

PROJETO CRUD - FRONTEND

SUMÁRIO

PROJETO CRUD - FRONTEND.....	1
OBJETIVO.....	10
DEPENDÊNCIAS.....	10
CASO DE USO.....	10
TESTES A SEREM REALIZADOS NO POSTMAN.....	11
INSTALAÇÃO DO NODEJS.....	11
BAIXAR.....	11
EXECUTAR.....	12
TESTAR NO TERMINAL.....	15
IMPLEMENTAR O YARN.....	16
INSTALAR.....	16
TESTAR.....	16
IMPLEMENTAR O GIT.....	17
BAIXAR.....	17
INSTALAR.....	17
TESTAR.....	18
IMPLEMENTAR O FRONTEND.....	19
ABRIR O GIT NO WORKSPACE DO PROJETO.....	19
CRIAR O PROJETO.....	19
ABRIR COM O VSCODE.....	20
DEPENDÊNCIAS.....	21
ATUALIZAR AS DEPENDÊNCIAS DO YARN.....	21
RODAR O PROJETO.....	21
VISUALIZAÇÃO WEB.....	22
GitHub-1.....	23
LIMPAR O PROJETO.....	24
INDEX.HTML.....	24
Antes.....	24
Depois.....	24
APP.TSX.....	25
Antes.....	25
Depois.....	25

INDEX.CSS.....	26
Antes.....	26
Depois.....	26
APP.CSS.....	27
Antes.....	27
Depois.....	27
ASSETS.....	28
Antes.....	28
Depois.....	28
RODAR O PROJETO.....	28
VISUALIZAÇÃO WEB.....	29
GitHub-2.....	29
CSS.....	30
FONTE.....	30
COR.....	30
GERAL.....	32
ESTILOS GENÉRICOS.....	33
VISUALIZAÇÃO WEB.....	35
GitHub-3.....	35
IMPLEMENTAR A ESTRUTURA DO PROJETO.....	36
CRIAR A ESTRUTURA BASE DAS PAGINAS DEFAULT.....	37
Home.....	37
Renderizar a página no App.tsx.....	37
Visualização web.....	38
Listing.....	38
Renderizar a página no App.tsx.....	39
Visualização web.....	39
NewForm.....	40
Renderizar a página no App.tsx.....	40
Visualização web.....	40
GitHub-4.....	41
ROTAS.....	42
INSTALAR O REACT-ROUTER-DOM.....	42
VERIFICAR NO PROJETO.....	43
IMPLEMENTAR AS ROTAS NO APP.TSX.....	44
VERIFICAR O MAIN.TSX.....	45
VISUALIZAÇÃO WEB.....	45
Home.....	45
Listing.....	45
NewForm.....	46

GitHub-5.....	46
CRIAR O COMPONENTE HEADER.....	47
IMPLEMENTAR O INDEX.....	47
ESTILIZAR.....	48
RENDERIZAR O HEADER NO APP.TSX.....	49
VISUALIZAÇÃO WEB.....	49
IMPLEMENTAR OS LINKS DO HEADER.....	50
GitHub-6.....	50
CRIAR O COMPONENTE BUTTON_PRIMARY.....	51
IMPLEMENTAR O INDEX.....	51
ESTILIZAR.....	51
CRIAR O COMPONENTE BUTTON_SECONDARY.....	52
IMPLEMENTAR O INDEX.....	52
ESTILIZAR.....	52
GitHub-7.....	53
PÁGINA HOME.....	54
IMPLEMENTAR O INDEX.....	54
ESTILIZAR.....	54
VISUALIZAÇÃO WEB.....	55
CHAMADO O COMPONENTE BUTTON_PRIMARY.....	56
ESTILIZAR.....	56
VISUALIZAÇÃO WEB.....	57
GitHub-8.....	57
PÁGINA LISTING.....	58
BAIXAR ICONES.....	58
Lápis.....	58
Lixeira.....	59
GitHub-9.....	60
LAYOUT DA LISTAGEM DE ALUNOS.....	61
Implementar o Index.....	61
Estilizar.....	63
Visualização web.....	64
GitHub-10.....	64
LISTAGEM DINÂMICA.....	65
INSTALAR O AXIOS.....	65
Verificar versões.....	65
Adicionar ao projeto.....	66
Verificar instalação.....	66
IMPLEMENTAR O UTILS.....	67
Criar o system.....	67

Criar o request.....	67
Criar o format.....	68
IMPLEMENTAR O SERVICE.....	68
Criar o student-service.....	68
IMPLEMENTAR O MODELS.....	69
Criar o students.ts.....	69
REALIZAR AS CHAMADAS NO INDEX DO LISTING.....	69
Implementar o tipo QueryParams para pegar a pagina atual.....	69
Implementar o useState, tipado com o QueryParams.....	70
Implementar o useState, tipado com o StudentDTO.....	70
Implementar o useEffect, chamado o Axios.....	70
Implementar a renderização da tabela dinamicamente.....	71
VISUALIZAÇÃO WEB.....	71
Iniciar o Backend.....	71
Iniciar o Frontend.....	72
Visualizar.....	72
GitHub-11.....	73
PAGINA NEW_FORM.....	74
PROJETO ATÉ AQUI.....	74
IMPLEMENTAR O LAYOUT DA PÁGINA NEW_FORM.....	75
Index.....	75
Styles.....	76
Visualização web.....	78
GitHub-12.....	78
ESTRATÉGIA PARA IMPLEMENTAÇÃO DE FORMULÁRIO.....	79
ESTRATÉGIA.....	79
IMPLEMENTAR O COMPONENTE FORM_INPUT.....	80
IMPLEMENTAR O UTILITÁRIO FORMS.....	80
• update.....	80
IMPLEMENTAR A PAGE NEW_FORM.....	80
Implementar os Imports.....	80
Implementar o formData com useState.....	81
Implementar o FormInput.....	82
Implementar a função handleInputChange.....	82
Visualização web.....	83
GitHub-13.....	83
DIFERENCIAR CRIAÇÃO E EDIÇÃO NO FORMULÁRIO.....	84
IMPLEMENTAR A ESTRATÉGIA PARA O FORMULÁRIO DE NOVO ALUNO.....	84
Rever a rota do studentId.....	85
Implementar o botão Novo, na página Listing.....	85

Implementar o Navigate.....	85
Implementar a função handleNewProduct.....	86
Visualização web.....	86
IMPLEMENTAR A ESTRATÉGIA PARA O FORMULÁRIO DE EDIÇÃO ALUNOS.....	87
Implementar o findByld no student-service para requisição no backend.....	87
Implementar useParams no NewForm.....	88
Implementar o useEffect, para pegar os produtos no banco.....	88
Visualização web.....	89
GitHub-14.....	89
ADICIONAR OS VALORES DO BANCO NO FORMULÁRIO DE EDIÇÃO.....	90
IMPLEMENTAR A FUNÇÃO UPDATE_ALL, NO UTILITÁRIO FORMS.....	90
CHAMAR O UPDATE_ALL NO USE_EFFECT DO NEW_FORM.....	90
VISUALIZAÇÃO WEB.....	91
IMPLEMENTAR O SET_FORM_DATA NO USE_EFFECT DO NEW_FORM.....	91
VISUALIZAÇÃO WEB.....	92
GitHub-15.....	93
ADICIONAR O NAVIGATE AO BOTÃO EDITAR.....	94
IMPLEMENTAR O ONCLICK DO BOTÃO EDITAR, NO LISTING.....	94
IMPLEMENTAR A FUNÇÃO PARA O HANDLEUPDATE.....	95
VISUALIZAÇÃO WEB.....	95
GitHub-16.....	96
SUBMETER VALORES AO BANCO (CRUD).....	97
EDITAR COM O PUT.....	97
Testar resposta no postman.....	97
Visualização web.....	98
Implementar o toValues, no utilitario “forms”, para renderizar o objeto.....	98
Implementar o handleSubmit, no NewForm.....	99
Implementar um teste.....	99
Visualização web.....	100
Implementar uma requisição PUT, no student-service.....	100
Implementar o navigate, no NewForm.....	101
Implementar o updateRequest, no NewForm.....	101
Visualização web.....	102
GitHub-17.....	103
INSERIR COM O POST.....	103
Testar resposta no postman.....	103
Visualização web.....	104
Implementar uma requisição POST, no student-service.....	104
Implementar o insertRequest, no NewForm.....	105
Visualização web.....	105

GitHub-18.....	107
IMPLEMENTAR O DELETE COM CONFIRMAÇÃO.....	108
Testar resposta no postman.....	108
Visualização web.....	108
Implementar o componente Modal.....	109
Estilizar o Modal.....	110
Implementar a chamado do Dialog no Listing.....	111
Implementar o “onClick” do botão Delete.....	111
Implementar a função handleDelete.....	111
Implementar o deleteRequest, no product-service.....	112
Implementar a função handleDialogConfirmationAnswer, no Listing.....	112
Visualização web.....	113
GitHub-19.....	114
CONTROLE DE FORMULÁRIO.....	115
O QUE SERÁ ABORDADO?.....	115
IMPLEMENTAR A FUNÇÃO VALIDATE.....	115
Criar a função validate, no forms do utils.....	115
Implementar um teste com valor “não positivo”, no campo “income”, do NewForm.....	116
Implementar uma chamada para o teste.....	116
Visualização web.....	117
Realizar teste com valor “positivo”, no ProductForm.....	117
Visualização web.....	118
Implementar condicional para função validate, no forms.....	118
Visualização web.....	119
Testar com um campo sem o validation.....	119
Visualização web.....	120
Remover os testes anteriores.....	120
GitHub-20.....	121
IMPLEMENTAR A VALIDAÇÃO.....	122
Verificar implementação do Componente FormInput.....	122
Refatorar a função handleInputChange, no NewForm.....	122
Validar os dados atualizados.....	122
Visualização web.....	123
Implementar o estilo CSS da borda.....	124
Visualização web.....	124
Implementar o validation em todos os campos.....	125
Implementar a mensagem de erro, no FormInpu.....	126
Implementar o estilo.....	127
Visualização web.....	127
GitHub-21.....	128

IMPLEMENTAR O COMPORTAMENTO SUJO (DIRTY).....	129
Simulação.....	129
Implementar o onTurnDirty, no componente FormInput.....	130
Implementar a função toDirty, no utilitário forms.....	130
Implementar um teste na página NewForm.....	131
Visualização web.....	131
Implementar o evento onBlur, no componente FormInput.....	132
Implementar o onTurnDirty no NewForm.....	133
Implementar a função para o handleTurnDirty.....	134
Visualização web.....	134
Simulação.....	135
Implementar o CSS para o data-dirty.....	135
Simulação.....	136
Default.....	136
Digitar um valor inválido no campo.....	136
Digitar um valor inválido e sair do campo.....	137
Melhorias no código.....	137
Remover testes, no NewForm.....	137
Criar uma função auxiliar para o Update e Validate, no forms.....	138
Criar uma função auxiliar para o Dirty e Validate, no forms.....	138
Refatorar o handleInputChange, do NewForm.....	138
Refatorar o handleTurnDirty, do NewForm.....	138
Visualização web.....	139
GitHub-22.....	139
IMPEDIR ENVIO DE CAMPO E FORMULÁRIO EM BRANCO (DIRTY_ALL).....	140
Simulação.....	140
Implementar a função toDirtyAll, no utilitário forms.....	142
Implementar a função validateAll, no utilitário forms.....	142
Implementar a função dirtyAndValidateAll, no forms do utils.....	143
Implementar a função hasAnyInvalid, no forms do utils.....	143
Implementar o formDataValidated, na função handleSubmit, do NewForm.....	144
Visualização web.....	145
Enviar formulário em branco.....	145
Enviar formulário com alguma pendência.....	146
GitHub-23.....	147
PAGINAÇÃO.....	148
IMPLEMENTAR O SEED DO BACKEND.....	148
Implementar o import.sql.....	148
Testar a paginação no postman.....	149
VISUALIZAR A RENDERIZAÇÃO NO FRONTEND.....	150

INSTALAR O PLUGIN DE PAGINAÇÃO.....	151
URL do react paginate:.....	151
Instalação.....	151
Rodar o projeto.....	152
IMPLEMENTAR O PAGINATION.....	152
Criar o componente de paginação.....	152
Implementar o pagination na página Listing.....	153
Visualização web.....	153
PAGINATION LAYOUT.....	154
Estilizar o layout da paginação no Listing.....	154
Estilizar o layout da paginação no Pagination.....	154
Visualização web.....	155
Retirar os marcadores no index.css.....	155
Visualização web.....	155
Aplicar estilo de cor ao número da página atual.....	156
Estilizar.....	156
Visualização web.....	156
Inativar o “previous” e o “next”, quando do início e fim da paginação.....	157
Visualização web.....	157
GitHub-24.....	158
RENDERIZAR A PAGINAÇÃO DINAMICAMENTE.....	158
Renderizar a quantidade de páginas no componente Pagination.....	158
Criando o props no componente pagination.....	158
Refatorar o student-service para dividir as páginas por quantidade de registro.....	159
Implementar um estado para pegar a quantidade de páginas.....	159
Renderizar pageCounts no Pagination, do Listing.....	160
Visualização web.....	160
GitHub-25.....	160
Renderizar a paginação na URL.....	161
Instalar o pacote query-string.....	161
Implementar os Hooks useNavigate e useLocation.....	162
Implementar uma função e um estado para a página atual.....	162
Implementar o useEffect para monitorar a mudança de página.....	163
Chamar o getActualPage no queryParams.....	163
Visualização web.....	164
GitHub-26.....	165
Renderizar a paginação no componente pagination.....	166
Criar um evento, no pagination, que irá disparar quando a página mudar.....	166
Implementar o handlePageChange e refatorar o useEffect, na página Listing.....	167
Chamar o handlePageChange no Pagination do Listing.....	167

Visualização web.....	168
GitHub-27.....	169
SISTEMA FINALIZADO.....	170
Home.....	170
Listagem de Alunos.....	171
Cadastro de alunos.....	172
Deletar Aluno.....	173
FIM.....	173

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

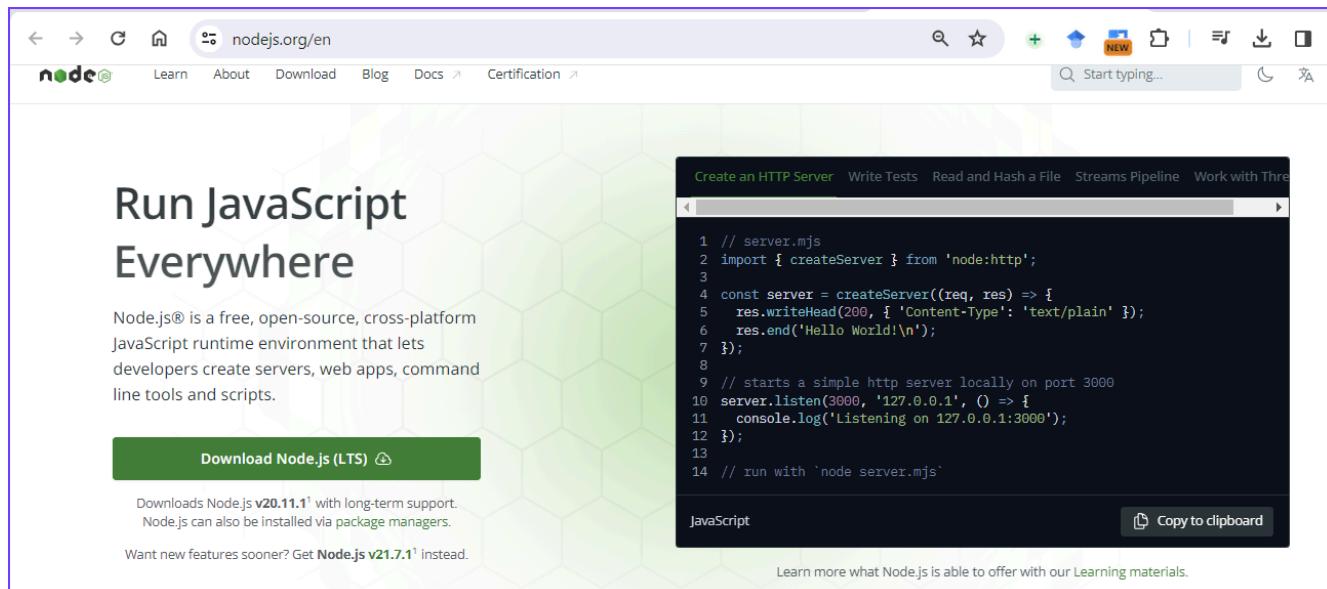
- VISUALIZAR TODOS OS ALUNOS NUMA TABELA
- INSERIR UM NOVO ALUNO
- EDITAR UM ALUNO EXISTENTE
- DELETAR UM ALUNO EXISTENTE

