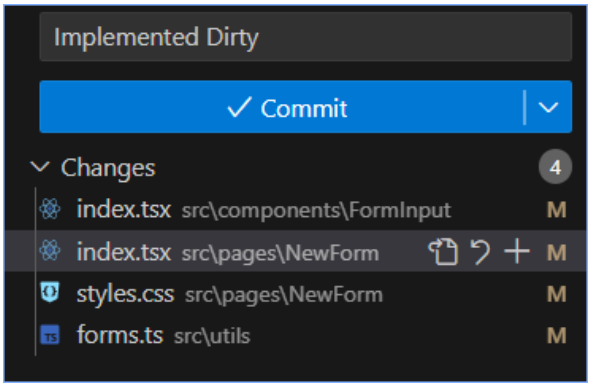
```
100     function handleTurnDirty(name: string) {
101         setFormData(forms.dirtyAndValidate(formData, name));
102     }
```

Visualização web

,

***NOTA**: Neste momento, se optarmos por sair do campo sem preenchimento, o mesmo irá reclamar que precisa ser preenchido corretamente.*

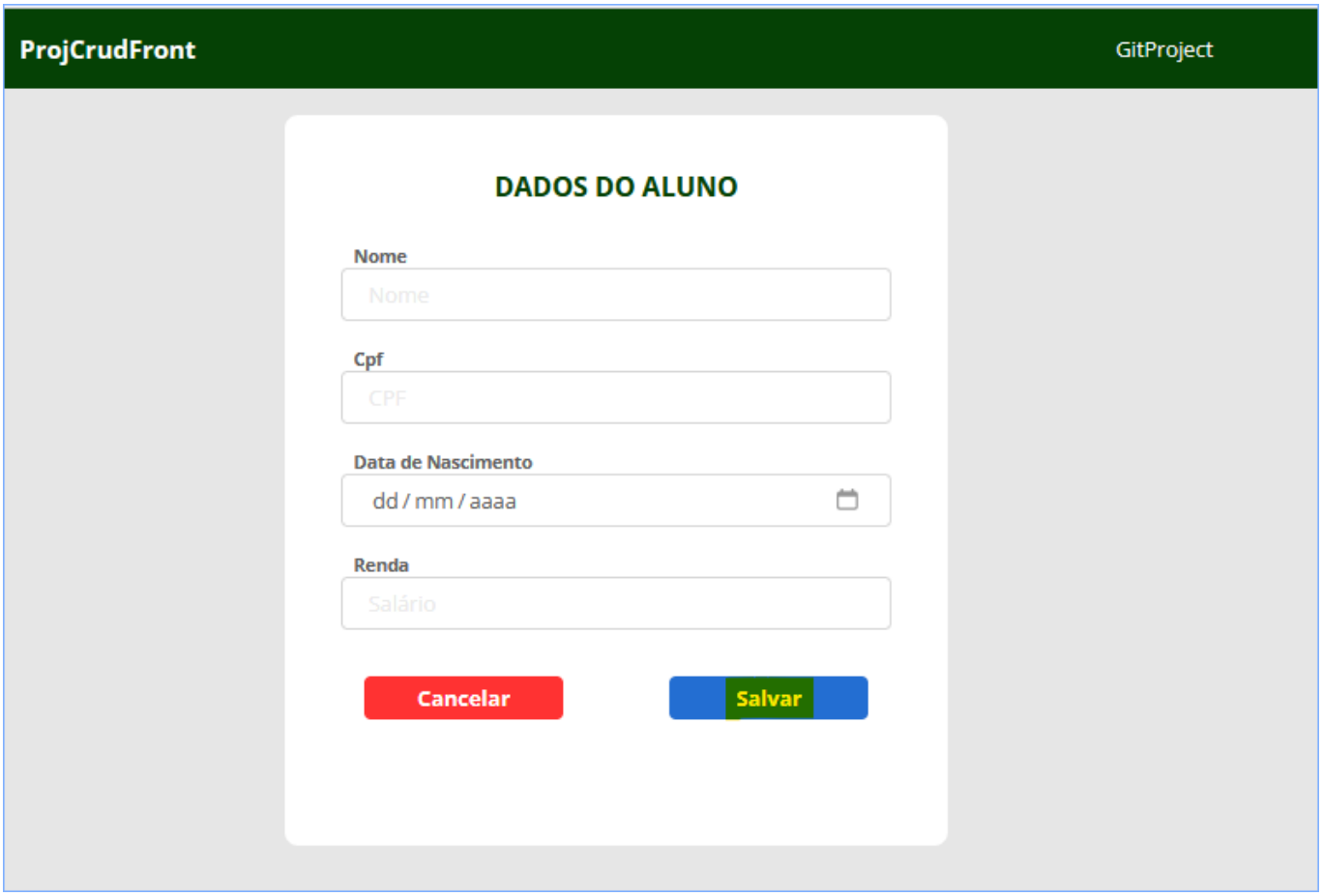
GitHub-22



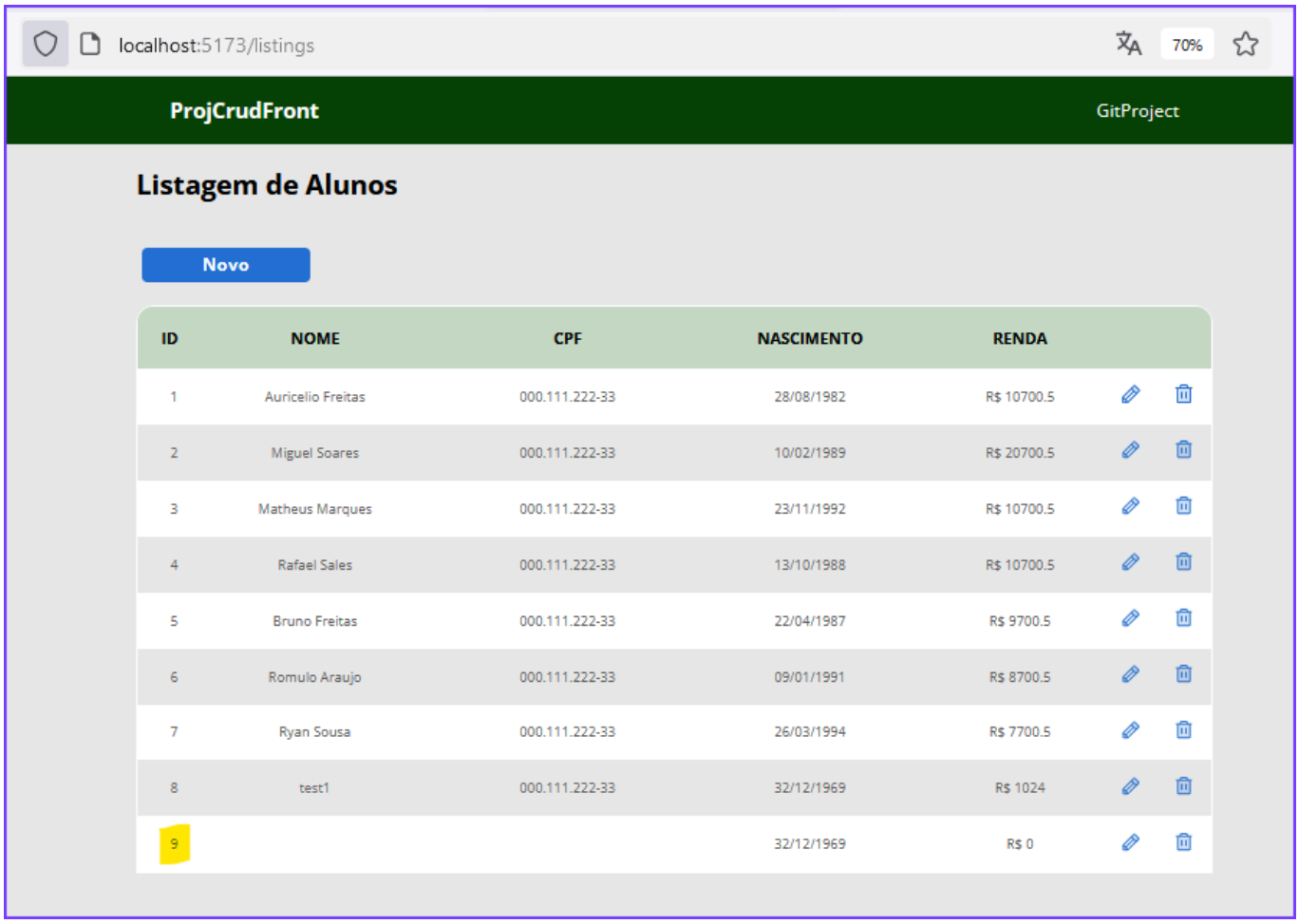
IMPEDIR ENVIO DE CAMPO E FORMULÁRIO EM BRANCO (DIRTY_ALL)

***NOTA**: A implementação é para evitar que o usuário consiga submeter um campo obrigatório em branco ou até mesmo todo o formulário.*

Simulação



***NOTA**: Quando é aberto um novo formulário, sem clicar em nenhum campo, em seguida clica-se em "Salvar", é gerado um objeto vazio.*



NOTA: *Precisamos barrar! Só devemos permitir submeter um novo objeto quando atendidos todos os critérios de validação obrigatórios.*

Implementar a função toDirtyAll, no utilitário forms

***OBJETIVO**: Sujar todos os campos.*

forms.ts M

src > utils > forms.ts > ...

```ts
export function toDirtyAll(inputs: any) {
    const newInputs: any = {};
        for (var name in inputs) {
            newInputs[name] = { ...inputs[name], dirty: "true" };
        }

    return newInputs;
}
```

Implementar a função validateAll, no utilitário forms

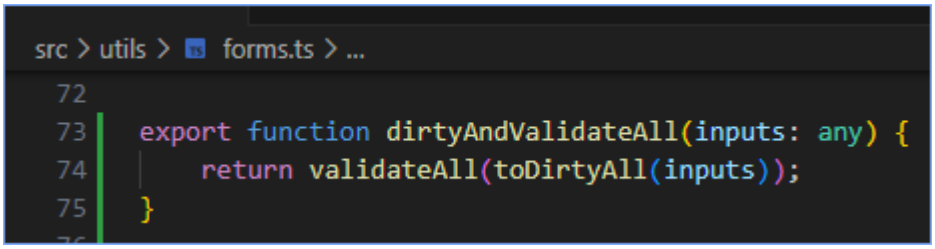
***OBJETIVO**: Validar apenas se existir o atributo "validation".*

src > utils > forms.ts > ...

```ts
export function validateAll(inputs: any) {
    const newInputs: any = {};

        for (var name in inputs) {
            if (inputs[name].validation) {
                const isInvalid = !inputs[name].validation(inputs[name].value);
                newInputs[name] = { ...inputs[name], invalid: isInvalid.toString()}
            }
            else {
                newInputs[name] = { ...inputs[name] };
            }
        }

    return newInputs;
}
```