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

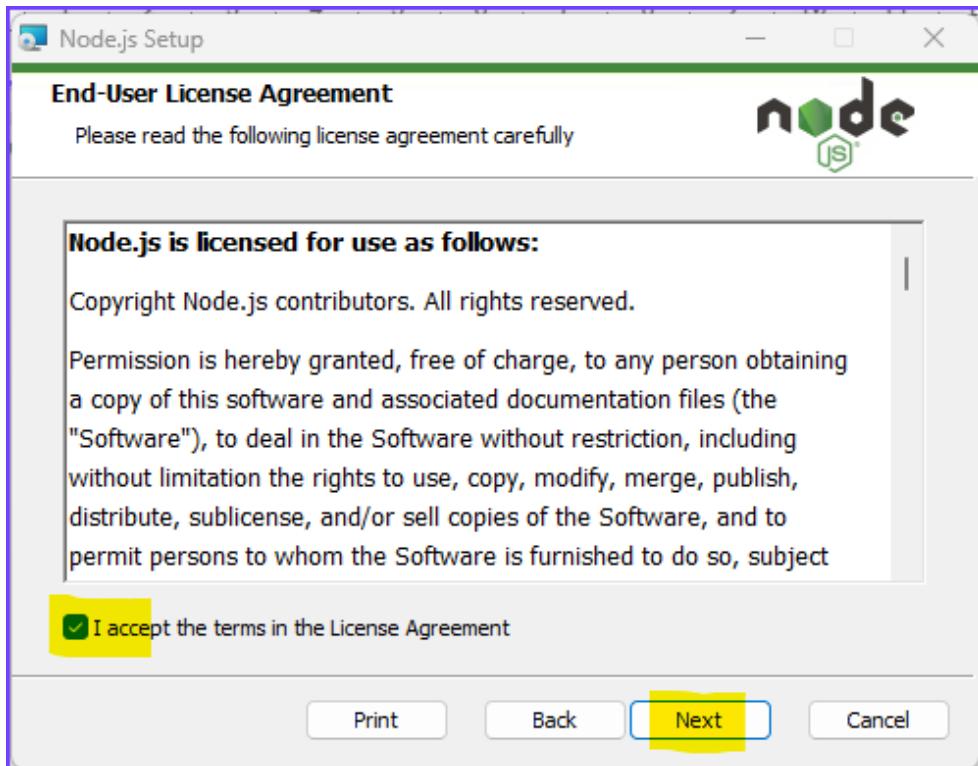
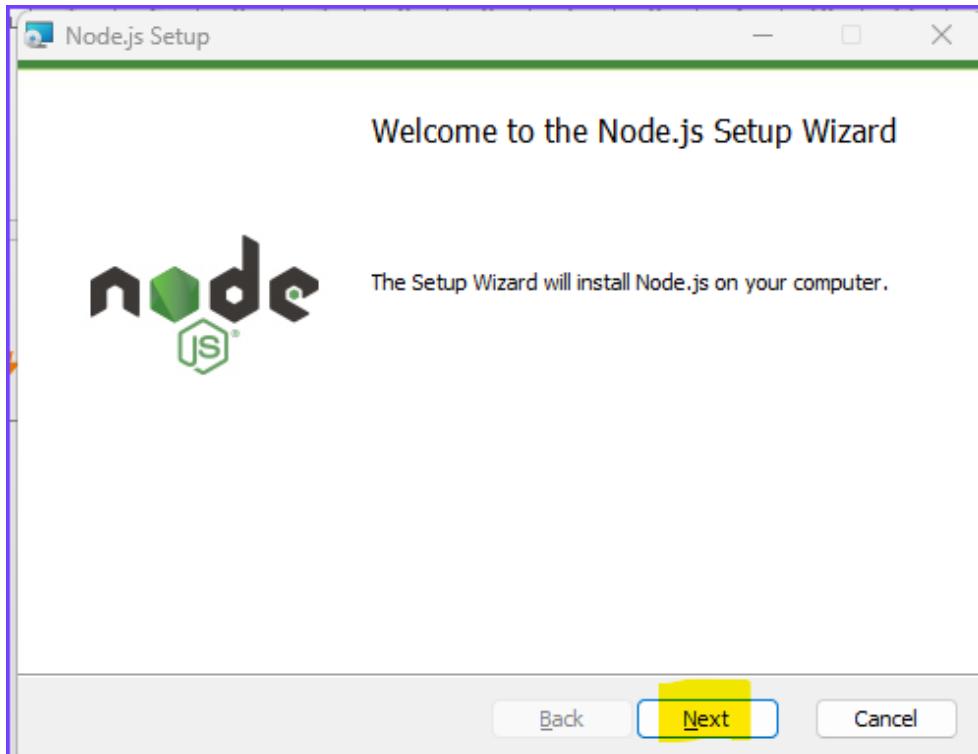
INSTALAÇÃO DO NODEJS

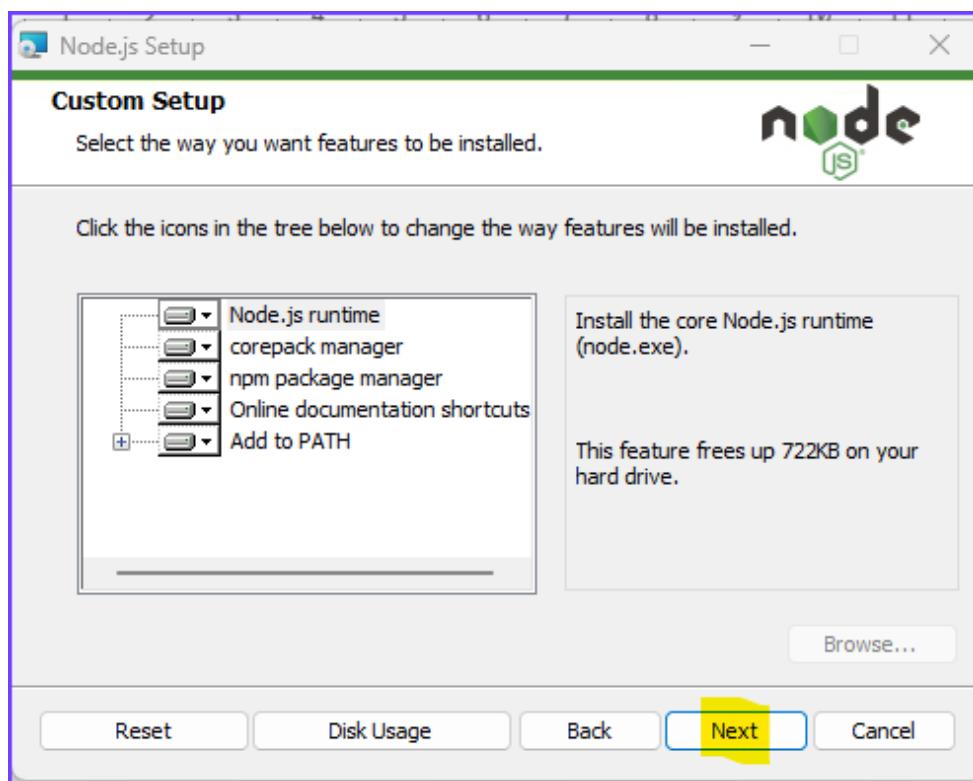
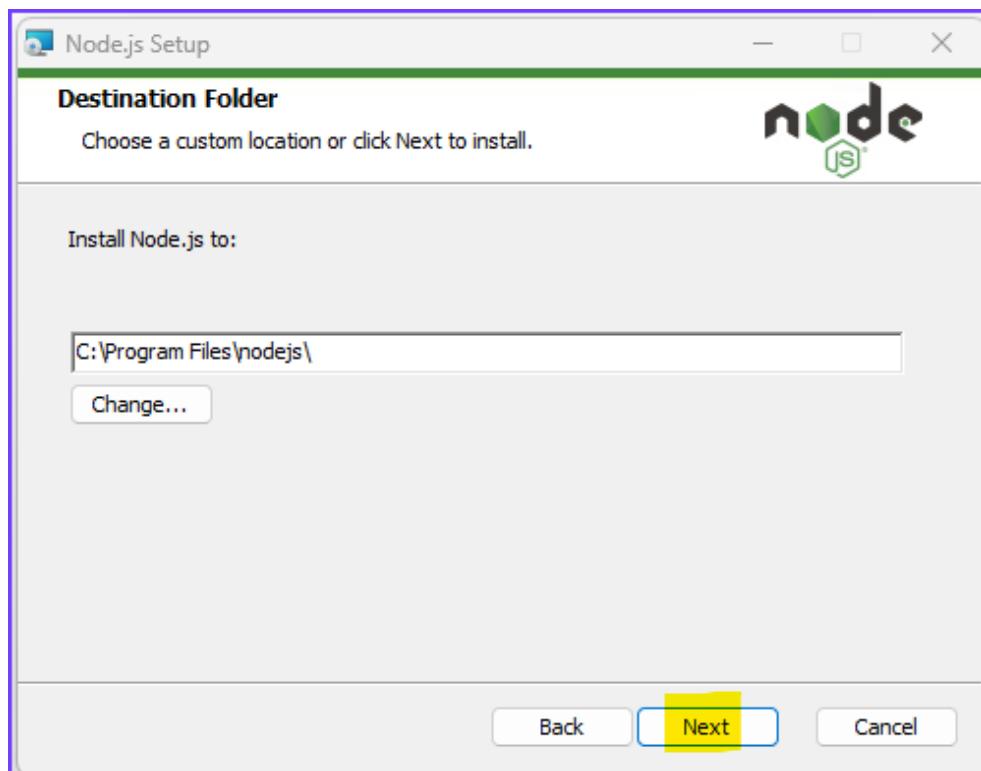
BAIXAR

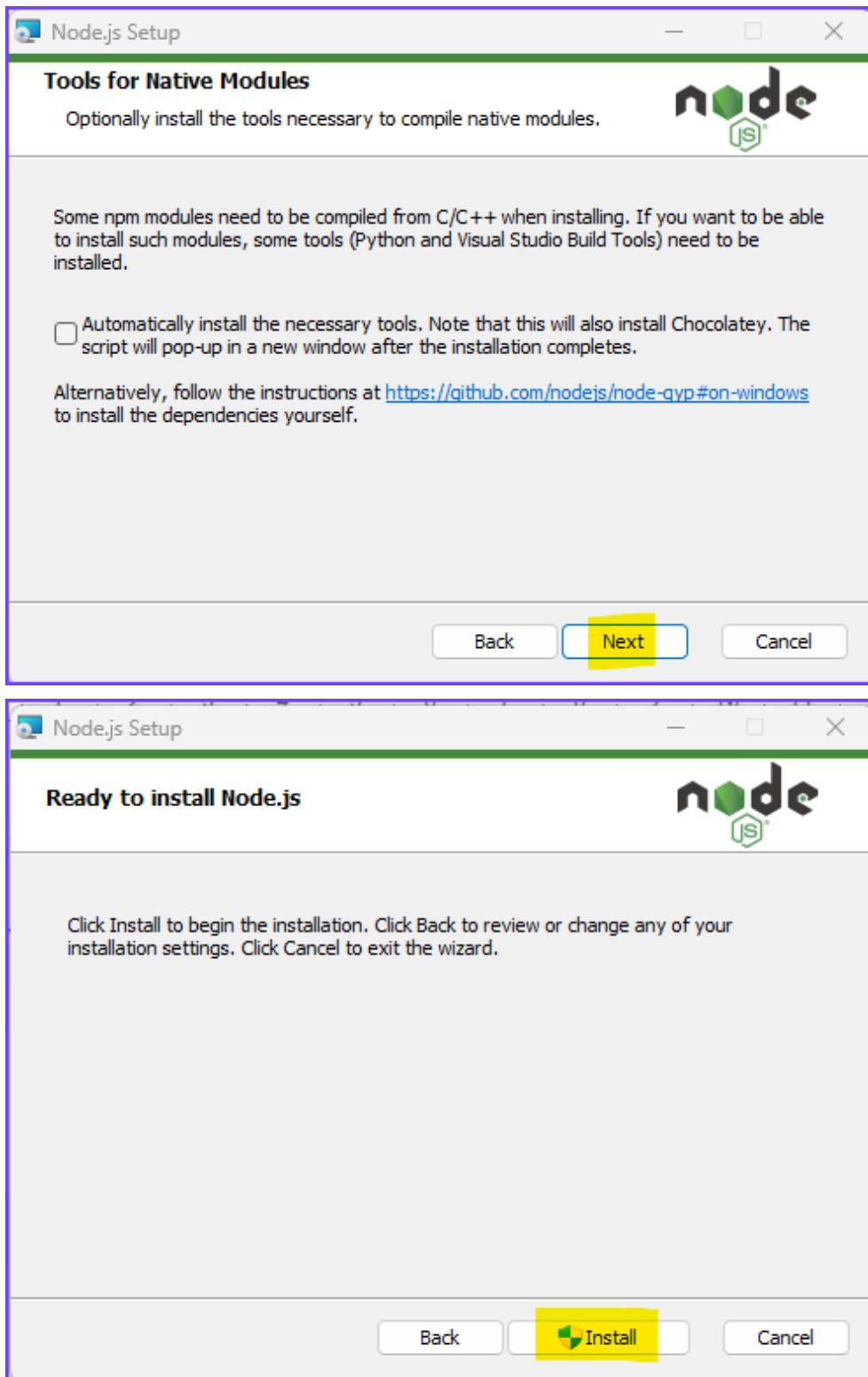
- <https://nodejs.org>

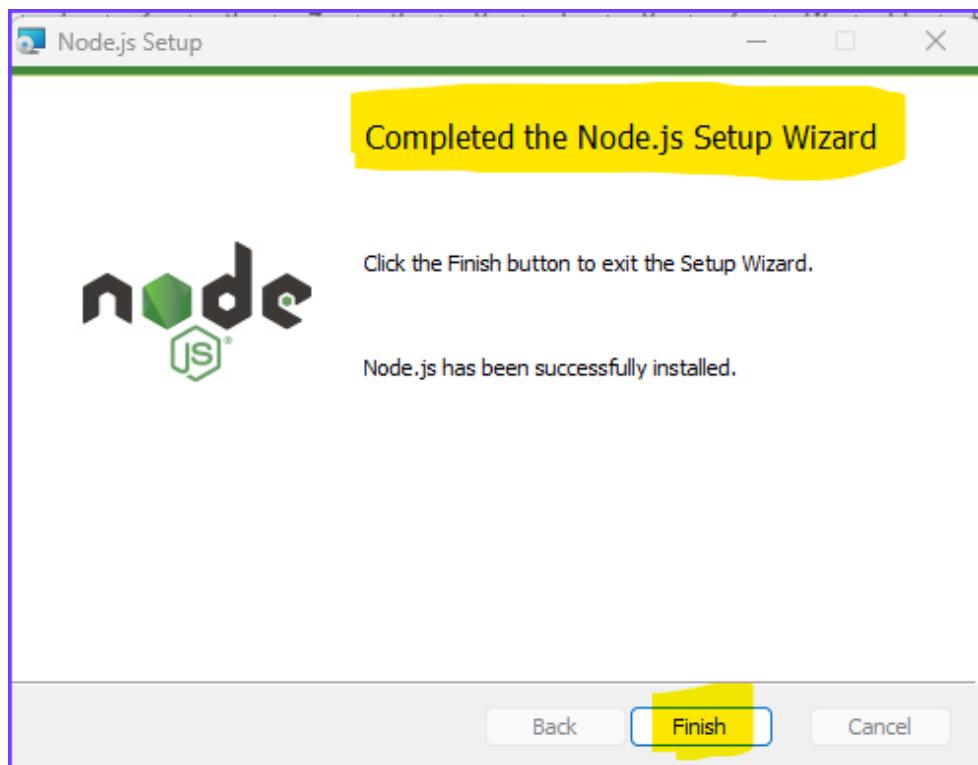


EXECUTAR









TESTAR NO TERMINAL

- npm -v

A screenshot of a Windows Command Prompt window titled "Prompt de Comando". The title bar also shows "Microsoft Windows [versão 10.0.22621.3296]" and "(c) Microsoft Corporation. Todos os direitos reservados.". The command "C:\Users\auric>npm -v" is entered, followed by the output "10.2.4". The prompt then changes to "C:\Users\auric>".

IMPLEMENTAR O YARN

INSTALAR

- npm install --global yarn

```
C:\Users\auric>npm install --global yarn

changed 1 package in 1s
npm notice
npm notice New minor version of npm available! 10.2.4 -> 10.5.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v10.5.0
npm notice Run npm install -g npm@10.5.0 to update!
npm notice
```

TESTAR

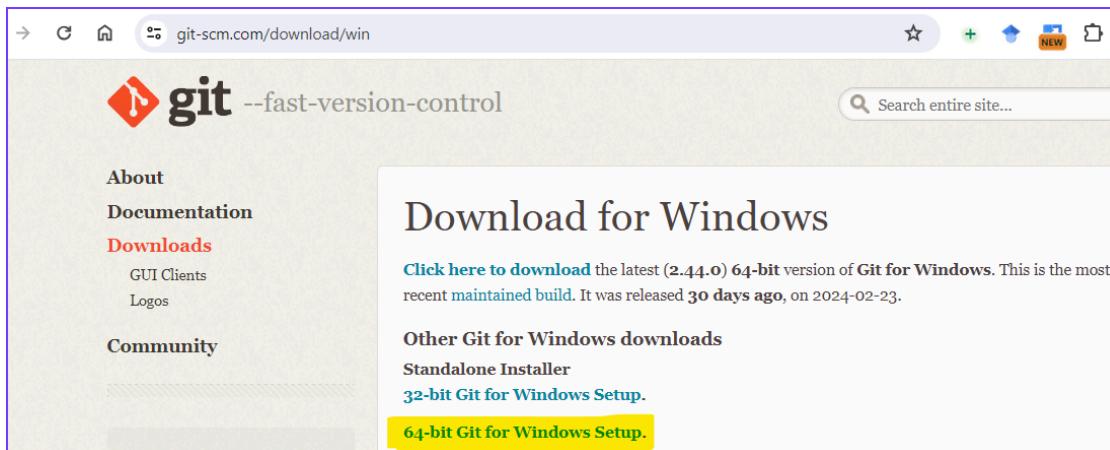
- yarn -v

```
C:\Users\auric>yarn -v
1.22.22
```

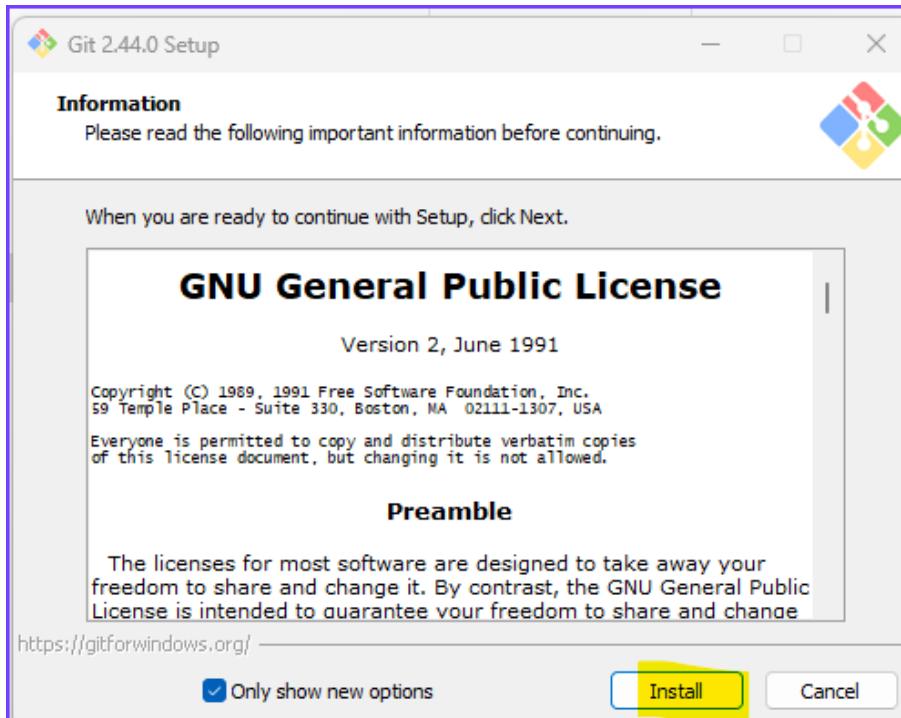
IMPLEMENTAR O GIT

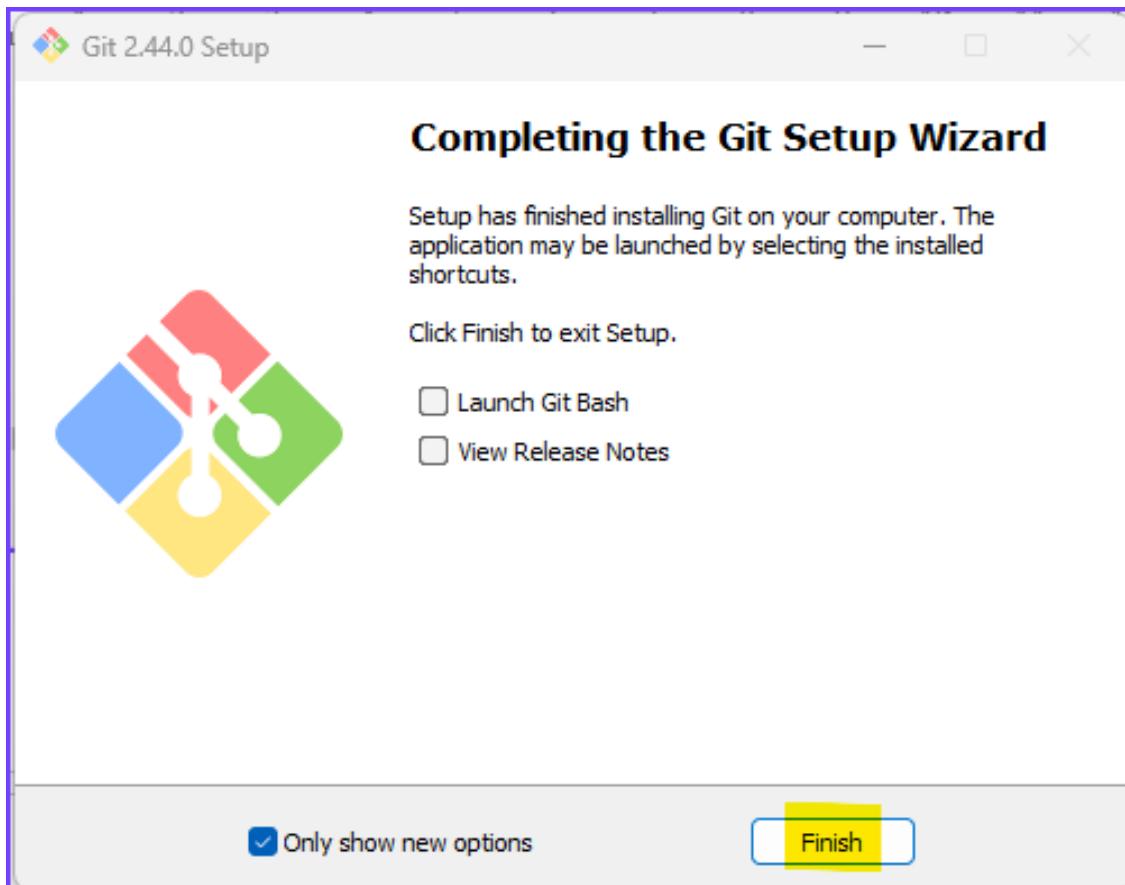
BAIXAR

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



INSTALAR





TESTAR

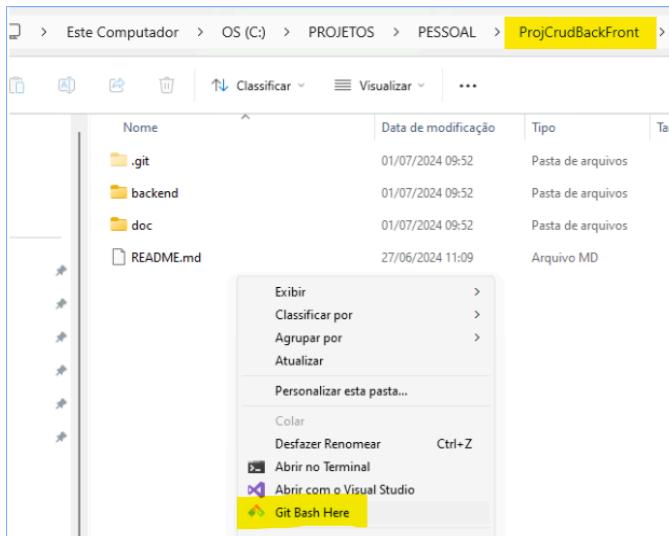
- git -v

The screenshot shows a Microsoft Windows Command Prompt window titled "Prompt de Comando". The window displays the following text:
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

```
MINGW64:/c/PROJETOS/PESSOAL/ProjCrudBackFront
SAP@DESKTOP-LSDERN6 MINGW64 /c/PROJETOS/PESSOAL/ProjCrudBackFront (main)
$ yarn create vite frontend --template react-ts
yarn create v1.22.22
[1/4] Resolving packages...
[2/4] Fetching packages...
[3/4] Linking dependencies...
[4/4] Building fresh packages...
success Installed "create-vite@5.3.0" with binaries:
  - create-vite
  - cva

Scaffolding project in C:\PROJETOS\PESSOAL\ProjCrudBackFront\frontend...

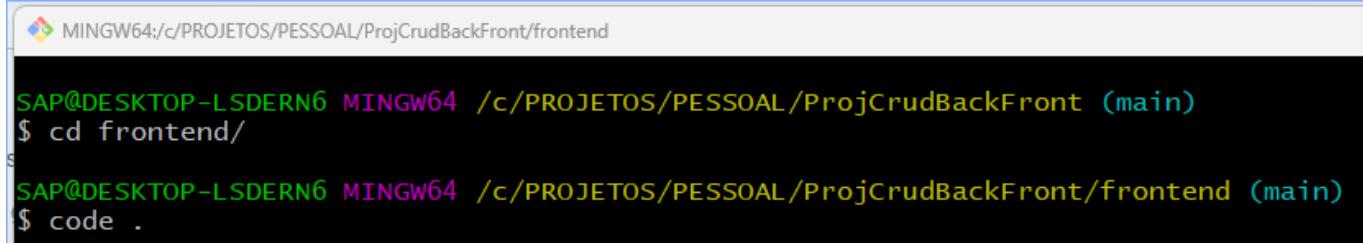
Done. Now run:

  cd frontend
  yarn
  yarn dev

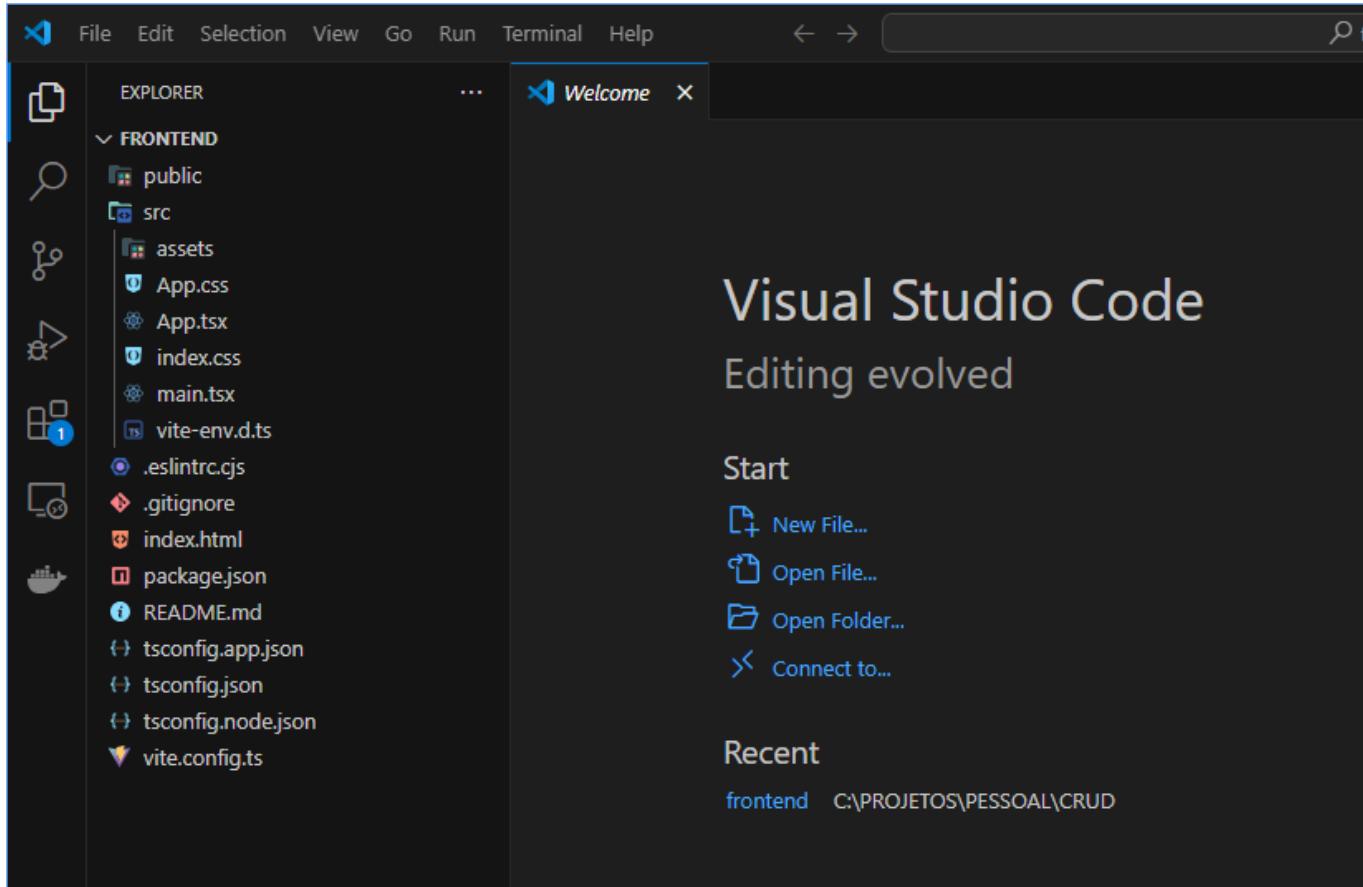
Done in 0.86s.
```

ABRIR COM O VS CODE

- cd /frontend
- code .



```
MINGW64:/c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend
SAP@DESKTOP-LSDERN6 MINGW64 /c/PROJETOS/PESSOAL/ProjCrudBackFront (main)
$ cd frontend/
SAP@DESKTOP-LSDERN6 MINGW64 /c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend (main)
$ code .
```



The screenshot shows the Visual Studio Code interface. On the left is the Explorer sidebar with a tree view of the project structure:

- FRONTEND folder
 - public
 - src
 - assets
 - App.css
 - App.tsx
 - index.css
 - main.tsx
 - vite-env.d.ts
 - .eslintrc.cjs
 - .gitignore
 - index.html
 - package.json
 - README.md
 - tsconfig.app.json
 - tsconfig.json
 - tsconfig.node.json
 - vite.config.ts

The main workspace shows the "Welcome" screen with the text "Visual Studio Code" and "Editing evolved". Below it are "Start" and "Recent" sections with options like "New File...", "Open File...", "Open Folder...", and "Connect to...".