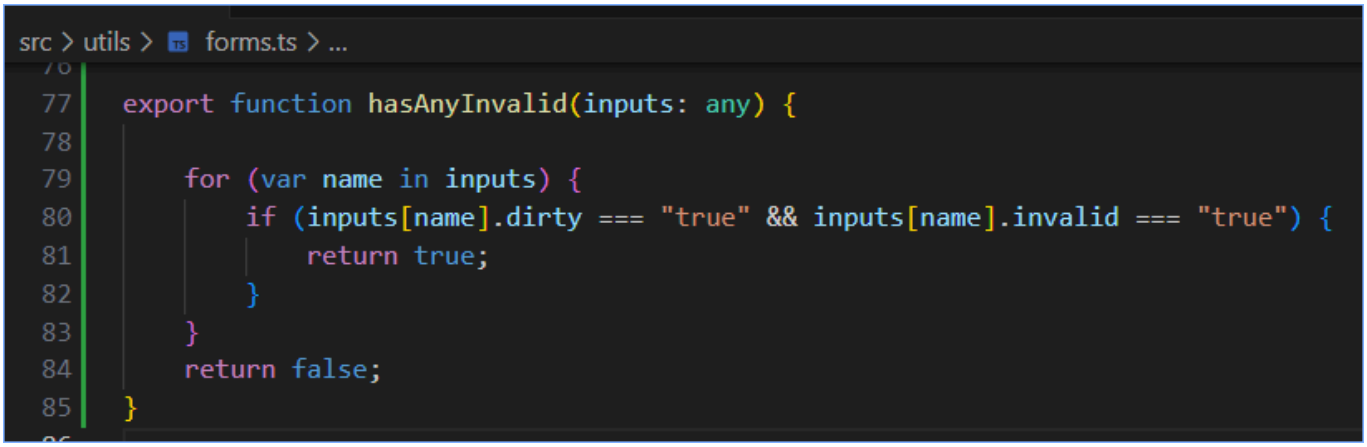
Implementar a função dirtyAndValidateAll, no forms do utils

***OBJETIVO**: Sujar e Validar todos os campos.*



Implementar a função hasAnyInvalid, no forms do utils

***OBJETIVO**: Testar se há algum campo inválido.*



Implementar o formDataValidated, na função handleSubmit, do NewForm

src > pages > NewForm > index.tsx > ...

```
77      function handleSubmit(event: any) {
78          event.preventDefault();
79
80          const formDataValidated = forms.dirtyAndValidateAll(formData);
81          if (forms.hasAnyInvalid(formDataValidated)) {
82              setFormData(formDataValidated);
83              return;
84          }
85
86          const requestBody = forms.toValues(formData);
87          if (isEditing) {
88              requestBody.id = params.studentId;
89          }
90
91          const request = isEditing
92
93              ? studentService.updateRequest(requestBody)
94              : studentService.insertRequest(requestBody)
95
96          request
97              .then(() => {
98                  navigate("/listings");
99              });
100     }
```

Visualização web

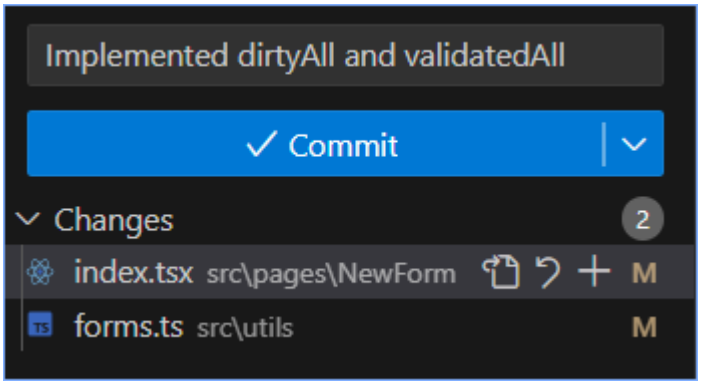
Enviar formulário em branco

***NOTA**: Se o usuário tentar salvar um formulário em branco, todas as dependências serão acionadas.*

Enviar formulário com alguma pendência

***NOTA**: Se houver alguma pendência, o formulário não será submetido.*

GitHub-23


Implemented dirtyAll and validatedAll
Commit
Changes
2
index.tsx src\pages\NewForm
M
forms.ts src\utils
M

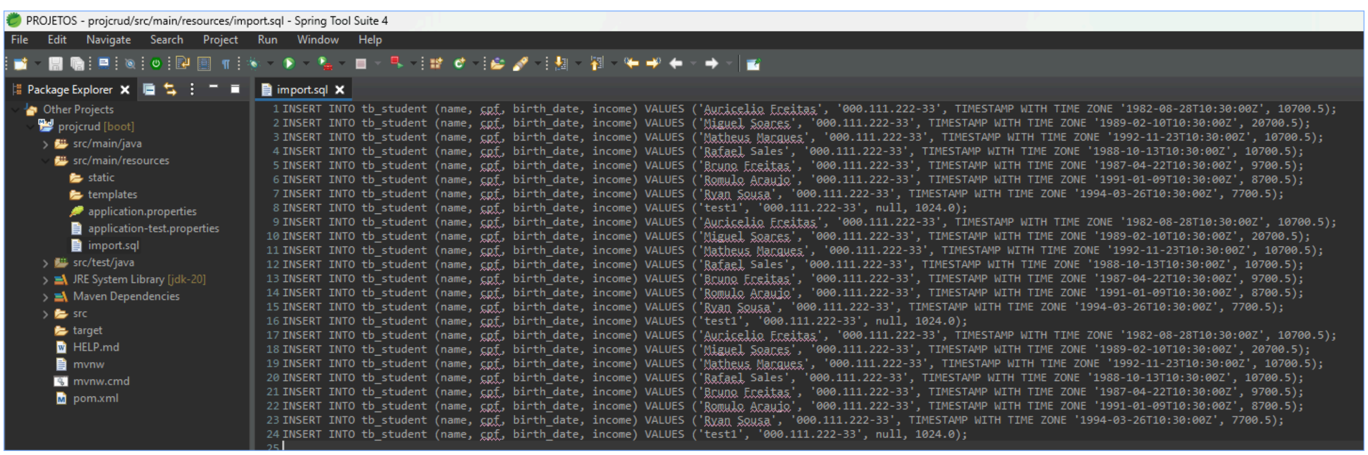
PAGINAÇÃO

IMPLEMENTAR O SEED DO BACKEND

***OBJETIVO**: Implementar o seed do backend com mais registros para que possamos testar a paginação.*

Implementar o import.sql

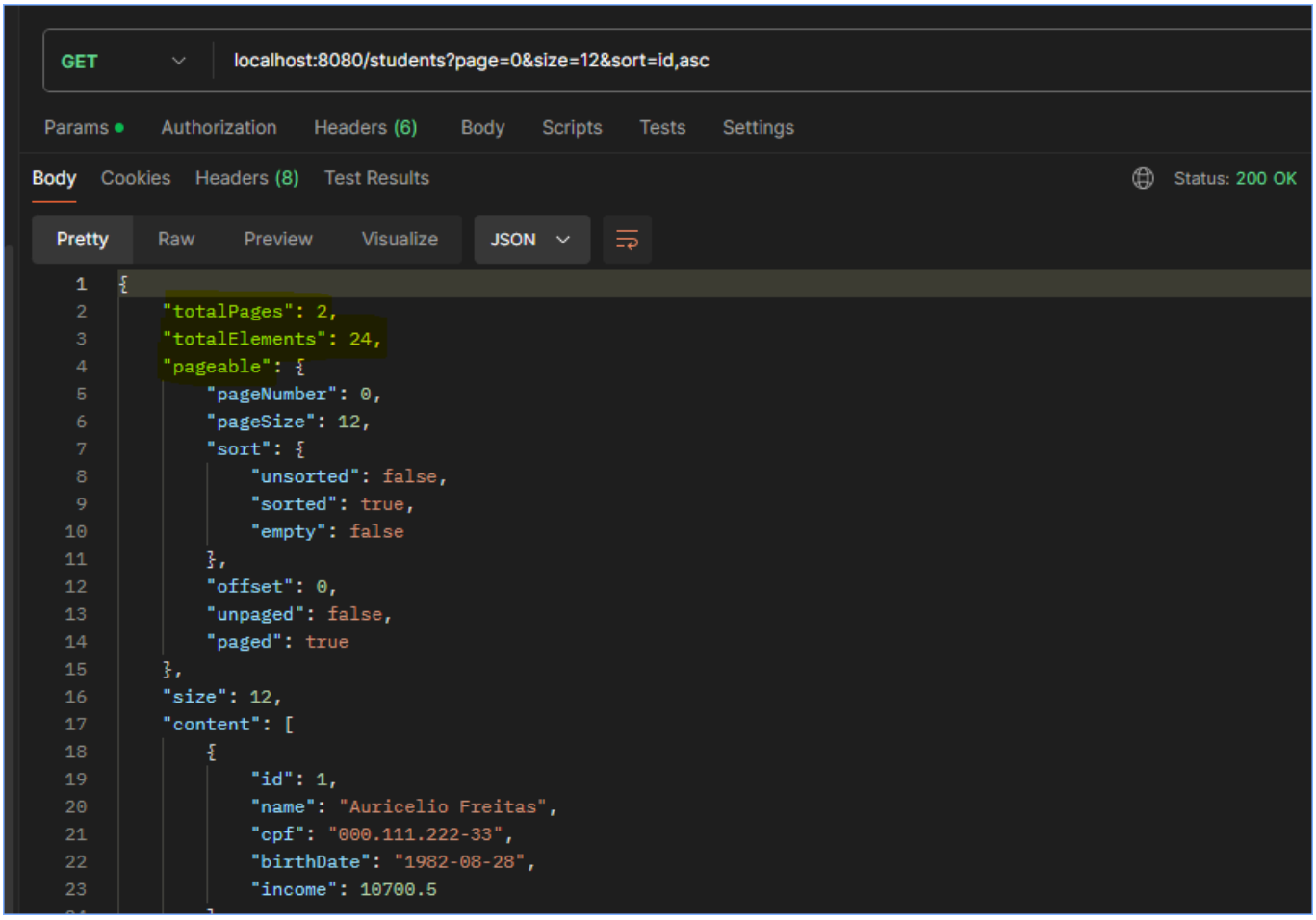
```
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Auricelio Freitas', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1982-08-28T10:30:00Z', 10700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Miguel Soares', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1989-02-10T10:30:00Z', 20700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Matheus Marques', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1992-11-23T10:30:00Z', 10700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Rafael Sales', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1988-10-13T10:30:00Z', 10700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Bruno Freitas', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1987-04-22T10:30:00Z', 9700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Romulo Araujo', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1991-01-09T10:30:00Z', 8700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Ryan Sousa', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1994-03-26T10:30:00Z', 7700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('test1', '000.111.222-33', null, 1024.0);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Auricelio Freitas', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1982-08-28T10:30:00Z', 10700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Miguel Soares', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1989-02-10T10:30:00Z', 20700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Matheus Marques', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1992-11-23T10:30:00Z', 10700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Rafael Sales', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1988-10-13T10:30:00Z', 10700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Bruno Freitas', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1987-04-22T10:30:00Z', 9700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Romulo Araujo', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1991-01-09T10:30:00Z', 8700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Ryan Sousa', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1994-03-26T10:30:00Z', 7700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('test1', '000.111.222-33', null, 1024.0);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Auricelio Freitas', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1982-08-28T10:30:00Z', 10700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Miguel Soares', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1989-02-10T10:30:00Z', 20700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Matheus Marques', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1992-11-23T10:30:00Z', 10700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Rafael Sales', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1988-10-13T10:30:00Z', 10700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Bruno Freitas', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1987-04-22T10:30:00Z', 9700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Romulo Araujo', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1991-01-09T10:30:00Z', 8700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Ryan Sousa', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1994-03-26T10:30:00Z', 7700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('test1', '000.111.222-33', null, 1024.0);
```



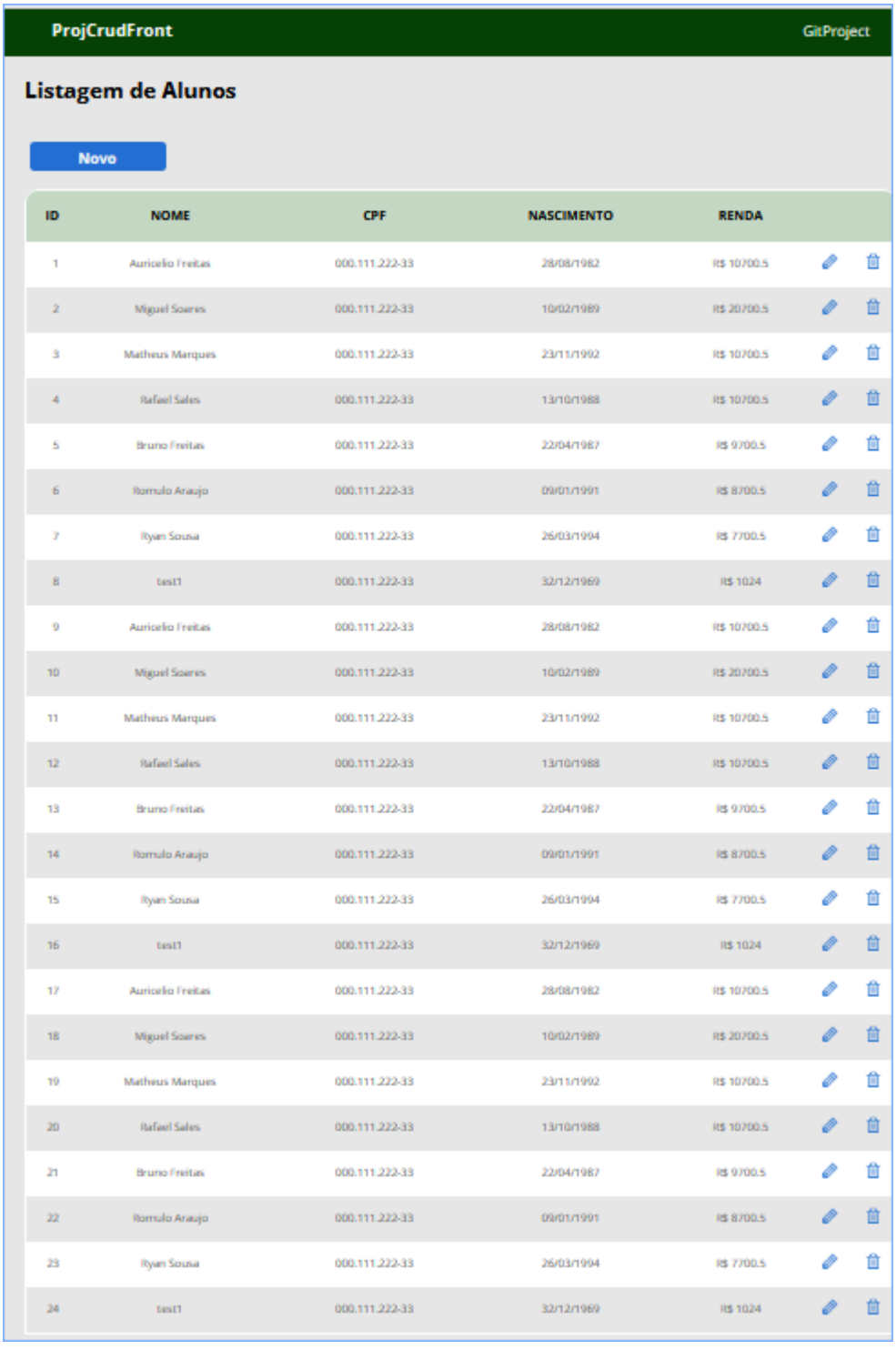
***NOTA**: Apenas replicamos os registros já existentes. No momento temos 24 registros para testarmos a paginação.*

Testar a paginação no postman

***NOTA**: Na implantação do backend (ver tutorial), já realizamos a implementação dos endpoints com paginação.*



VISUALIZAR A RENDERIZAÇÃO NO FRONTEND



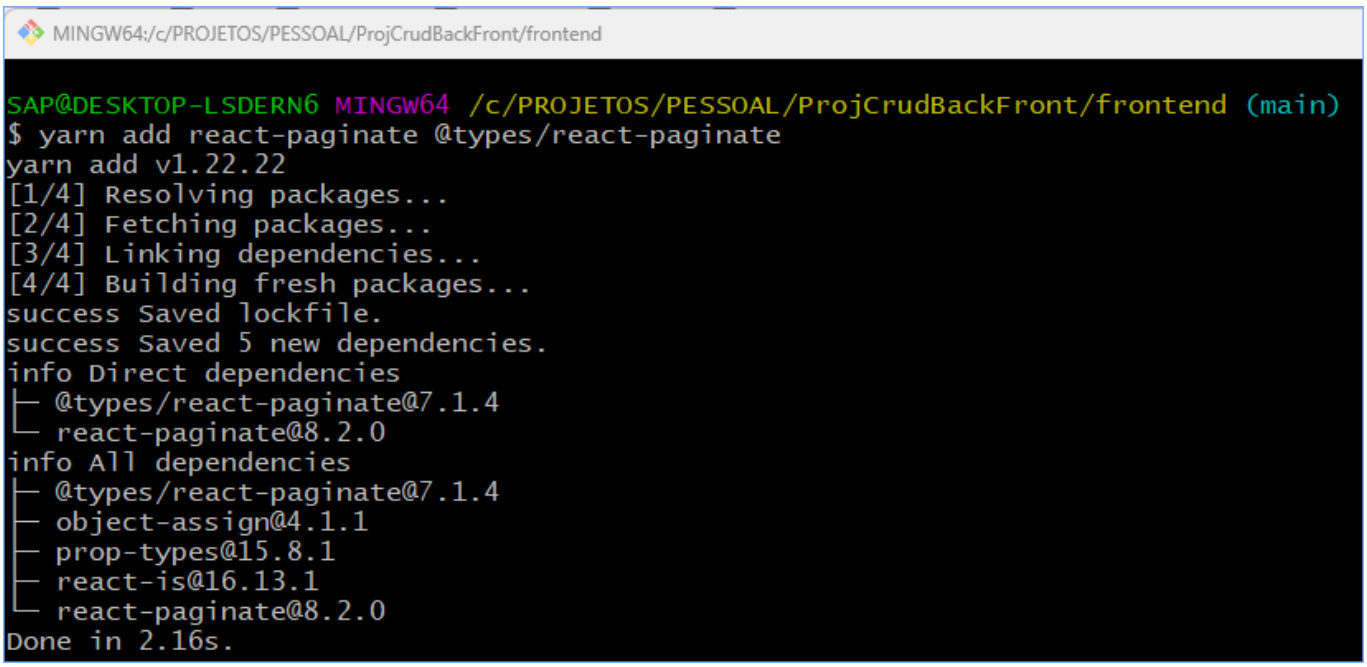
INSTALAR O PLUGIN DE PAGINAÇÃO

URL do react paginate:

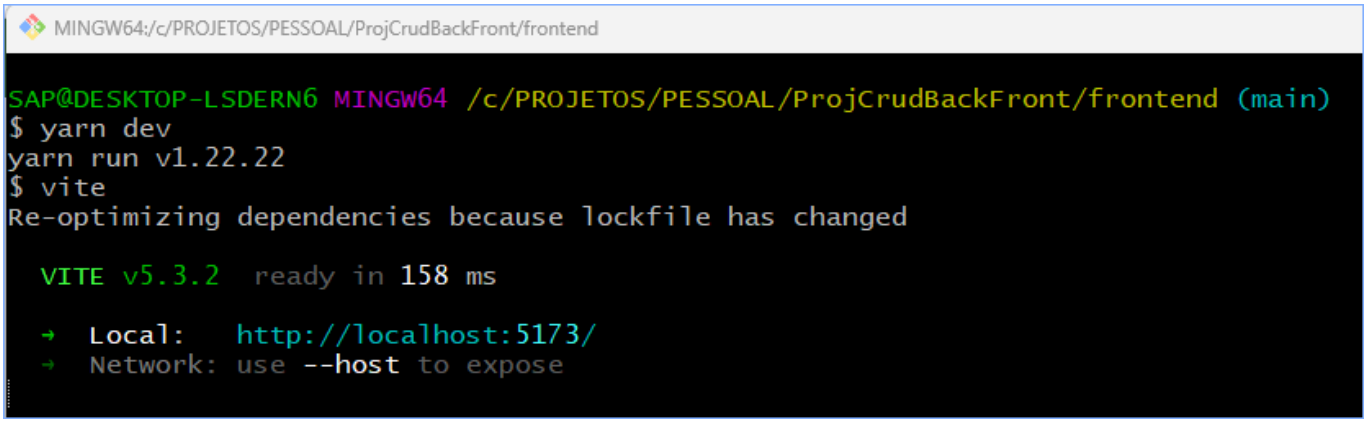
- https://github.com/AdeleD/react-paginate