DEPENDÊNCIAS

ATUALIZAR AS DEPENDÊNCIAS DO YARN

- yarn

```
MINGW64:/c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend
SAP@DESKTOP-LSDERN6 MINGW64 /c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend (main)
$ yarn
yarn install v1.22.22
info No lockfile found.
[1/4] Resolving packages...
warning eslint > @humanwhocodes/config-array@0.11.14: Use @eslint/config-array instead
warning eslint > @humanwhocodes/config-array > @humanwhocodes/object-schema@2.0.3: Instead
warning eslint > file-entry-cache > flat-cache > rimraf@3.0.2: Rimraf versions prioritized
warning eslint > file-entry-cache > flat-cache > rimraf > glob@7.2.3: Glob versions supported
warning eslint > file-entry-cache > flat-cache > rimraf > glob > inflight@1.0.6: This and leaks memory. Do not use it. Check out lru-cache if you want a good and tested
tests by a key value, which is much more comprehensive and powerful.
[2/4] Fetching packages...
[3/4] Linking dependencies...
[4/4] Building fresh packages...
success Saved lockfile.
Done in 34.75s.
```

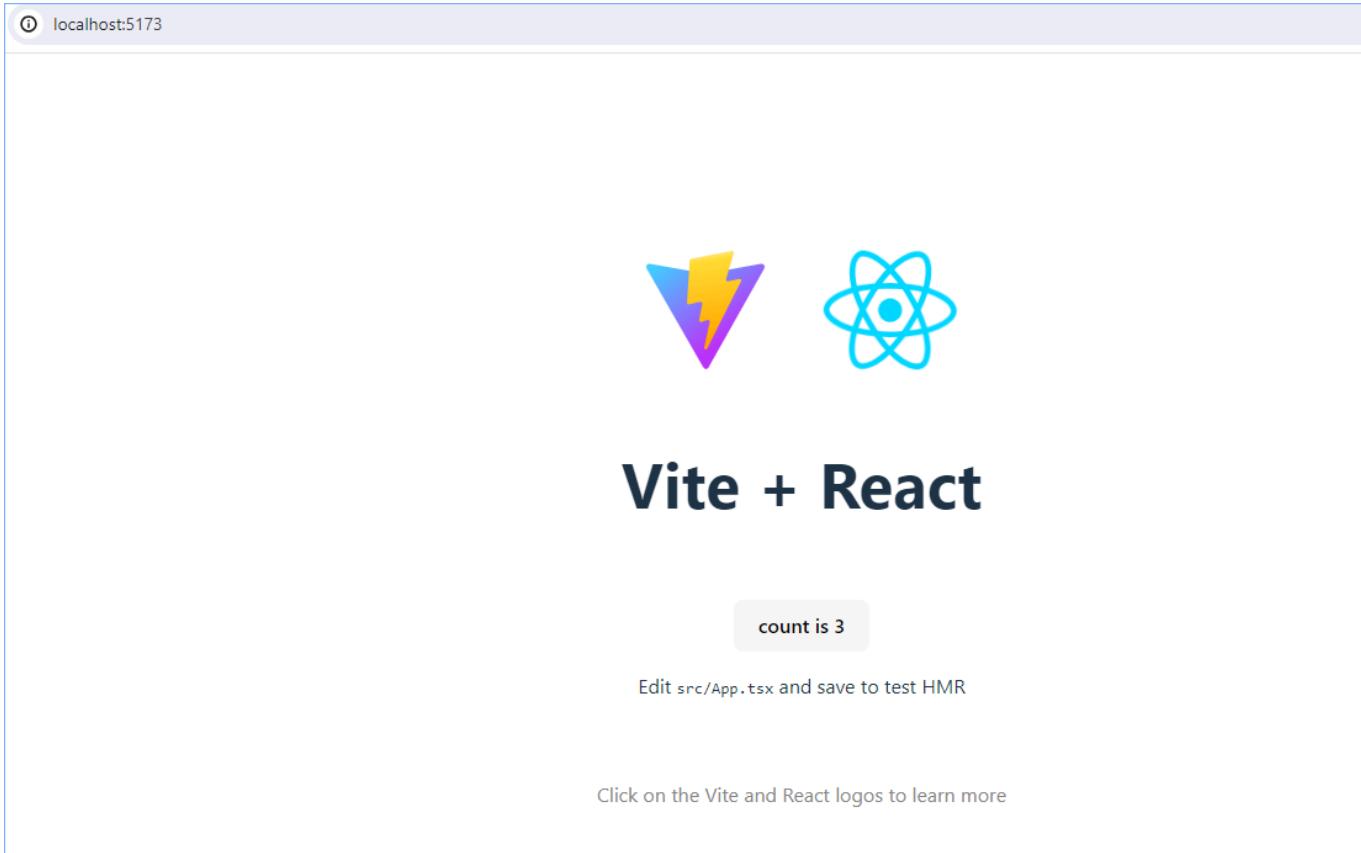
RODAR O PROJETO

- yarn dev

```
MINGW64:/c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend
SAP@DESKTOP-LSDERN6 MINGW64 /c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend (main)
$ yarn dev
yarn run v1.22.22
$ vite

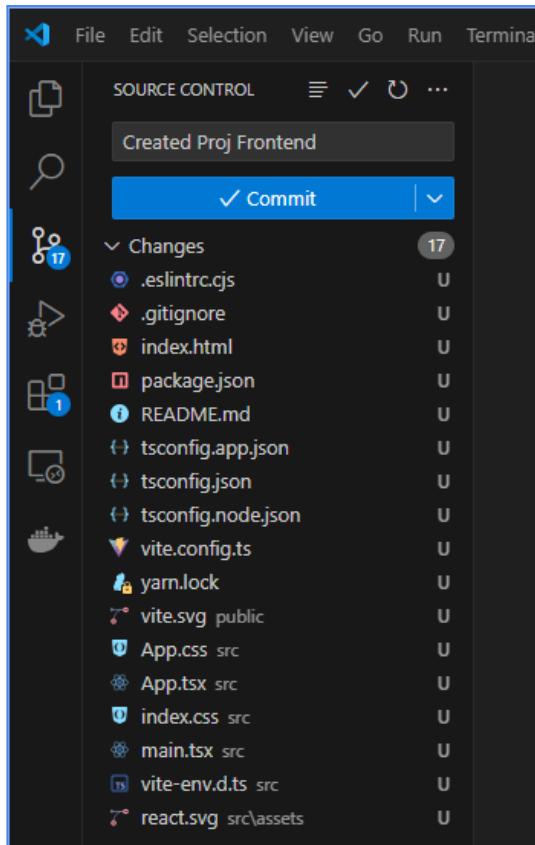
VITE v5.3.2 ready in 137 ms
→ Local:  http://localhost:5173/
→ Network: use --host to expose
```

VISUALIZAÇÃO WEB



GitHub-1

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



A screenshot of a GitHub repository page. The repository name is 'pessoal-ProjCrudBackFront' and it is public. At the top right, there are 'Pin' and 'Unwatch' buttons. Below the repository name, there are buttons for 'main' (with a dropdown arrow), '1 Branch', '0 Tags', 'Go to file' (with a search icon), 'Add file' (with a plus icon), and 'Code' (with a code icon). The commit history table has columns for file/folder, commit message, author, date, and time. The commits listed are: 'auriceliof Merge branch 'main' of https://github.com/auriceliof/pessoal-ProjCrud... now 042d1db · 22 Commits', 'backend Implemented Swagger 16 hours ago', 'doc Add Tutorial Backend 15 hours ago', 'frontend Created Proj Frontend now', and 'README.md Update README.md 1 hour ago'.

LIMPAR O PROJETO

INDEX.HTML

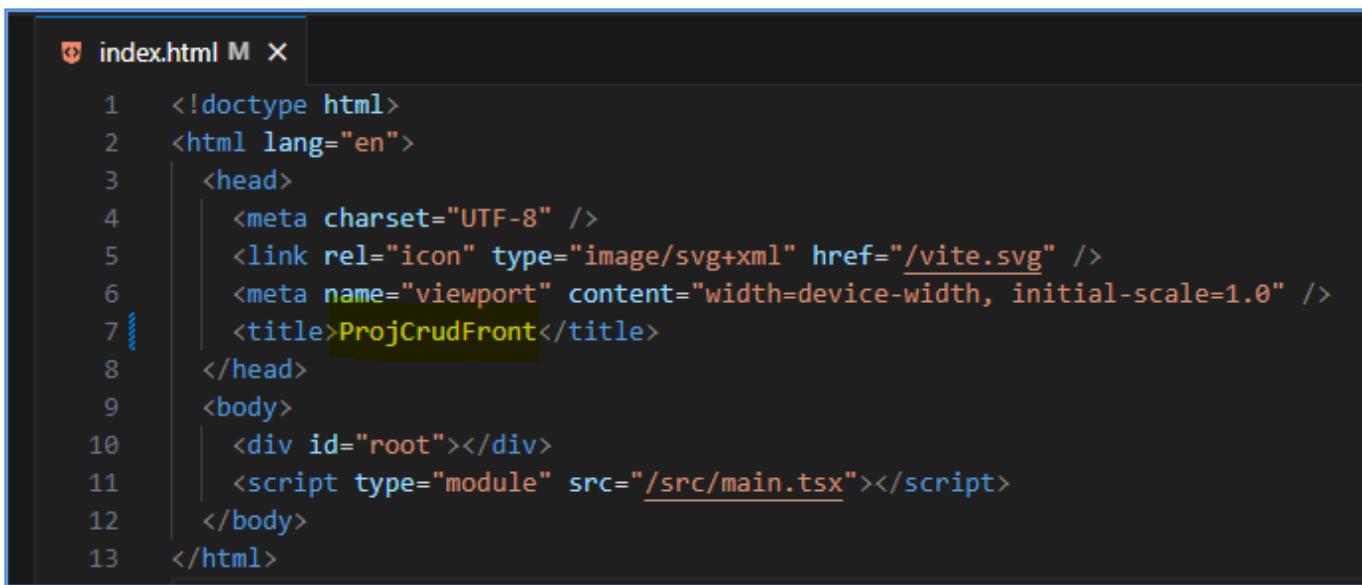
Antes



```
index.html X
1  <!doctype html>
2  <html lang="en">
3  |  <head>
4  |  |  <meta charset="UTF-8" />
5  |  |  <link rel="icon" type="image/svg+xml" href="/vite.svg" />
6  |  |  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7  |  |  <title>Vite + React + TS</title>
8  |  </head>
9  |  <body>
10 |  |  <div id="root"></div>
11 |  |  <script type="module" src="/src/main.tsx"></script>
12 |  </body>
13 |  </html>
14
```

Depois

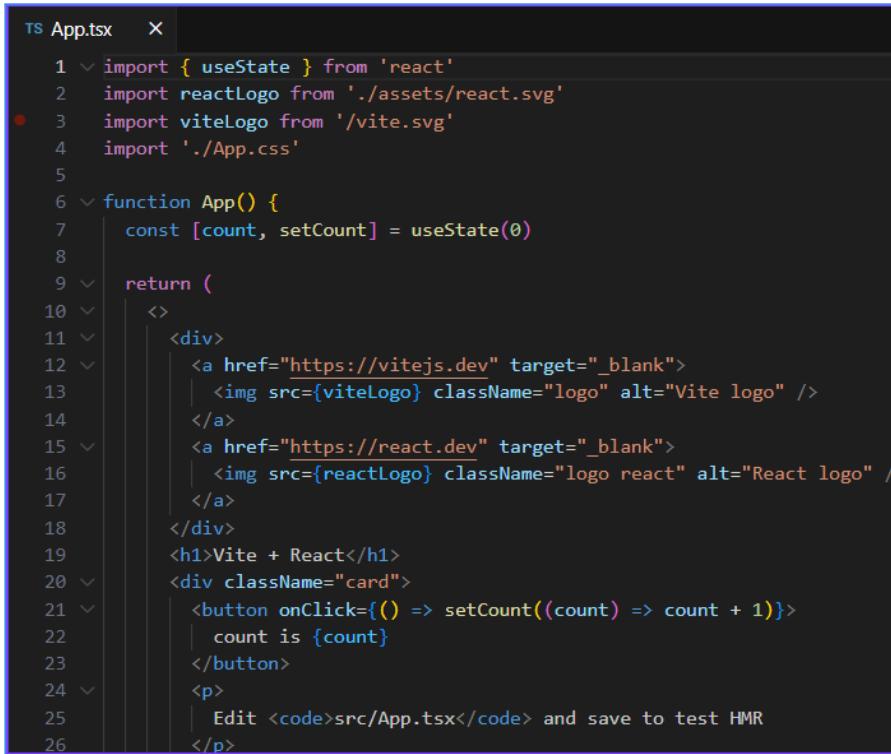
- Mudar o nome do projeto



```
index.html M X
1  <!doctype html>
2  <html lang="en">
3  |  <head>
4  |  |  <meta charset="UTF-8" />
5  |  |  <link rel="icon" type="image/svg+xml" href="/vite.svg" />
6  |  |  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7  |  |  <title>ProjCrudFront</title>
8  |  </head>
9  |  <body>
10 |  |  <div id="root"></div>
11 |  |  <script type="module" src="/src/main.tsx"></script>
12 |  </body>
13 |  </html>
```