Instalação

- yarn add react-paginate @types/react-paginate

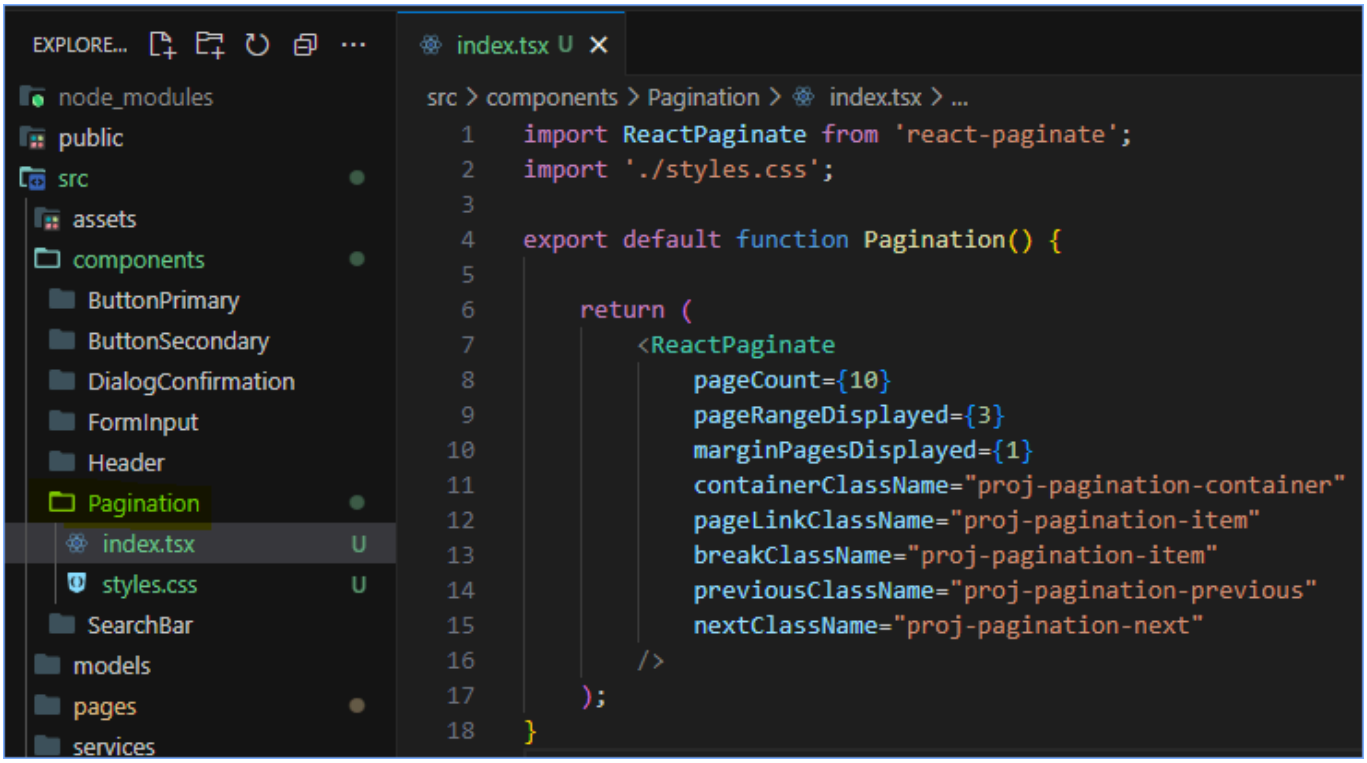


Rodar o projeto



IMPLEMENTAR O PAGINATION

Criar o componente de paginação



Implementar o pagination na página Listing

src > pages > Listing > index.tsx > ...

```
90                  <tbody>
91                      {
92                          students.map(student => (
93                              <tr key={student.id}>
94                                  <td className='proj-listing-table-id'>{student.id}</td>
95                                  <td>{student.name}</td>
96                                  <td>{student.cpf}</td>
97                                  <td>{formatDateBR(student.birthDate)}</td>
98                                  <td>R$ {student.income}</td>
99                                  <td><img src={editIcon} alt='Editar' onClick={() => {handleUpdate(student.id)}}/></td>
100                                 <td><img src={deleteIcon} alt='Deletar' onClick={() => handleDelete(student.id)}/></td>
101                             </tr>
102                         ))
103                     }
104                 </tbody>
105             </table>
106
107             <div className="proj-listing-pagination">
108                 <Pagination />
109             </div>
110
111         </section>
```

Visualização web

19	Matheus Marques	000.111.222-33	23/11/1992	R$ 10700.5	
20	Rafael Sales	000.111.222-33	13/10/1988	R$ 10700.5	
21	Bruno Freitas	000.111.222-33	22/04/1987	R$ 9700.5	
22	Romulo Araujo	000.111.222-33	09/01/1991	R$ 8700.5	
23	Ryan Sousa	000.111.222-33	26/03/1994	R$ 7700.5	
24	test1	000.111.222-33	32/12/1969	R$ 1024	

- Previous
- 1
- 2
- 3
- ...
- 10
- Next

PAGINATION LAYOUT

Estilizar o layout da paginação no Listing

src > pages > Listing > styles.css > ...

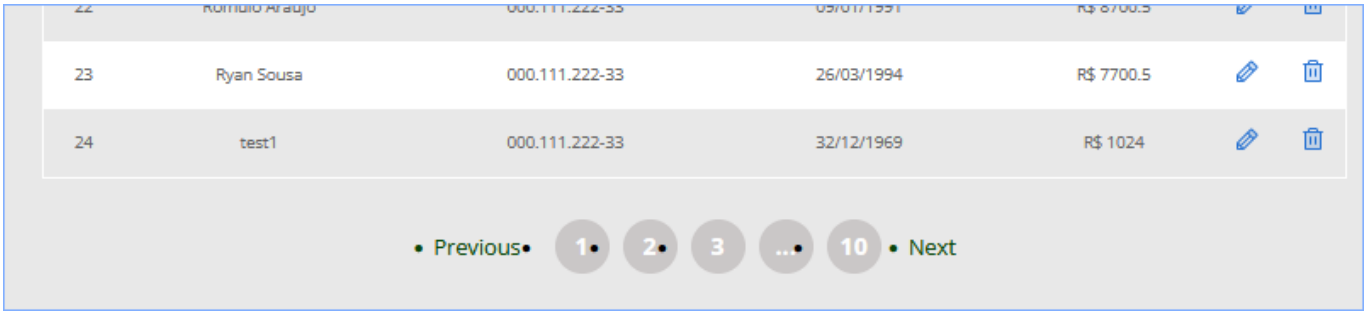
```
46
47  .proj-listing-pagination {
48      margin: 30px 30px;
49  }
50
```

Estilizar o layout da paginação no Pagination

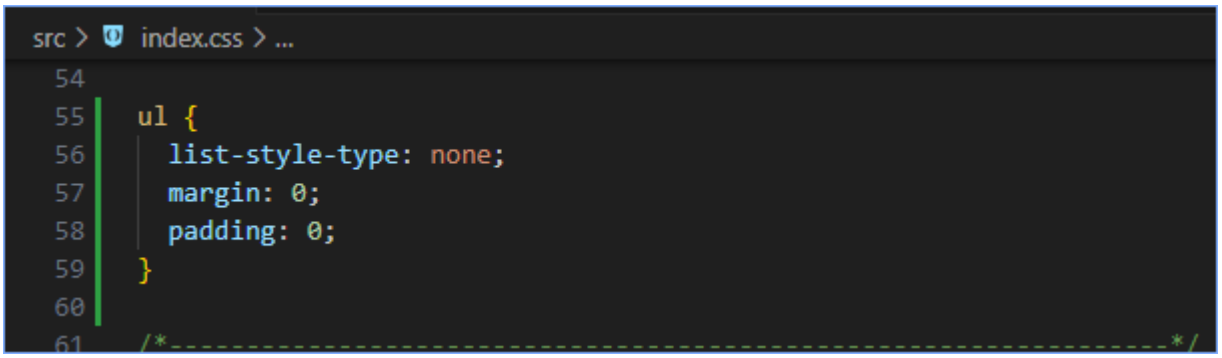
src > components > Pagination > styles.css > ...

```
1   .proj-pagination-container {
2       display: flex;
3       justify-content: center;
4       align-items: center;
5   }
6
7   .proj-pagination-item {
8       width: 40px;
9       height: 40px;
10      border-radius: 50%;
11      background-color: var(--proj-color-bg-tertiary);
12      font-size: 18px;
13      font-weight: bold;
14      color: var(--proj-color-font-tertiary);
15      display: flex;
16      justify-content: center;
17      align-items: center;
18      margin-left: 10px;
19  }
20
21  .proj-pagination-previous {
22      margin-right: 15px;
23      color: var(--proj-color-bg-secondary);
24      cursor: pointer;
25  }
26
27  .proj-pagination-next {
28      margin-left: 20px;
29      color: var(--proj-color-bg-secondary);
30      cursor: pointer;
31  }
```

Visualização web

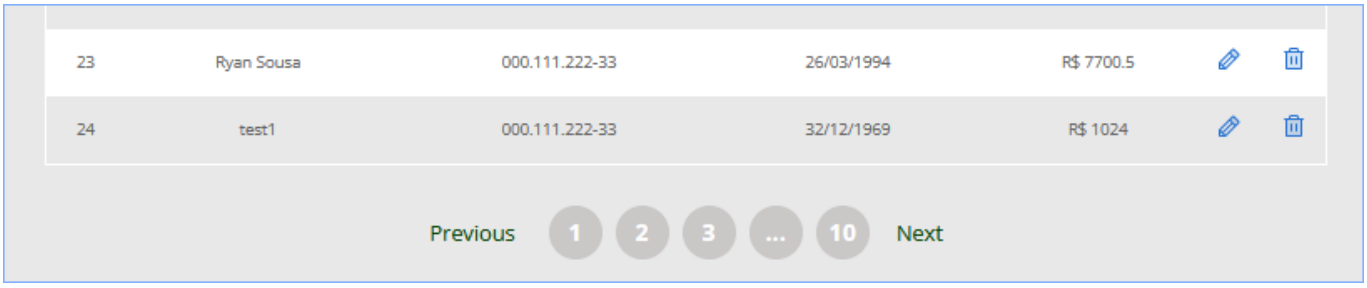


Retirar os marcadores no index.css



NOTA: *Aqui optamos por realizar uma implementação de forma global.*

Visualização web



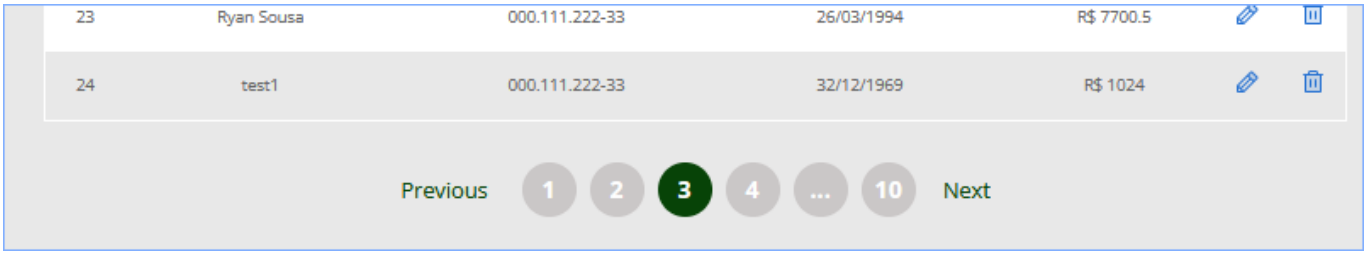
Aplicar estilo de cor ao número da página atual

```
src > components > Pagination > index.tsx > ...
                pageLinkClassName="proj-pagination-item"
                breakClassName="proj-pagination-item"
                previousClassName="proj-pagination-previous"
                nextClassName="proj-pagination-next"
                activeLinkClassName="proj-pagination-link-active"
            />
        );
    }
```

Estilizar

```
src > components > Pagination > styles.css > ...

.proj-pagination-link-active {
    background-color: var(--proj-color-bg-secondary);
}
```

Visualização web

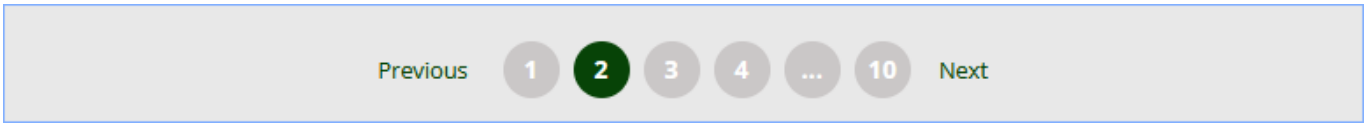


Inativar o “previous” e o “next”, quando do início e fim da paginação

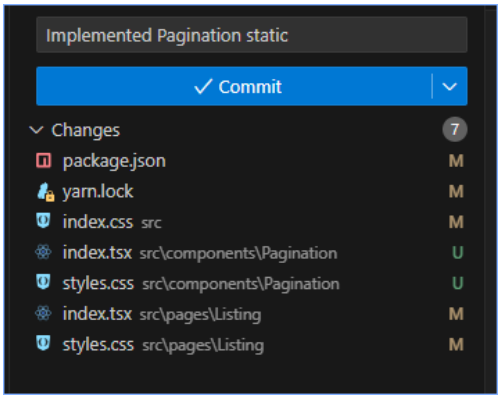
src > components > Pagination > index.tsx > ...

```tsx
export default function Pagination() {

    return (
        <ReactPaginate
            pageCount={10}
            pageRangeDisplayed={3}
            marginPagesDisplayed={1}
            containerClassName="proj-pagination-container"
            pageLinkClassName="proj-pagination-item"
            breakClassName="proj-pagination-item"
            previousClassName="proj-pagination-previous"
            nextClassName="proj-pagination-next"
            activeLinkClassName="proj-pagination-link-active"
            disableInitialCallback
            disabledClassName="proj-pagination-inactive"
        />
    );
}
```

Visualização web



GitHub-24



RENDERIZAR A PAGINAÇÃO DINAMICAMENTE

Renderizar a quantidade de páginas no componente Pagination

Criando o props no componente pagination

src > components > Pagination > index.tsx > ...

```tsx
import ReactPaginate from 'react-paginate';
import './styles.css';

type Props = {
    pageCount: number;
    range: number;
}

export default function Pagination( {pageCount, range} : Props ) {

    return (
        <ReactPaginate
            pageCount={pageCount}
            pageRangeDisplayed={range}
            marginPagesDisplayed={1}
            containerClassName="proj-pagination-container"
            pageLinkClassName="proj-pagination-item"
            breakClassName="proj-pagination-item"
            previousClassName="proj-pagination-previous"
            nextClassName="proj-pagination-next"
            activeLinkClassName="proj-pagination-link-active"
            disableInitialCallback
            disabledClassName="proj-pagination-inactive"
        />
    );
}
```

Refatorar o student-service para dividir as páginas por quantidade de registro

src > services > student-service.ts > ...

```
5
6   export function findPageRequest(page: number, name: string, size = 10, sort = "id") {
7       const config : AxiosRequestConfig = {
8           method: "GET",
9           baseURL: BASE_URL,
10          url: "/students",
11          params: {
12              page,
13              name,
14              size,
15              sort
16          }
17      }
18
19      return requestBackend(config)
20  }
```

NOTA: *Como temos 24 registros implementados no seed do backend, então a renderização será de 3 páginas.*

Implementar um estado para pegar a quantidade de páginas

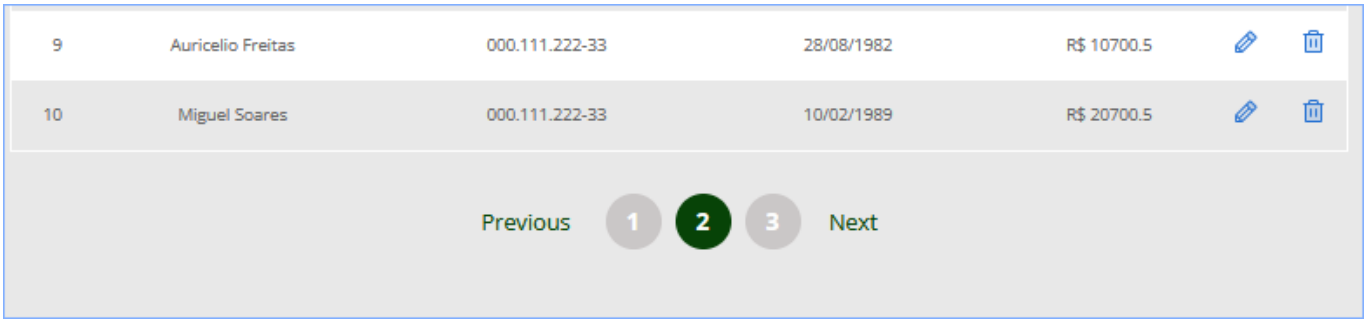
src > pages > Listing > index.tsx > ...

```
35      const [pageCounts, setPageCounts] = useState();
36
37      useEffect(() => {
38          studentService.findPageRequest(queryParams.page, queryParams.name)
39              .then(response => {
40                  setStudents(response.data.content)
41                  setPageCounts(response.data.totalPages)
42              })
43      }, [])
44
```

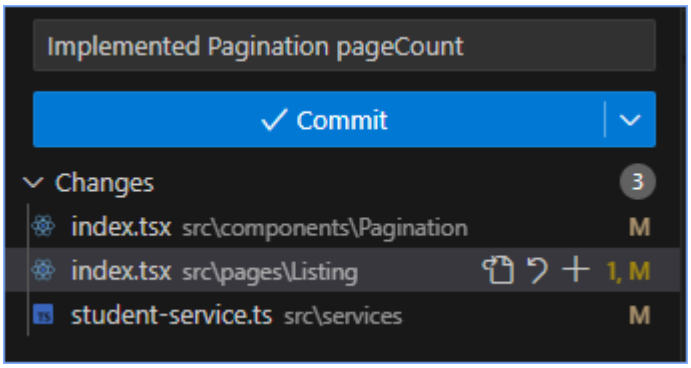
Renderizar pageCounts no Pagination, do Listing

```
src > pages > Listing > index.tsx > ...
109
110                 <div className="proj-listing-pagination">
111                     <Pagination
112                         pageCount={Number((pageCounts) ? pageCounts : 0)}
113                         range={3}
114                     />
115                 </div>
```

Visualização web



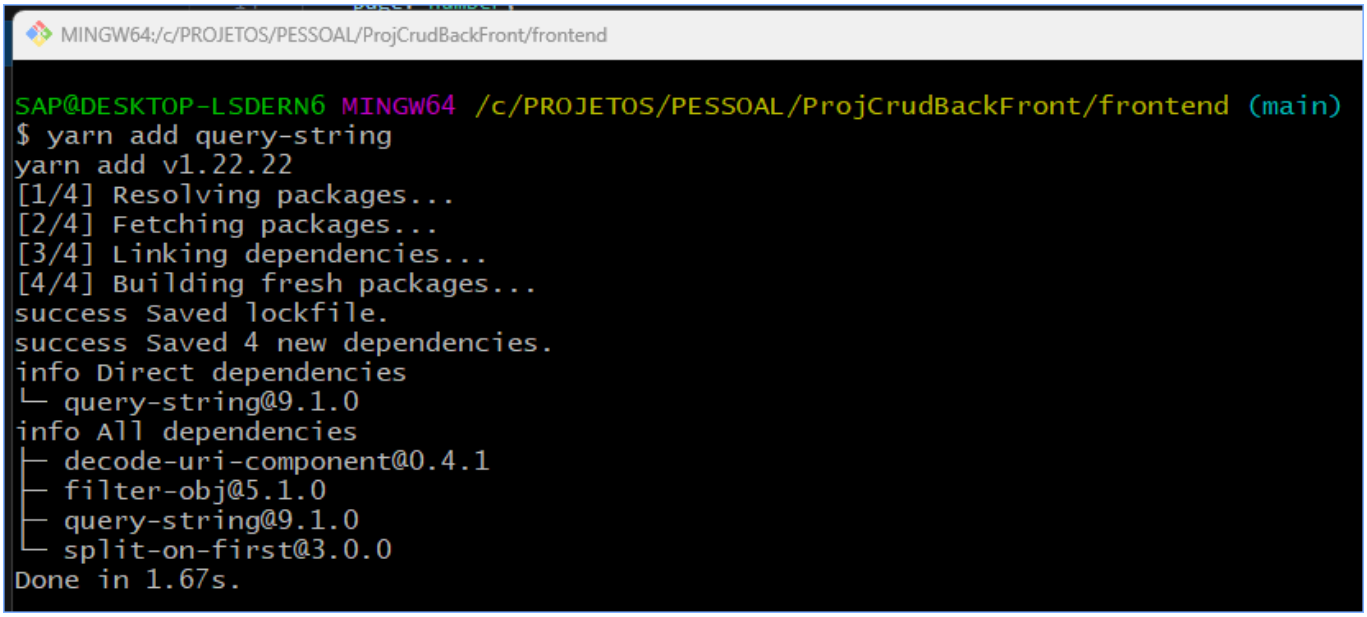
GitHub-25



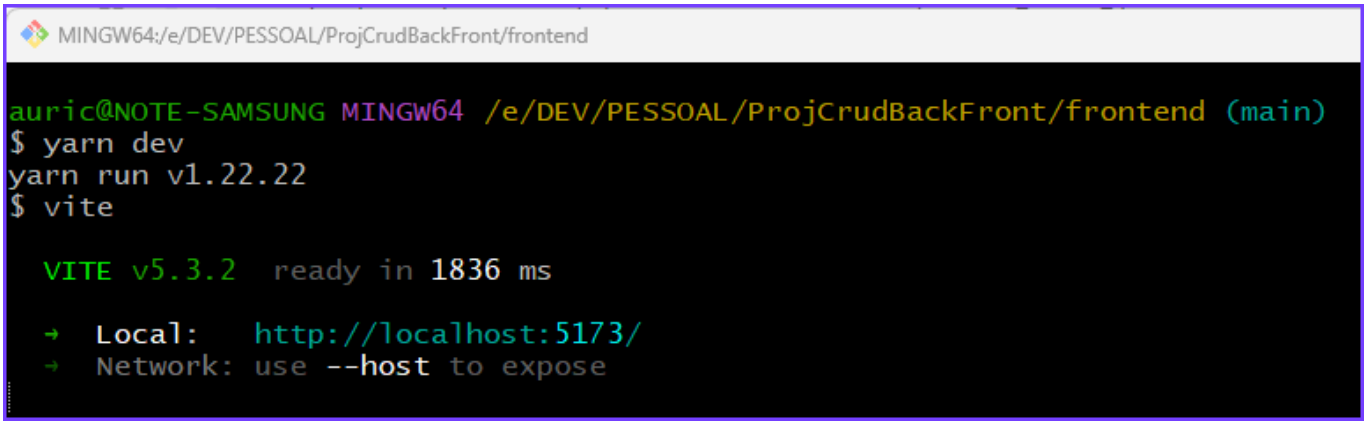
Renderizar a paginação na URL

Instalar o pacote query-string

- yarn add query-string



***Nota**: startar o projeto em seguida*



Implementar os Hooks useNavigate e useLocation

```
src > pages > Listing > index.tsx > ...
20  export default function Listing() {
21
22      const navigate = useNavigate()
23
24      const location = useLocation()
```

Implementar uma função e um estado para a página atual

```
src > pages > Listing > index.tsx > ...
41      const [actualPage, setActualPage] = useState(getActualPage());
42
43      function getActualPage() {
44          const params = qs.parse(location.search);
45          const page = params.page;
46          return page ? Number(page) : 0;
47      }
```

Implementar o useEffect para monitorar a mudança de página

src > pages > Listing > index.tsx > ...

```
52      useEffect(() => {
53          studentService.findPageRequest(queryParams.page, queryParams.name)
54              .then(response => {
55                  setStudents(response.data.content)
56                  setPageCounts(response.data.totalPages)
57                  setActualPage(actualPage)
58              })
59
60              const params = qs.parse(location.search)
61
62              navigate({
63                  search: qs.stringify({
64                      ...params,
65                      page: actualPage
66                  })
67              })
68
69      }, [actualPage]);
```