APP.TSX

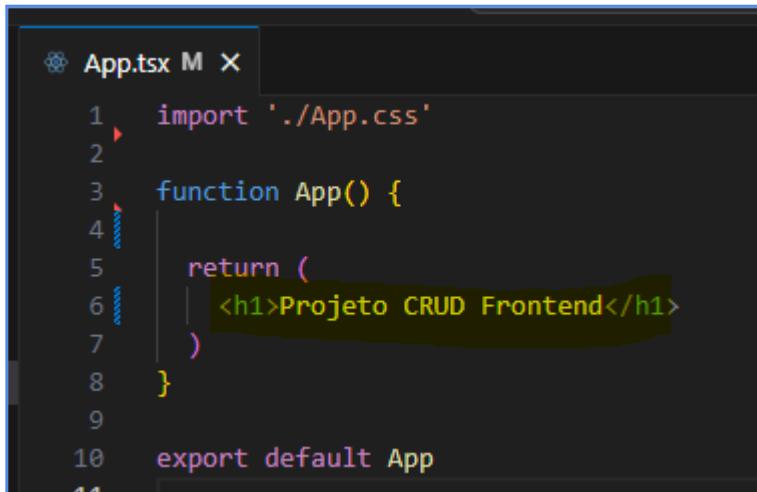
Antes



```
ts App.tsx  X
1 import { useState } from 'react'
2 import reactLogo from './assets/react.svg'
3 import viteLogo from '/vite.svg'
4 import './App.css'
5
6 function App() {
7   const [count, setCount] = useState(0)
8
9   return (
10    <>
11      <div>
12        <a href="https://vitejs.dev" target="_blank">
13          <img src={viteLogo} className="logo" alt="Vite logo" />
14        </a>
15        <a href="https://react.dev" target="_blank">
16          <img src={reactLogo} className="logo react" alt="React logo" />
17        </a>
18      </div>
19      <h1>Vite + React</h1>
20      <div className="card">
21        <button onClick={() => setCount((count) => count + 1)}>
22          count is {count}
23        </button>
24      <p>
25        Edit <code>src/App.tsx</code> and save to test HMR
26      </p>

```

Depois



```
App.tsx M X
1 import './App.css'
2
3 function App() {
4
5   return (
6     <h1>Projeto CRUD Frontend</h1>
7   )
8 }
9
10 export default App
11
```

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

INDEX.CSS

Antes

index.css X

```
1  :root {  
2      font-family: Inter, system-ui, Avenir, Helvetica, Arial, sans-serif;  
3      line-height: 1.5;  
4      font-weight: 400;  
5  
6      color-scheme: light dark;  
7      color: #242424;  
8      background-color: #242424;  
9  
10     font-synthesis: none;  
11     text-rendering: optimizeLegibility;  
12     -webkit-font-smoothing: antialiased;  
13     -moz-osx-font-smoothing: grayscale;  
14 }  
15  
16 a {  
17     font-weight: 500;  
18     color: #646cff;  
19     text-decoration: inherit;  
20 }  
21 a:hover {  
22     color: #535bf2.  
23 }
```

Depois

index.css M X

```
1  :root {  
2      --color-red: #ff0000;  
3 }  
4
```

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

APP.CSS

Antes

```
# App.css X
1  ✓ #root {
2      max-width: 1280px;
3      margin: 0 auto;
4      padding: 2rem;
5      text-align: center;
6  }
7
8  ✓ .logo {
9      height: 6em;
10     padding: 1.5em;
11     will-change: filter;
12     transition: filter 300ms;
13 }
14 ✓ .logo:hover {
15     filter: drop-shadow(0 0 2em #646cffaa);
```

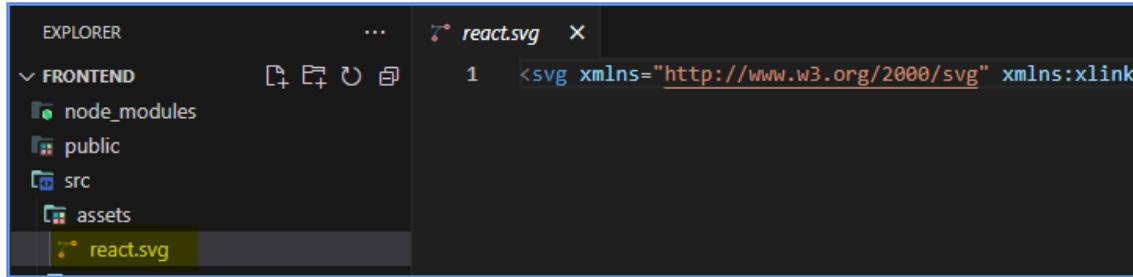
Depois

```
# App.css X
1   h1 {
2     color: var(--color-red);
3 }
```

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

ASSETS

Antes

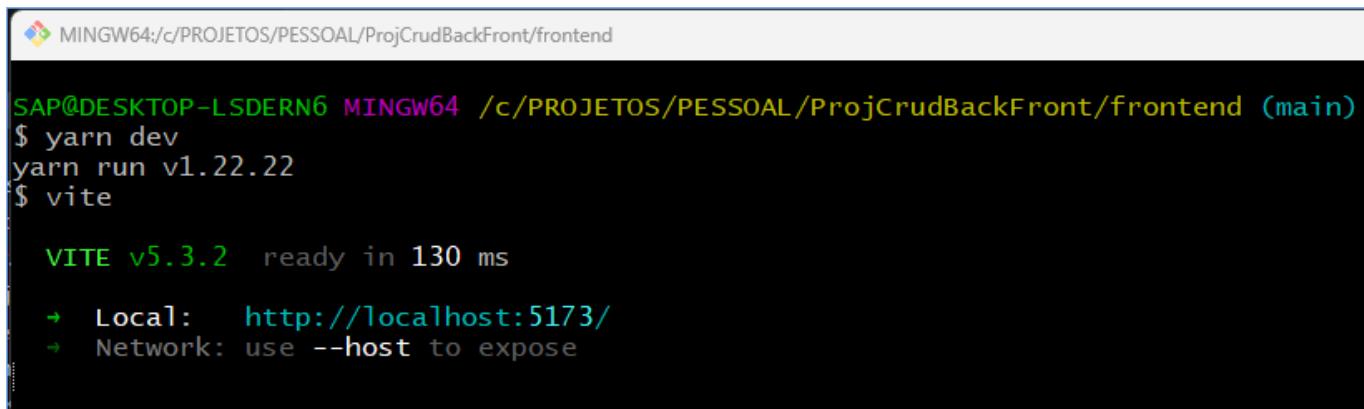


A screenshot of a code editor interface. On the left, the Explorer sidebar shows a project structure under 'FRONTEND': 'node_modules', 'public', 'src', and 'assets'. The 'assets' folder contains a file named 'react.svg', which is highlighted with a yellow background. On the right, there is a single open file tab titled 'react.svg' containing the XML code for an SVG element.

Depois

- *Deletar imagem.*

RODAR O PROJETO

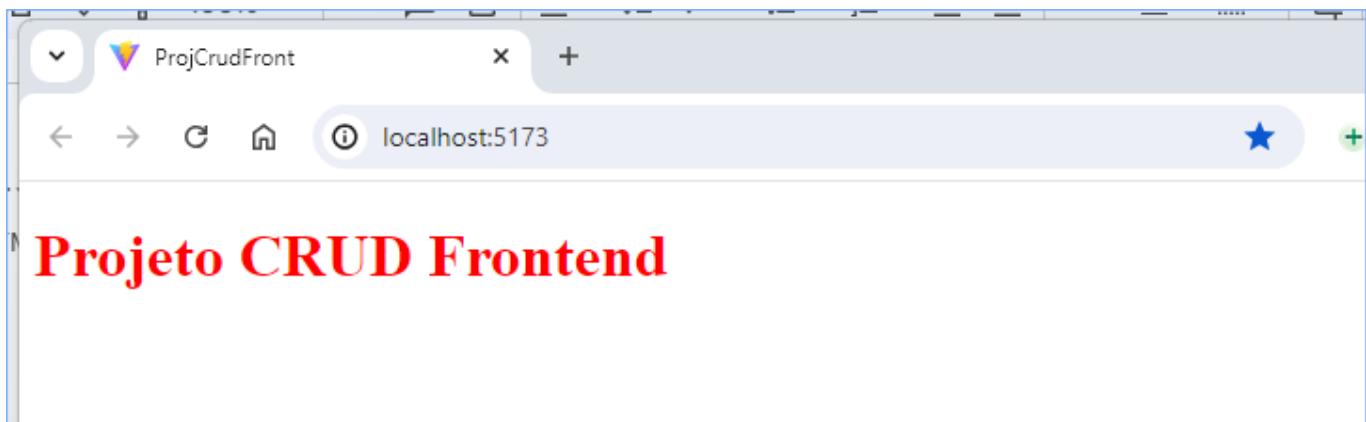


A screenshot of a terminal window. The title bar indicates the path: 'MINGW64:/c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend'. The terminal output shows the command '\$ yarn dev' being run, followed by the output of Vite v5.3.2 starting up. It displays the local host URL as 'http://localhost:5173/' and provides instructions for exposing the network.

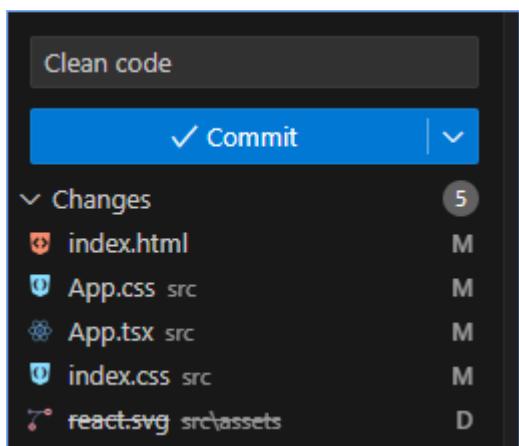
```
SAP@DESKTOP-LSDERN6 MINGW64 /c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend (main)
$ yarn dev
yarn run v1.22.22
$ vite

VITE v5.3.2 ready in 130 ms
→ Local: http://localhost:5173/
→ Network: use --host to expose
```

VISUALIZAÇÃO WEB



GitHub-2

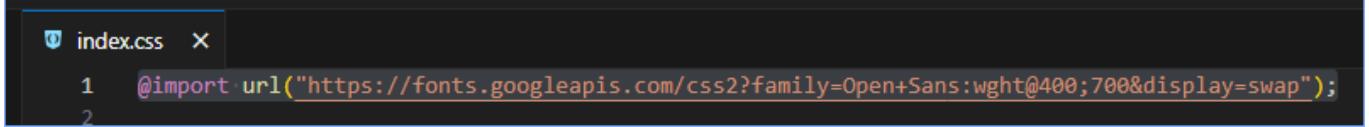


CSS

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  X
1  @import url("https://fonts.googleapis.com/css2?family=Open+Sans:wght@400;700&display=swap");
2
```

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;
}
```

The screenshot shows a code editor window with a dark theme. The file path 'src > index.css > ...' is visible at the top. The code itself is identical to the one provided above, with line numbers 30 through 51 displayed on the left side of the editor.

```
src > index.css > ...
30
31  * {
32    box-sizing: border-box;
33    margin: 0;
34    padding: 0;
35    font-family: "Open Sans";
36  }
37
38  a,
39  a:hover {
40    text-decoration: none;
41    color: unset;
42  }
43
44  html,
45  body {
46    background-color: var(--proj-color-bg-primary);
47  }
48
49  main {
50    padding: 0 20px;
51 }
```

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;
```

```
}
```

```
}
```

The screenshot shows a code editor window with a dark theme. The file is named 'index.css' and contains CSS styles for a project card. The styles include generic styles, a container width, a card height, and various margin and padding rules. A media query at the bottom targets a minimum width of 576px, changing the alignment and size for the section title and card height.

```
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: 380px;
```

```
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;
```

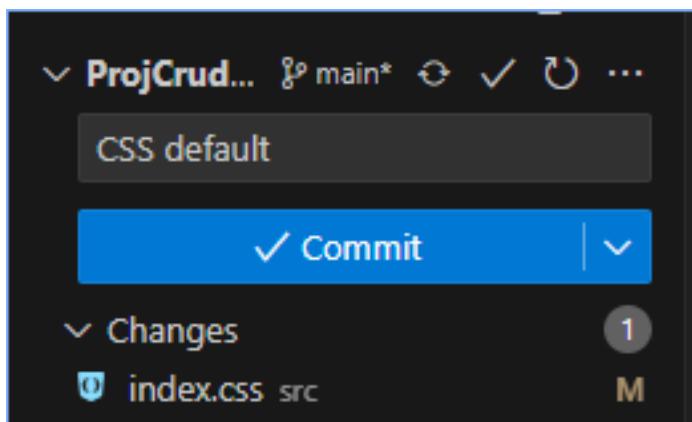
```
103    }
```

```
104  }
```

VISUALIZAÇÃO WEB



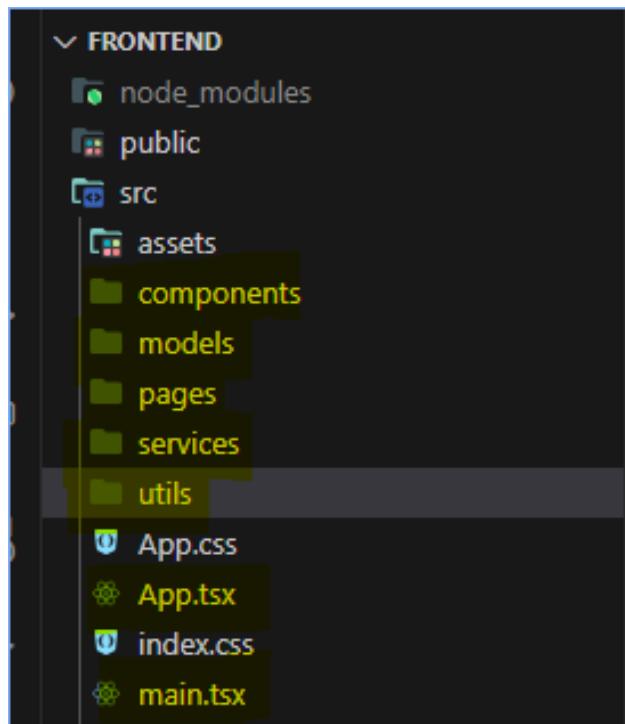
GitHub-3



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

The screenshot shows the VS Code interface. On the left, the Explorer sidebar displays a project structure under 'FRONTEND': node_modules, public, src (selected), assets, components, models, pages\Home (selected), index.tsx, and styles.css. The main editor window shows the code for 'index.tsx':

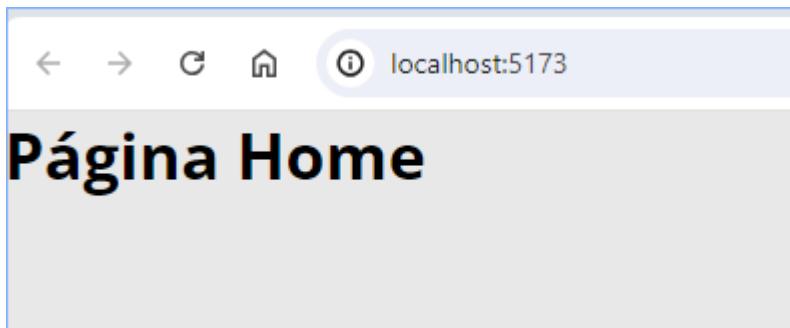
```
1 import './styles.css';
2
3 export default function Home() {
4
5     return (
6         <h1>Página Home</h1>
7     )
8 }
9
```

Renderizar a página no App.tsx

The screenshot shows the VS Code interface with the 'App.tsx' file open. The code defines a functional component 'App' that imports 'Home' from the 'pages/Home' directory:

```
1 import './App.css'
2 import Home from './pages/Home'
3
4 function App() {
5
6     return (
7         <Home />
8     )
9 }
10
11 export default App
12
```

Visualização web

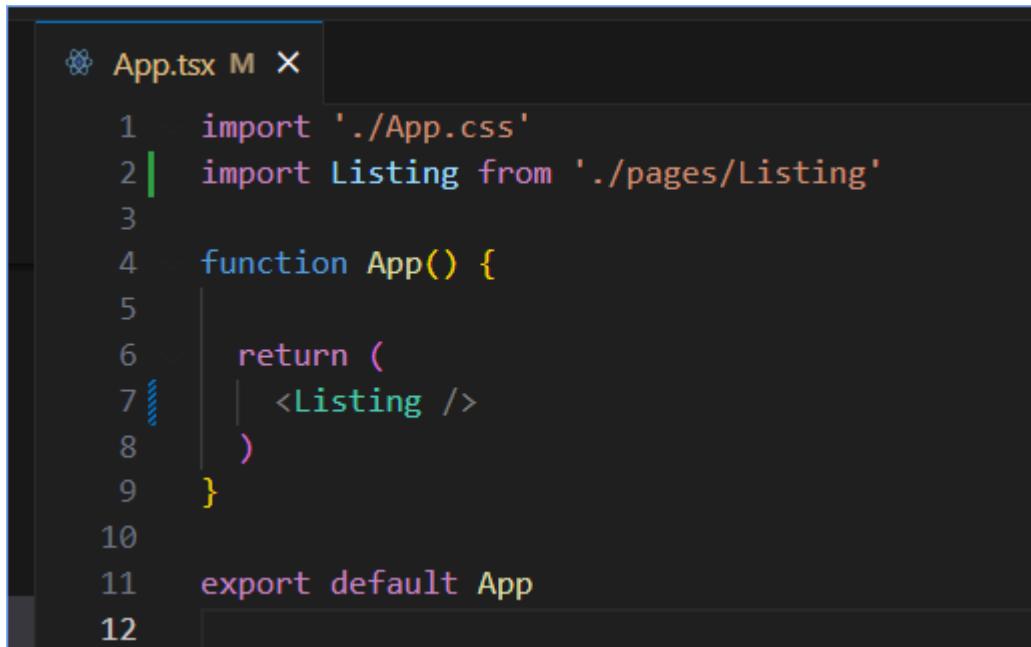


Listing

The image shows a code editor interface with a dark theme. On the left is the Explorer sidebar, which lists a project structure under 'FRONTEND': 'src' contains 'pages' (which has 'Listing' selected), 'index.tsx', and 'styles.css'; 'services' is also present. The right pane shows the content of 'index.tsx':

```
1 import './styles.css';
2
3 export default function Listing() {
4
5     return (
6         <h1>Página de Listagem</h1>
7     )
8 }
```

Renderizar a página no App.tsx



```
App.tsx M X
1 import './App.css'
2 import Listing from './pages/Listing'
3
4 function App() {
5
6     return (
7         <Listing />
8     )
9 }
10
11 export default App
12
```

Visualização web



NewForm

The screenshot shows the VS Code interface. On the left is the Explorer sidebar with a tree view of the project structure under 'FRONTEND'. The 'src' folder contains 'pages', which has 'Listing' and 'NewForm' subfolders, and 'index.tsx'. Other files like 'index.css' and 'styles.css' are also listed. The main editor area displays the code for 'index.tsx':

```
1 import './styles.css';
2
3 export default function NewForm() {
4
5     return (
6         <h1>Página de Novo Formulário</h1>
7     )
8 }
```

Renderizar a página no App.tsx

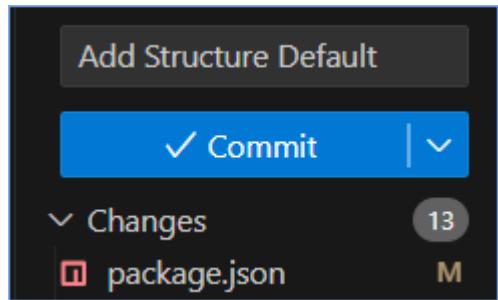
The screenshot shows the code for 'App.tsx' in the main editor area:

```
1 import './App.css'
2 import NewForm from './pages/NewForm'
3
4 function App() {
5
6     return (
7         <NewForm />
8     )
9 }
10
11 export default App
12
```

Visualização web



GitHub-4



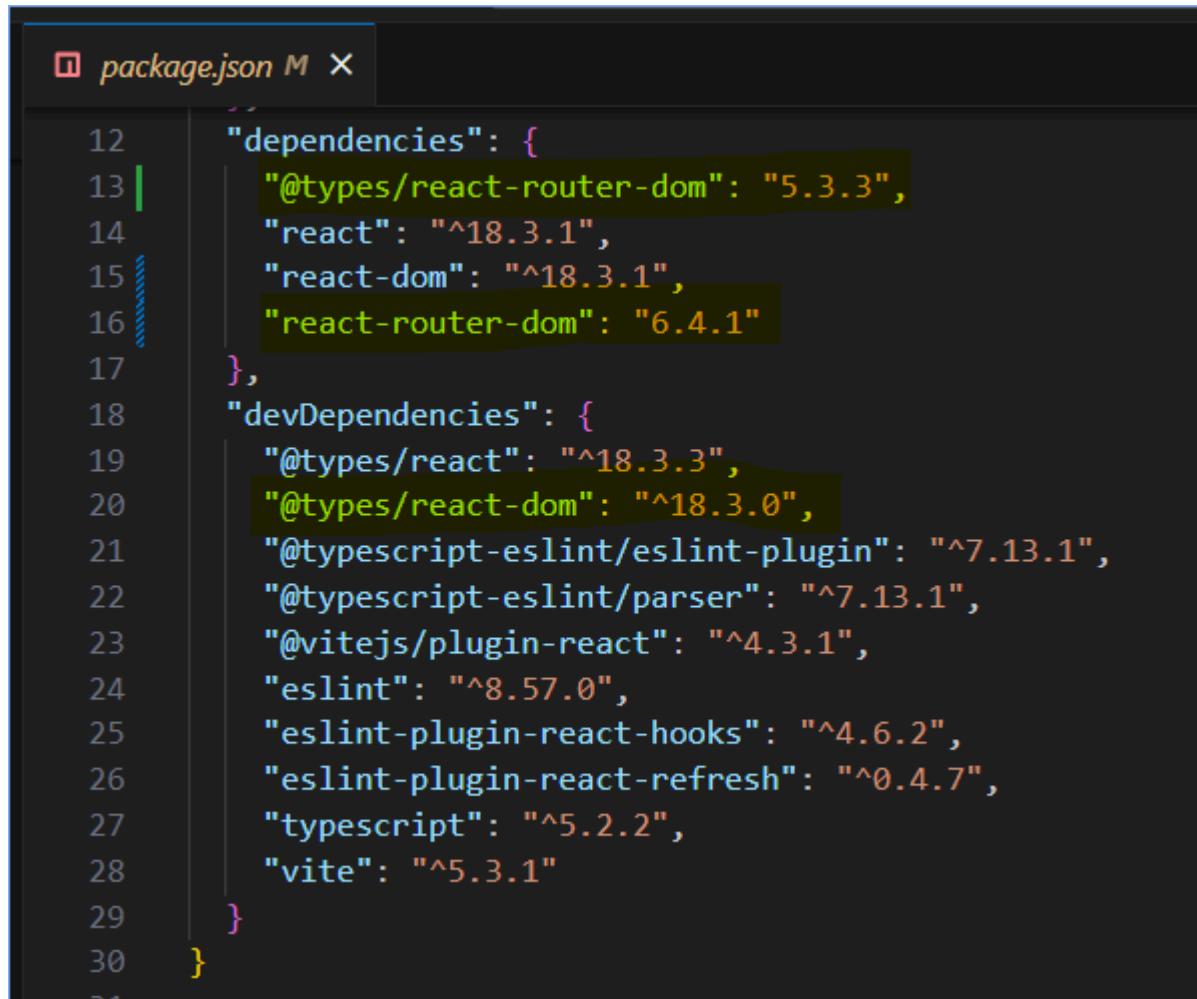
ROTAS

INSTALAR O REACT-ROUTER-DOM

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

```
MINGW64:/c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend
SAP@DESKTOP-LSDERN6 MINGW64 /c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend (main)
$ yarn add react-router-dom@6.4.1 @types/react-router-dom@5.3.3
yarn add v1.22.22
[1/4] Resolving packages...
[2/4] Fetching packages...
[3/4] Linking dependencies...
[4/4] Building fresh packages...
success Saved lockfile.
success Saved 4 new dependencies.
info Direct dependencies
└─ @types/react-router-dom@5.3.3
    └─ react-router-dom@6.4.1
info All dependencies
└─ @types/react-router-dom@5.3.3
   └─ @types/react-router@5.1.20
      └─ react-router-dom@6.4.1
         └─ react-router@6.4.1
Done in 2.91s.
```

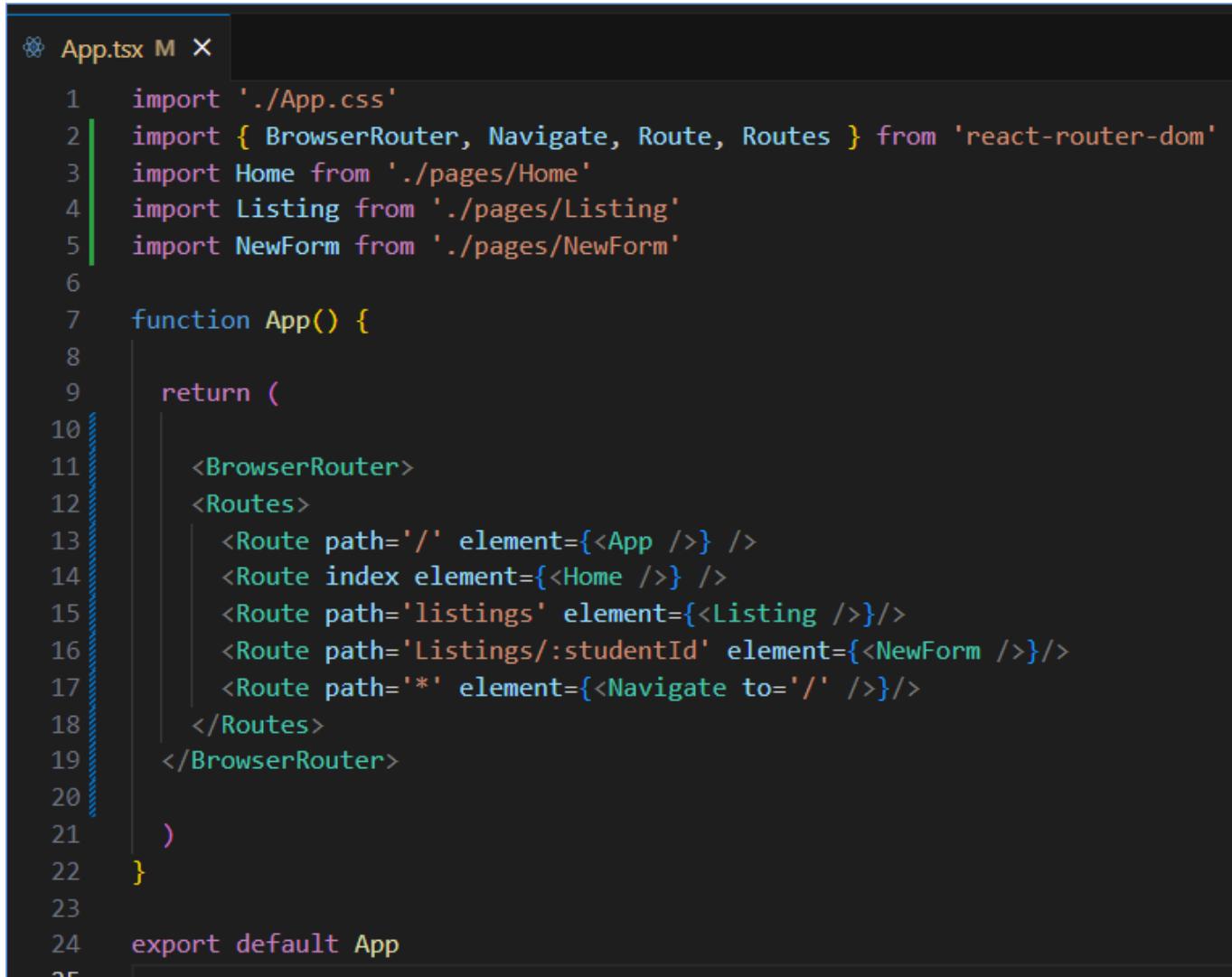
VERIFICAR NO PROJETO



A screenshot of a code editor showing the `package.json` file. The file contains the following JSON code:

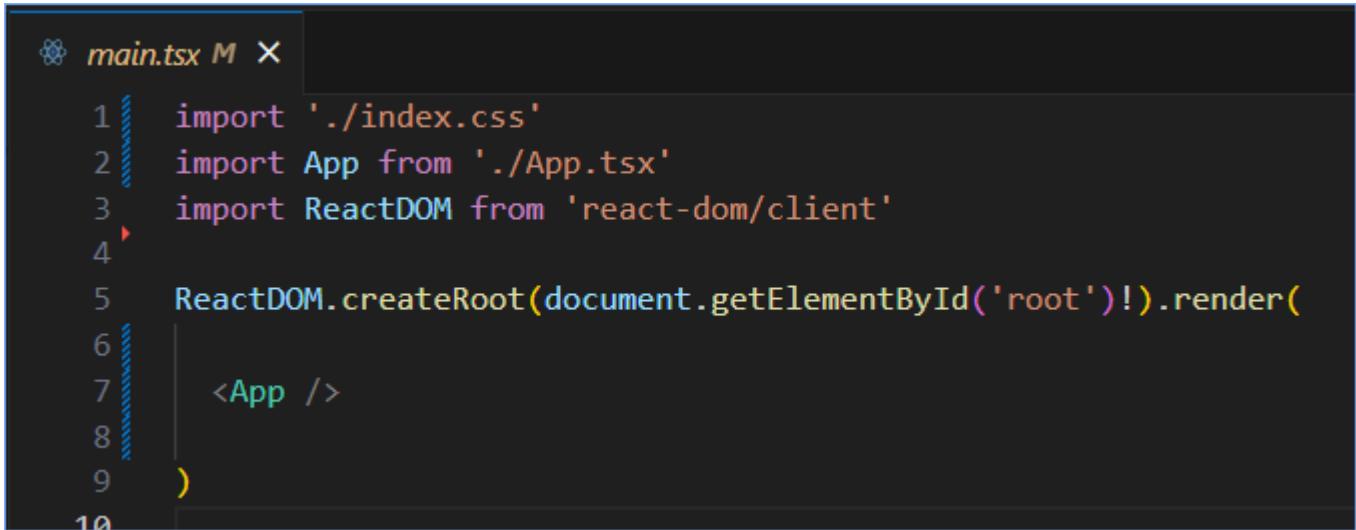
```
12 "dependencies": {  
13     "@types/react-router-dom": "5.3.3",  
14     "react": "^18.3.1",  
15     "react-dom": "^18.3.1",  
16     "react-router-dom": "6.4.1"  
17 },  
18 "devDependencies": {  
19     "@types/react": "^18.3.3",  
20     "@types/react-dom": "^18.3.0",  
21     "@typescript-eslint/eslint-plugin": "^7.13.1",  
22     "@typescript-eslint/parser": "^7.13.1",  
23     "@vitejs/plugin-react": "^4.3.1",  
24     "eslint": "^8.57.0",  
25     "eslint-plugin-react-hooks": "^4.6.2",  
26     "eslint-plugin-react-refresh": "^0.4.7",  
27     "typescript": "^5.2.2",  
28     "vite": "^5.3.1"  
29 }  
30 }  
31 }
```

IMPLEMENTAR AS ROTAS NO APP.TSX