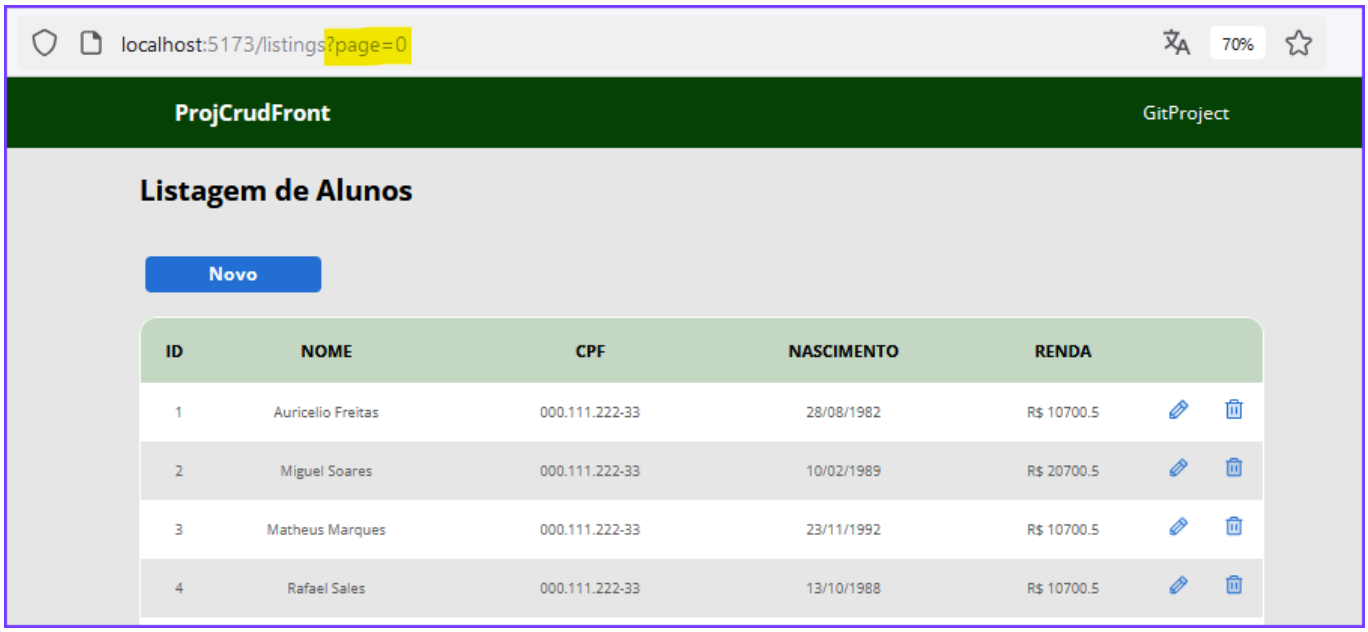
Chamar o getActualPage no queryParams

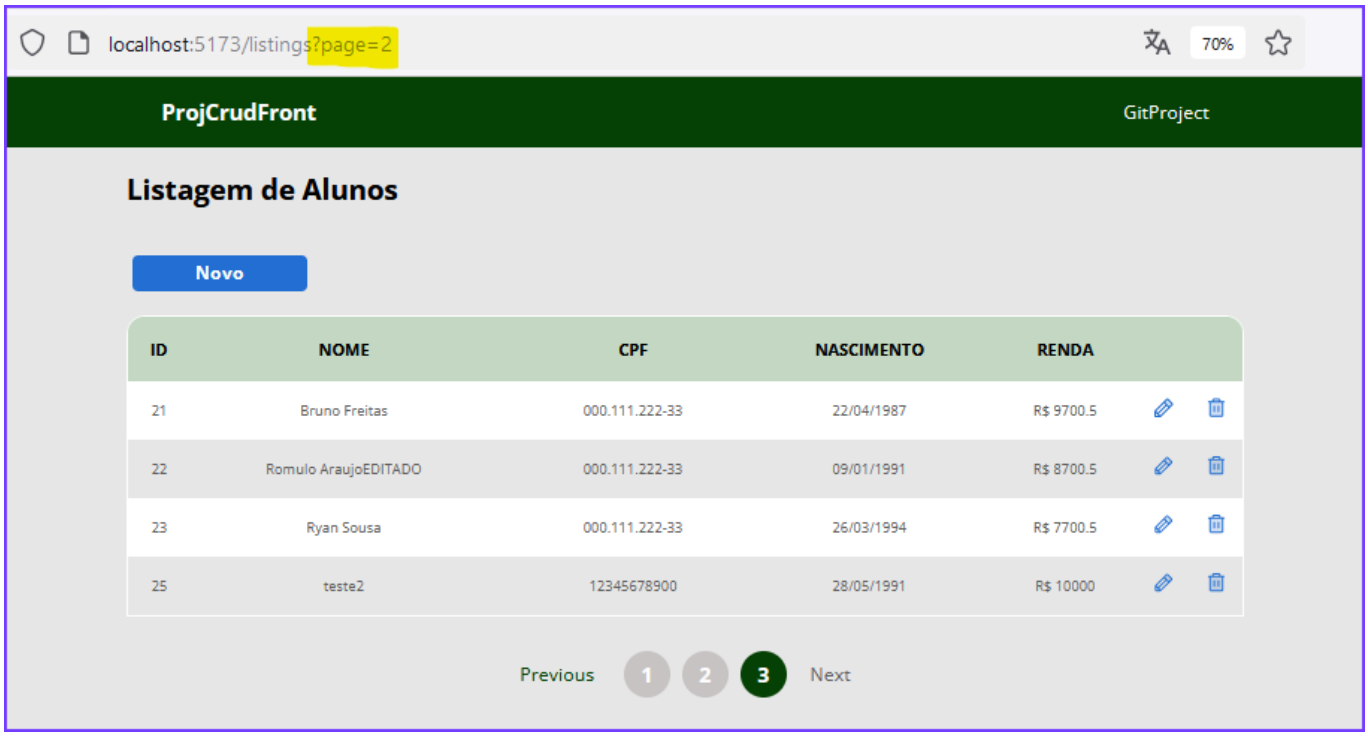
src > pages > Listing > index.tsx > ...

```
31
32      const [queryParams] = useState<QueryParams>({
33          page: getActualPage() || 0,
34          name: "",
35      });
36
```

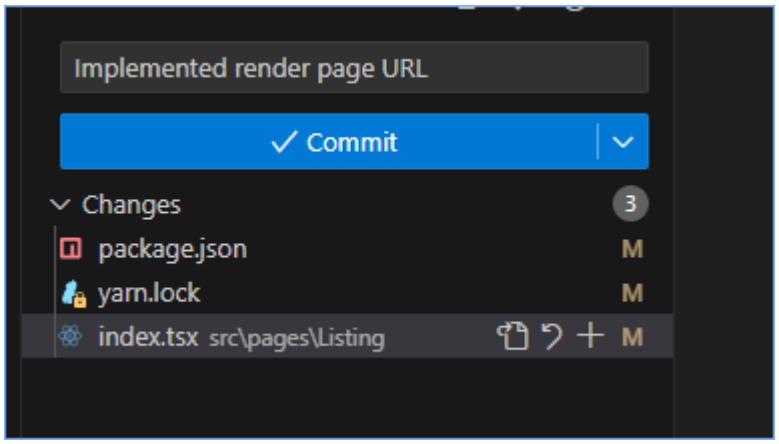
***NOTA**: Caso ele não pegue a página atual, irá para a página "0", que é a primeira página do Listing*

Visualização web



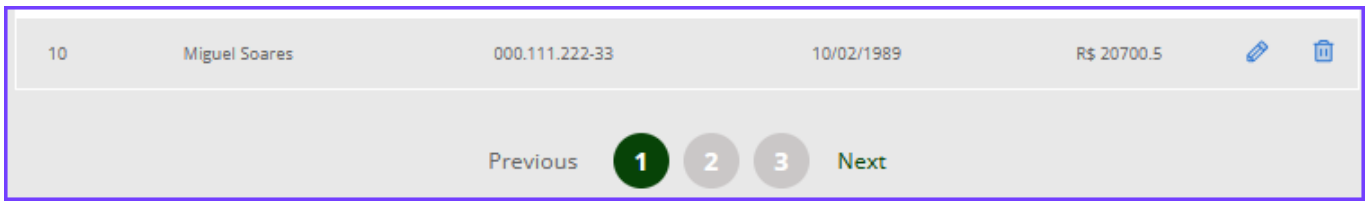

localhost:5173/listings?page=2
70%
ProjCrudFront
GitProject
Listagem de Alunos
Novo
ID
NOME
CPF
NASCIMENTO
RENDA
21
Bruno Freitas
000.111.222-33
22/04/1987
R$ 9700.5
22
Romulo AraujoEDITADO
000.111.222-33
09/01/1991
R$ 8700.5
23
Ryan Sousa
000.111.222-33
26/03/1994
R$ 7700.5
25
teste2
12345678900
28/05/1991
R$ 10000
Previous
1
2
3
Next

GitHub-26


Implemented render page URL
Commit
Changes
3
package.json
M
yarn.lock
M
index.tsx src\pages\Listing
M

Renderizar a paginação no componente pagination

OBJETIVO: Ao clicar no componente, a página mudará conforme seleção.



Criar um evento, no pagination, que irá disparar quando a página mudar

src > components > Pagination > index.tsx > ...

```tsx
import ReactPaginate from 'react-paginate';
import './styles.css';

type Props = {
    pageCount: number;
    range: number;
    onChange: (pageNumber: number) => void;
}

export default function Pagination({ pageCount, range, onChange }: Props) {
    return (
        <ReactPaginate
            pageCount={pageCount}
            pageRangeDisplayed={range}
            marginPagesDisplayed={1}
            containerClassName="proj-pagination-container"
            pageLinkClassName="proj-pagination-item"
            breakClassName="proj-pagination-item"
            previousClassName="proj-pagination-previous"
            nextClassName="proj-pagination-next"
            activeLinkClassName="proj-pagination-link-active"
            disableInitialCallback
            disabledClassName="proj-pagination-inactive"
            onPageChange={(items) => onChange(items.selected)}
        />
    );
}
```

***NOTA**: O onChange espera um número de página como argumento, que será passado para o atributo onPageChange.*

Implementar o handlePageChange e refatorar o useEffect, na página Listing

src > pages > Listing > index.tsx > ...

```
48
49      const handlePageChange = (pageNumber: number) => {
50          setActualPage(pageNumber);
51          const params = qs.parse(location.search);
52          navigate({
53              search: qs.stringify({ ...params, page: pageNumber })
54          });
55      };
56
57      useEffect(() => {
58          studentService.findPageRequest(actualPage, queryParams.name)
59              .then(response => {
60                  setStudents(response.data.content);
61                  setPageCounts(response.data.totalPages);
62              });
63
64      }, [actualPage, queryParams.name]);
```

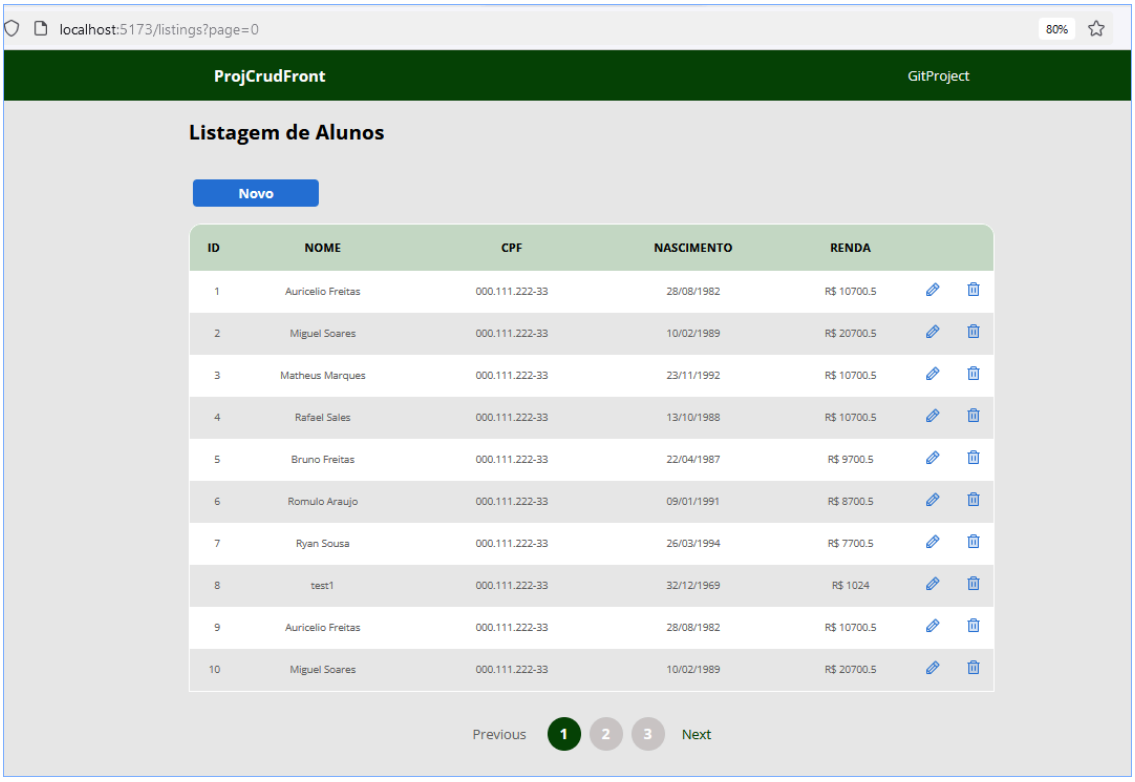
NOTA: *No handlePageChange, quando o usuário selecionar uma nova página, precisamos atualizar o estado actualPage e também navegar para a nova URL com a página selecionada usando o "navigate". Enxugar o useEffect retirando as redundâncias.*

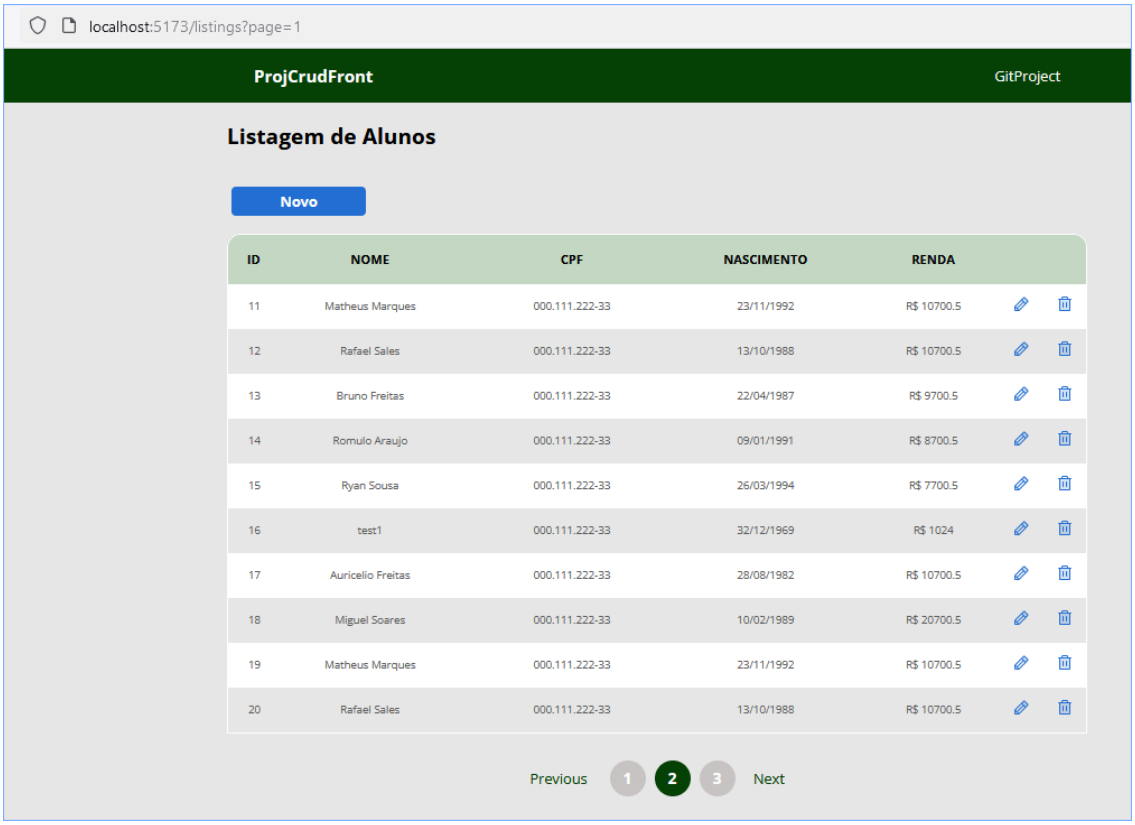
Chamar o handlePageChange no Pagination do Listing

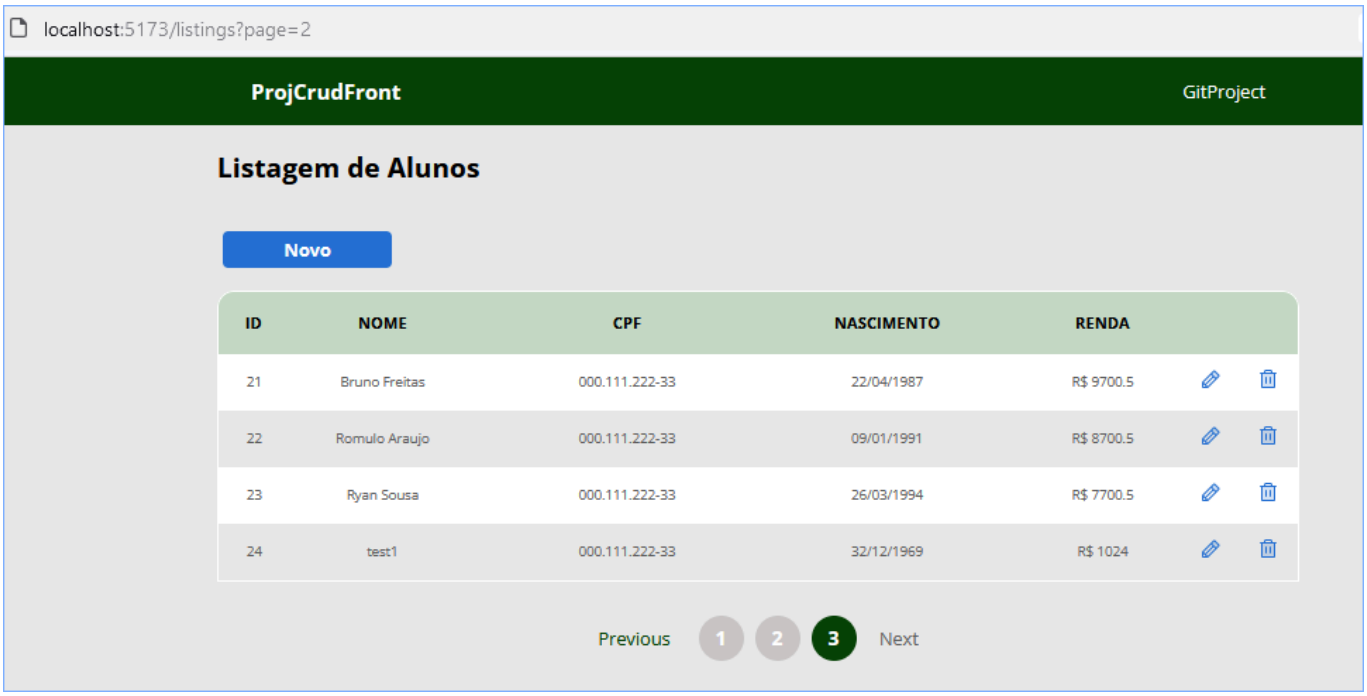
src > pages > Listing > index.tsx > ...

```
140
141                 <div className="proj-listing-pagination">
142                     <Pagination
143                         pageCount={Number((pageCounts) ? pageCounts : 0)}
144                         range={3}
145                         onChange={handlePageChange}
146                     />
```

Visualização web

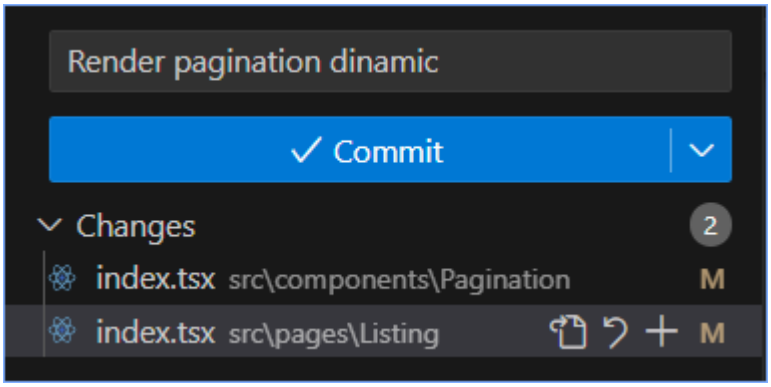






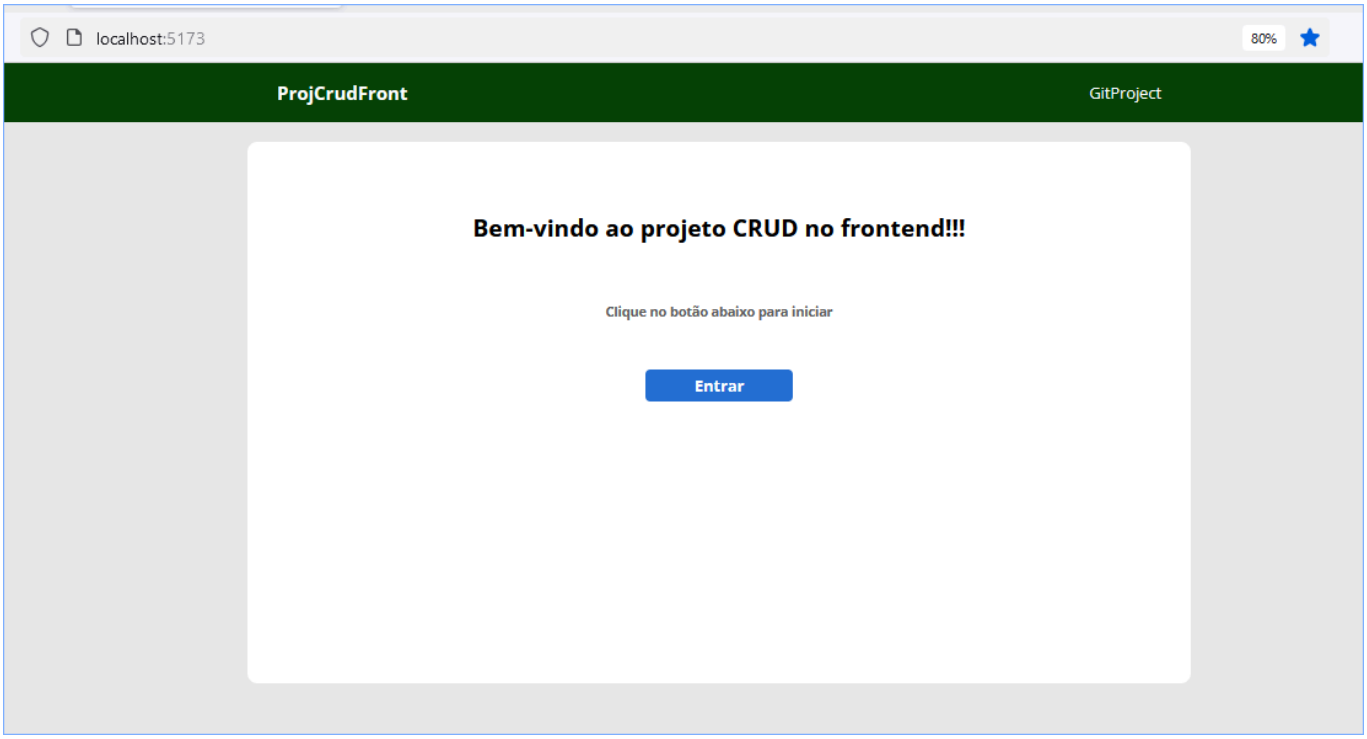
***NOTA**: Neste momento o nosso componente já está passando as páginas ao clicar.*