```
App.tsx M X
1 import './App.css'
2 import { BrowserRouter, Navigate, Route, Routes } from 'react-router-dom'
3 import Home from './pages/Home'
4 import Listing from './pages/Listing'
5 import NewForm from './pages/NewForm'
6
7 function App() {
8
9     return (
10
11         <BrowserRouter>
12         <Routes>
13             <Route path='/' element={<App />} />
14             <Route index element={<Home />} />
15             <Route path='listings' element={<Listing />}/>
16             <Route path='Listings/:studentId' element={<NewForm />}/>
17             <Route path='*' element={<Navigate to='/' />}/>
18         </Routes>
19     </BrowserRouter>
20
21 }
22
23
24 export default App
25
```

VERIFICAR O MAIN.TSX

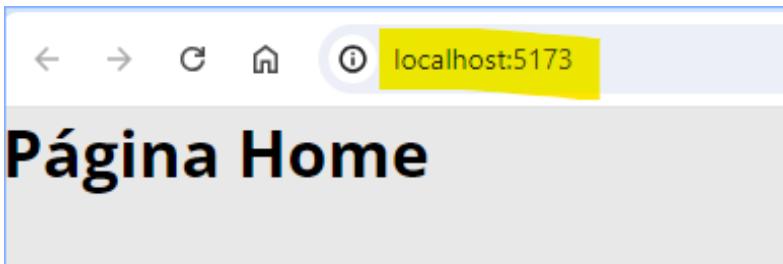


A screenshot of a code editor showing the content of the `main.tsx` file. The code is written in TypeScript and uses React. It imports `index.css`, the `App` component from `./App.tsx`, and the `ReactDOM` module. It then creates a root element using `ReactDOM.createRoot` and renders the `App` component into it.

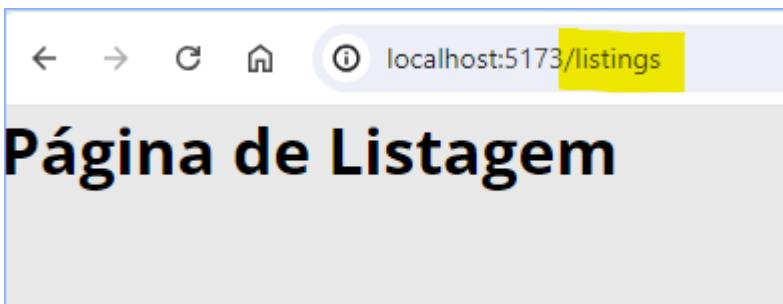
```
main.tsx
1 import './index.css'
2 import App from './App.tsx'
3 import ReactDOM from 'react-dom/client'
4
5 ReactDOM.createRoot(document.getElementById('root')!).render(
6
7   <App />
8
9 )
10
```

VISUALIZAÇÃO WEB

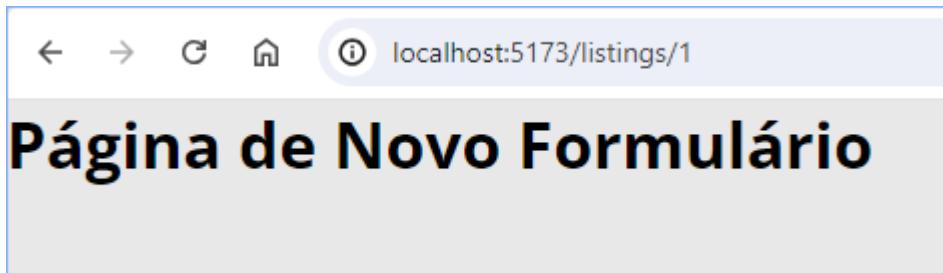
Home



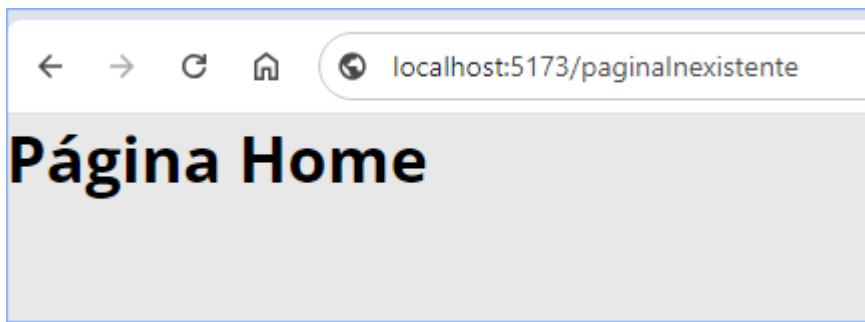
Listing



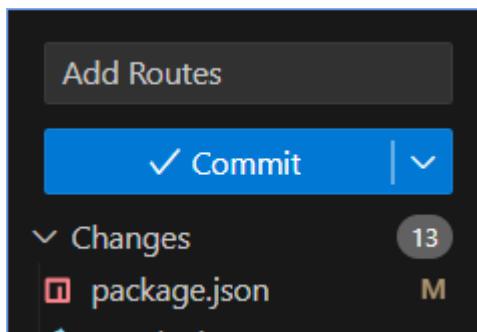
NewForm



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

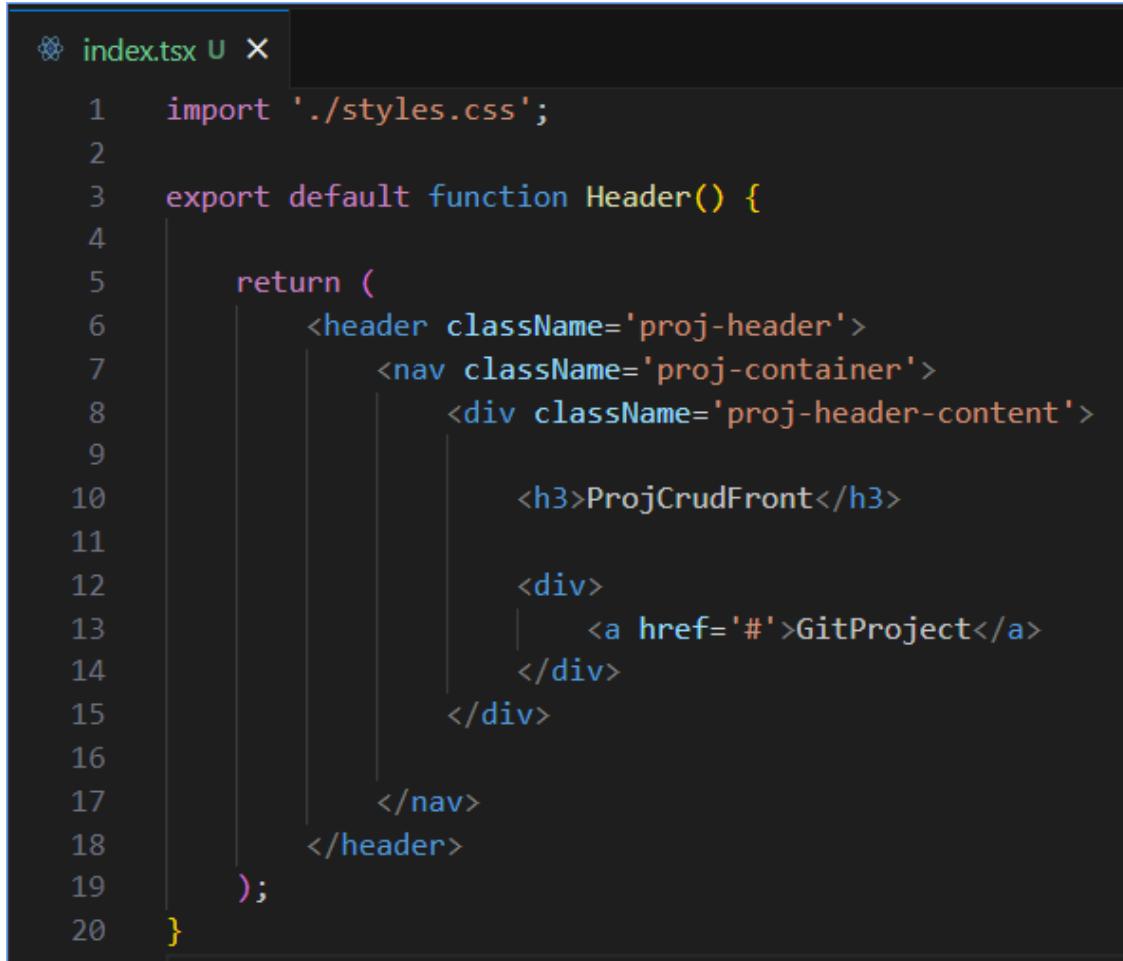


GitHub-5



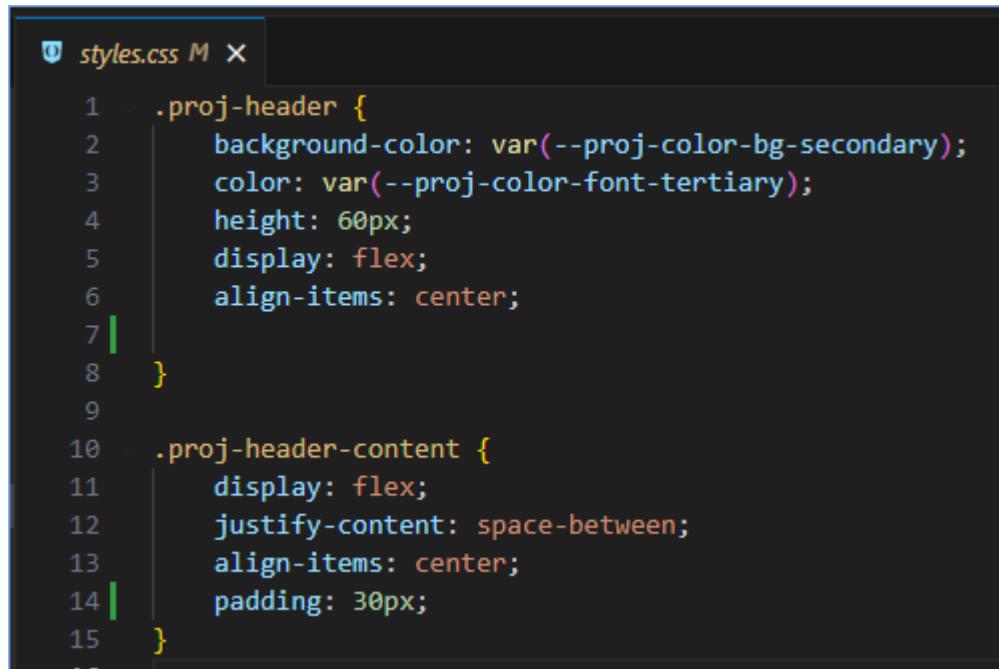
CRIAR O COMPONENTE HEADER

IMPLEMENTAR O INDEX



```
index.tsx
1 import './styles.css';
2
3 export default function Header() {
4
5     return (
6         <header className='proj-header'>
7             <nav className='proj-container'>
8                 <div className='proj-header-content'>
9
10                     <h3>ProjCrudFront</h3>
11
12                     <div>
13                         <a href='#'>GitProject</a>
14                     </div>
15                 </div>
16
17             </nav>
18         </header>
19     );
20 }
```

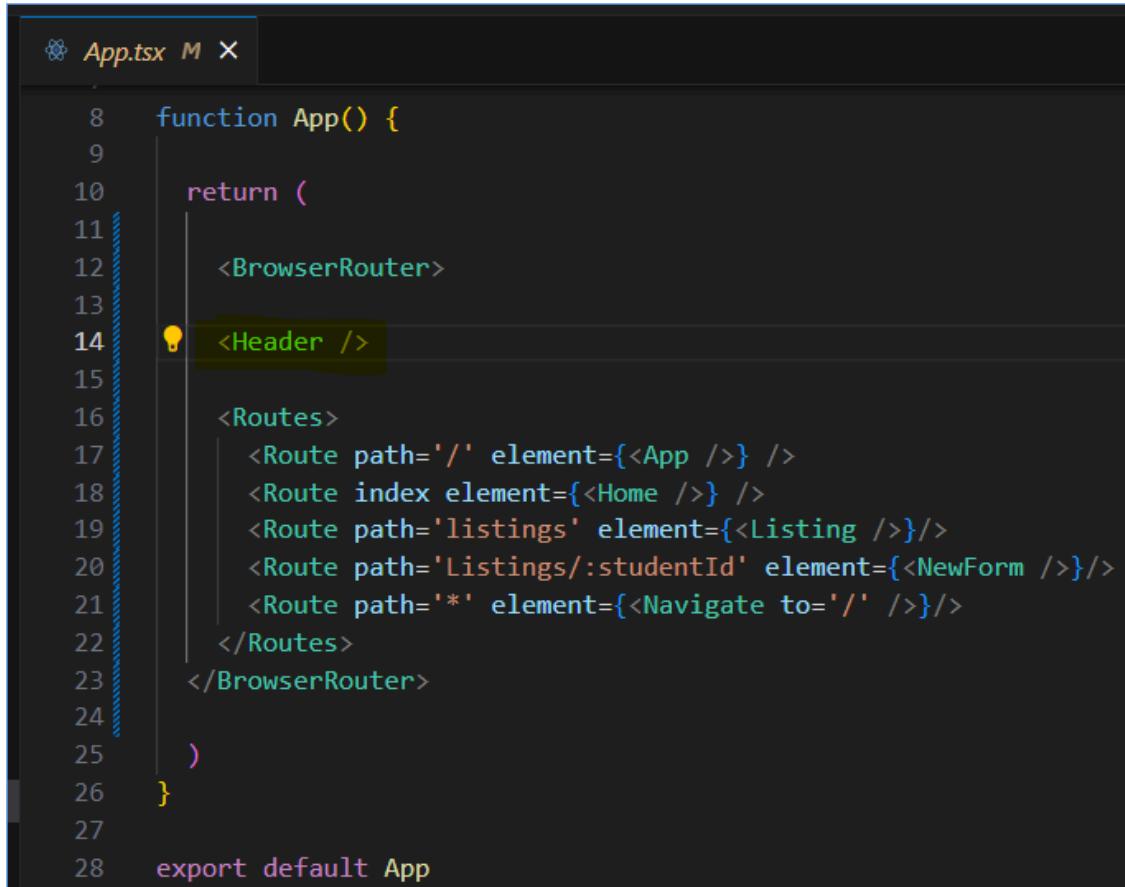
ESTILIZAR



The screenshot shows a code editor window with a dark theme. The file is named "styles.css". The code defines two CSS classes for a header component:

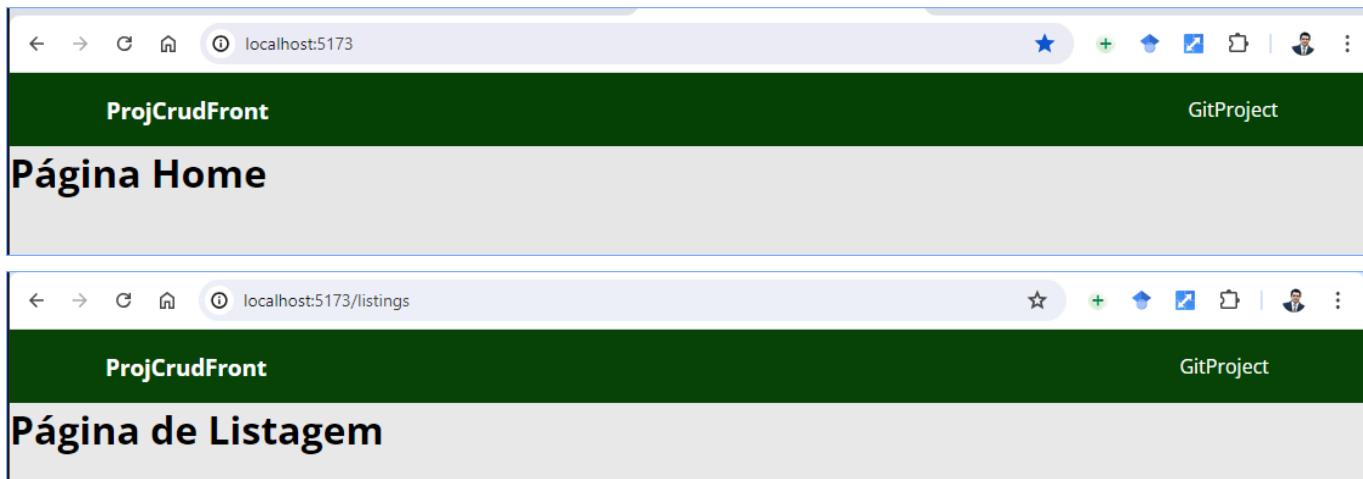
```
1 .proj-header {  
2     background-color: var(--proj-color-bg-secondary);  
3     color: var(--proj-color-font-tertiary);  
4     height: 60px;  
5     display: flex;  
6     align-items: center;  
7 }  
8  
9  
10 .proj-header-content {  
11     display: flex;  
12     justify-content: space-between;  
13     align-items: center;  
14     padding: 30px;  
15 }
```

RENDERIZAR O HEADER NO APP.TSX