GitHub-27

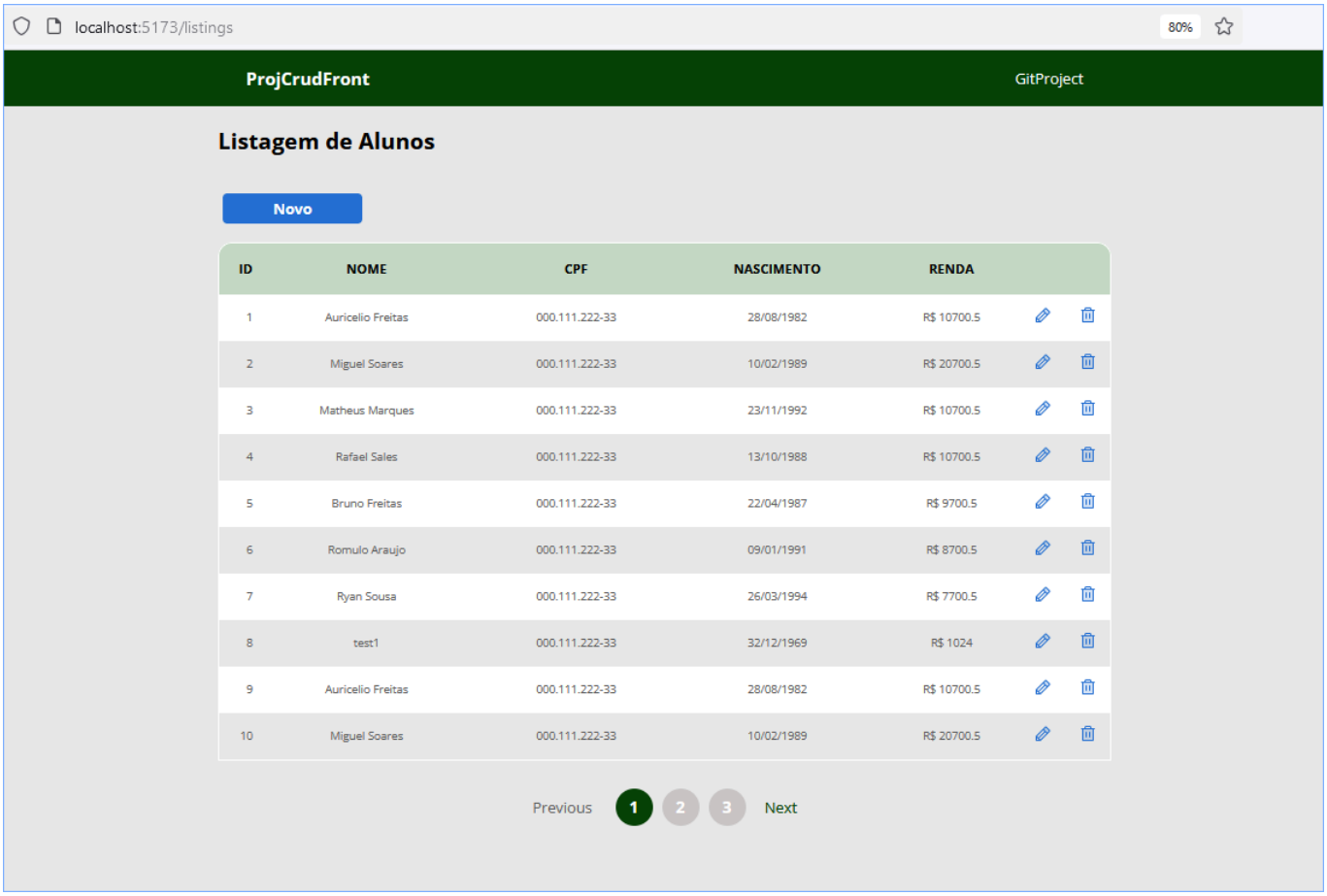


SISTEMA FINALIZADO

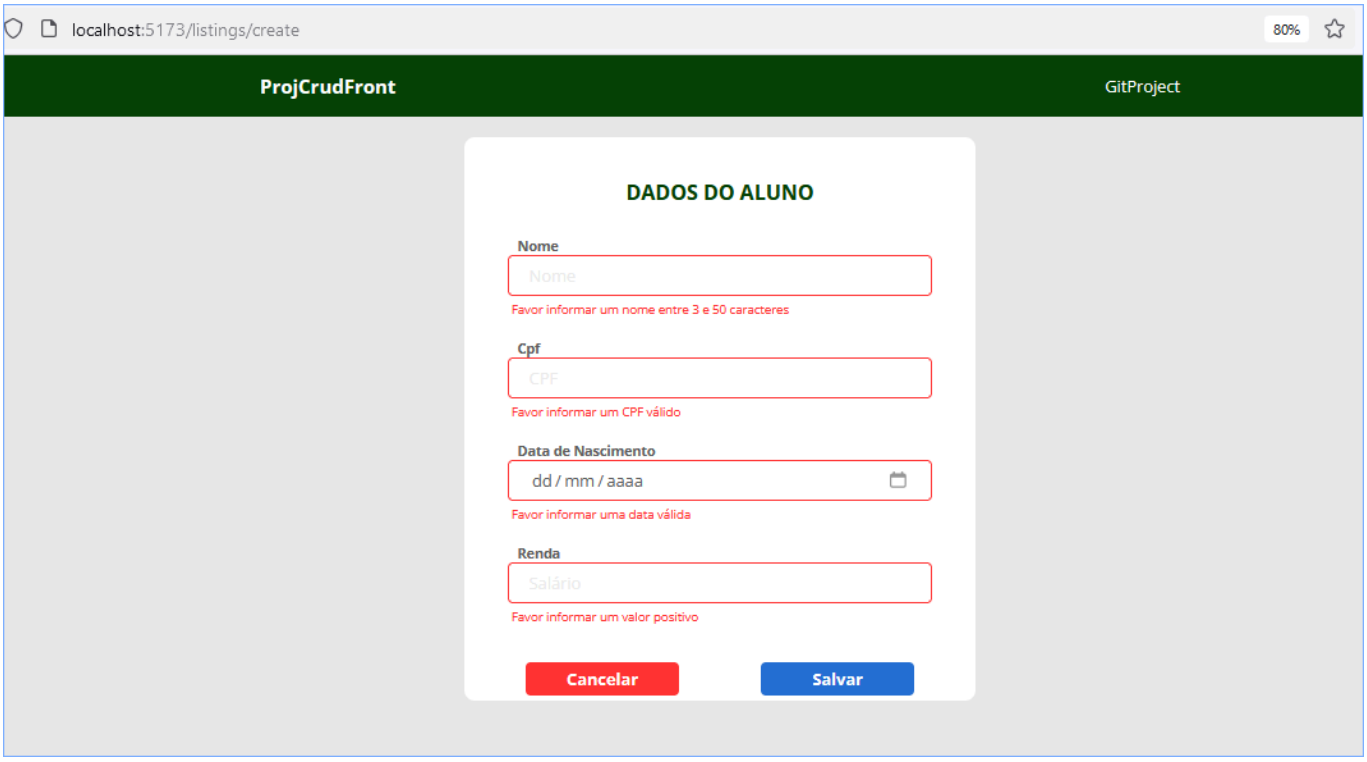
Home



Listagem de Alunos

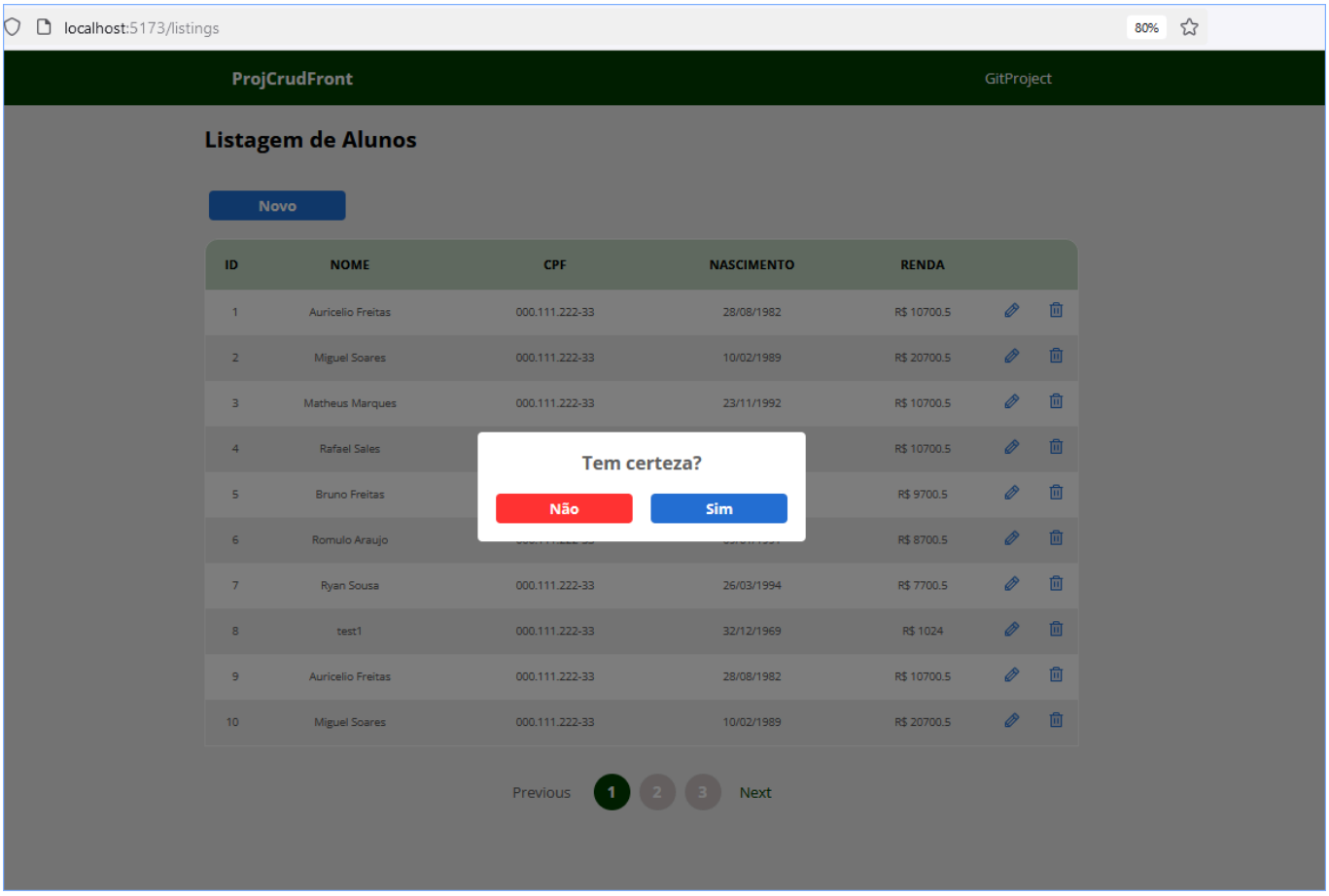


Cadastro de alunos



***NOTA**: Formulário implementado com validação em todos os campos.*

Deletar Aluno



NOTA: *O processo de exclusão de alunos foi implementado com um Modal de confirmação, para evitar exclusões acidentais.*

FIM