```
App.tsx M X
8  function App() {
9
10 return (
11   <BrowserRouter>
12     <Header />
13
14     <Routes>
15       <Route path='/' element={<App />} />
16       <Route index element={<Home />} />
17       <Route path='listings' element={<Listing />}/>
18       <Route path='Listings/:studentId' element={<NewForm />}/>
19       <Route path='*' element={<Navigate to='/' />}/>
20     </Routes>
21   </BrowserRouter>
22 )
23
24 }
25
26
27
28 export default App
```

VISUALIZAÇÃO WEB



localhost:5173

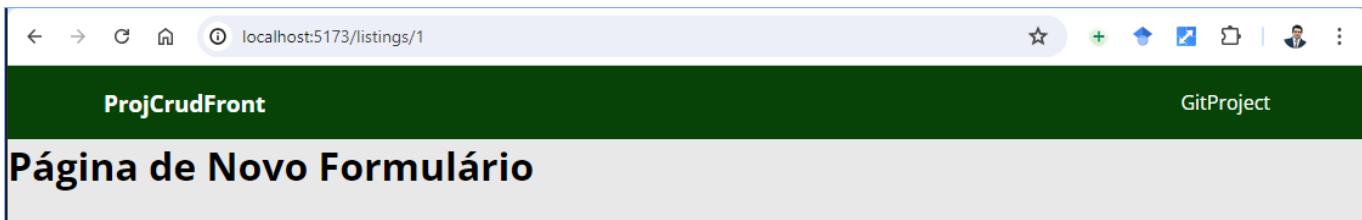
ProjCrudFront GitProject

Página Home

localhost:5173/listings

ProjCrudFront GitProject

Página de Listagem

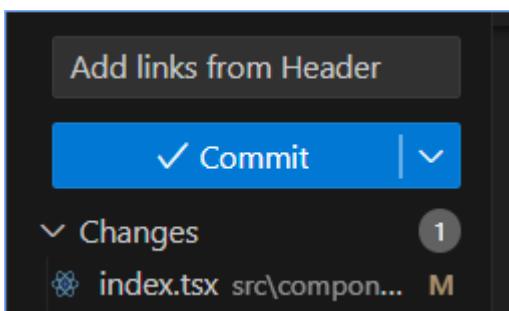


IMPLEMENTAR OS LINKS DO HEADER

```
index.tsx
6     return (
7         <header className='proj-header'>
8             <nav className='proj-container'>
9                 <div className='proj-header-content'>
10                     <Link to="/">
11                         <h3>ProjCrudFront</h3>
12                     </Link>
13
14                     <div>
15                         <a href='https://github.com/auriceliof/pessoal-ProjCrudBackFront'>GitProject</a>
16                     </div>
17                 </div>
```

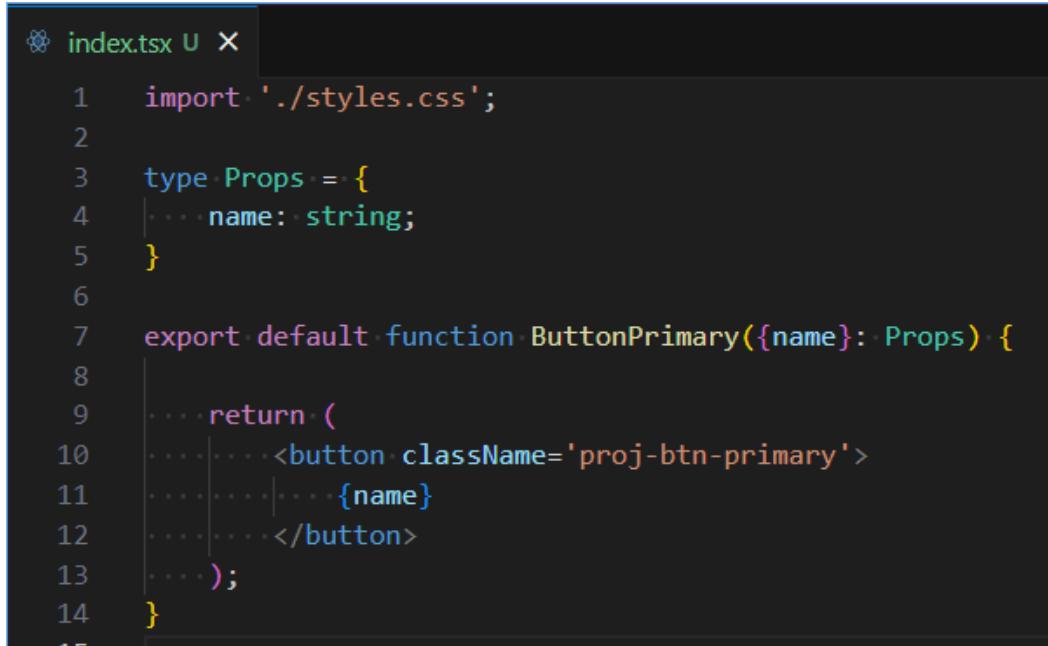
NOTA: Ao clicar no “ProjCrudFront” a página é redirecionada para o home. Ao clicar no “GitProject”, será direcionado para a página do projeto no GitHub.

GitHub-6



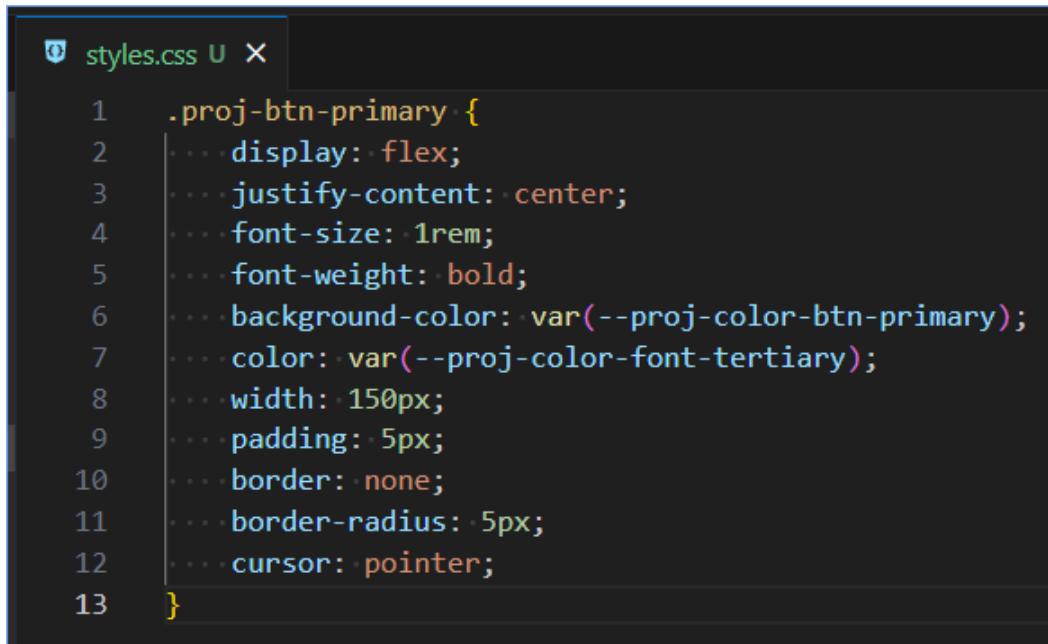
CRIAR O COMPONENTE BUTTON_PRIMARY

IMPLEMENTAR O INDEX



```
index.tsx
1 import './styles.css';
2
3 type Props = {
4   name: string;
5 }
6
7 export default function ButtonPrimary({name}: Props) {
8
9   return (
10     <button className='proj-btn-primary'>
11       {name}
12     </button>
13   );
14 }
```

ESTILIZAR



```
styles.css
1 .proj-btn-primary {
2   display: flex;
3   justify-content: center;
4   font-size: 1rem;
5   font-weight: bold;
6   background-color: var(--proj-color-btn-primary);
7   color: var(--proj-color-font-tertiary);
8   width: 150px;
9   padding: 5px;
10  border: none;
11  border-radius: 5px;
12  cursor: pointer;
13 }
```

CRIAR O COMPONENTE BUTTON_SECONDARY

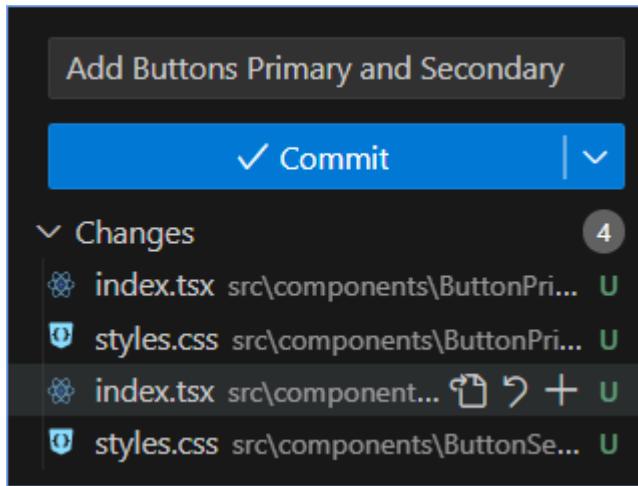
IMPLEMENTAR O INDEX

```
src > components > ButtonSecondary > index.tsx > ...
1  import './styles.css';
2
3  type Props = {
4    name: string;
5  }
6
7  export default function ButtonSecondary({name}: Props) {
8
9    return (
10      
11        {name}
12      
13    );
14  }
15
```

ESTILIZAR

```
src > components > ButtonSecondary > styles.css > ...
1 .proj-btn-secondary {
2   display: flex;
3   justify-content: center;
4   font-size: 1rem;
5   font-weight: bold;
6   background-color: var(--proj-color-btn-secondary);
7   color: var(--proj-color-font-tertiary);
8   width: 150px;
9   padding: 5px;
10  border: none;
11  border-radius: 5px;
12  cursor: pointer;
13}
```

GitHub-7



PÁGINA HOME

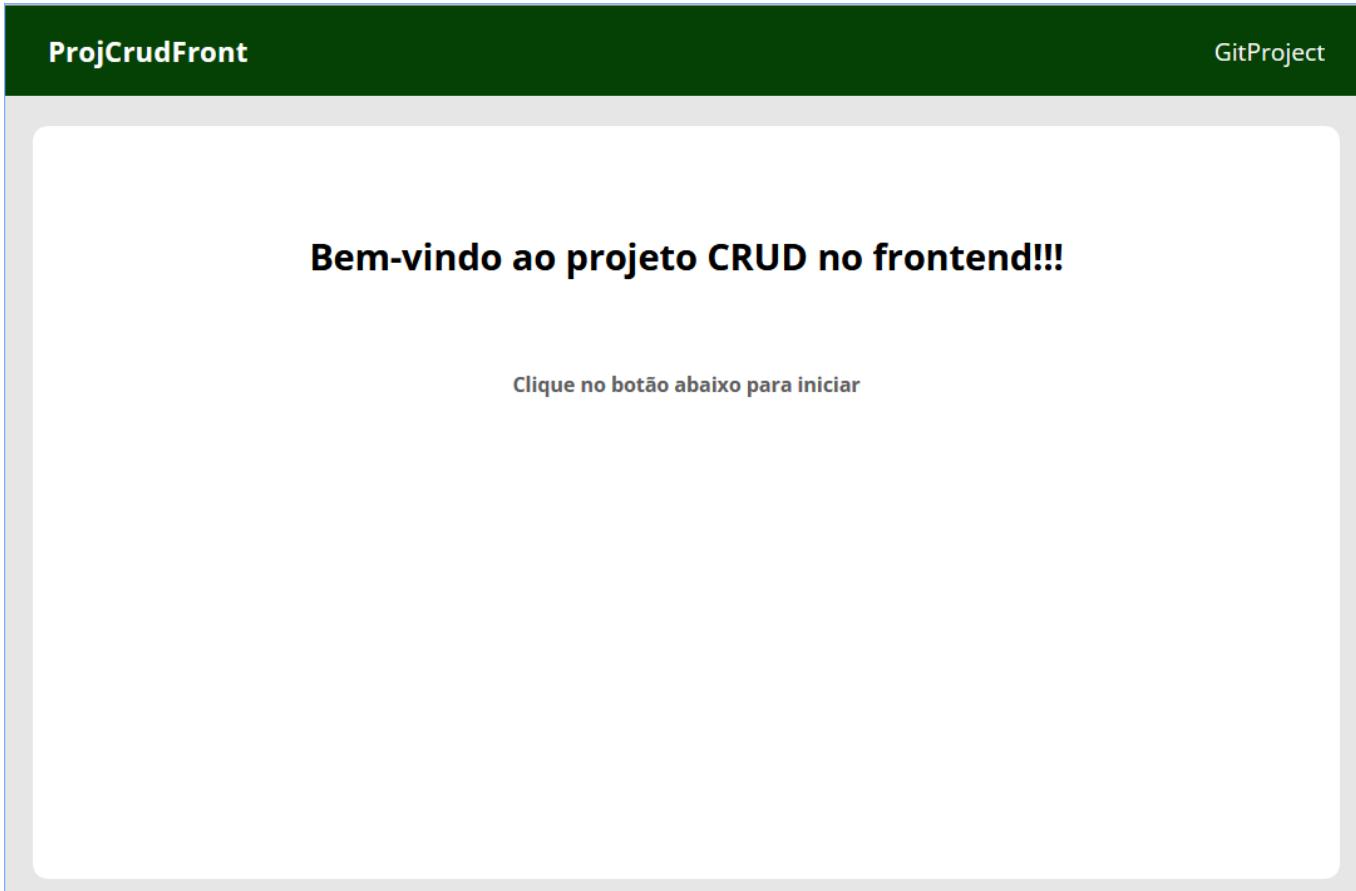
IMPLEMENTAR O INDEX

```
src > pages > Home > index.tsx > ...
1  import './styles.css';
2
3  export default function Home() {
4
5      return (
6          <main>
7              <section className='proj-container'>
8                  <div className='proj-mt20 proj-card'>
9                      <div className="proj-home-content">
10                         <h1>Bem-vindo ao projeto CRUD no frontend!!!</h1>
11                         </div>
12                         <div className="proj-home-content">
13                             <h5>Clique no botão abaixo para iniciar</h5>
14                             </div>
15                     </div>
16                 </section>
17             </main>
18         )
19     }
```

ESTILIZAR

```
src > pages > Home > styles.css > ...
1  .proj-home-content {
2      display: flex;
3      justify-content: center;
4  }
5
6  .proj-home-content h1 {
7      margin-top: 70px;
8  }
9
10 .proj-home-content h5 {
11     margin-top: 60px;
12     color: var(--proj-color-font-primary);
13 }
```

VISUALIZAÇÃO WEB



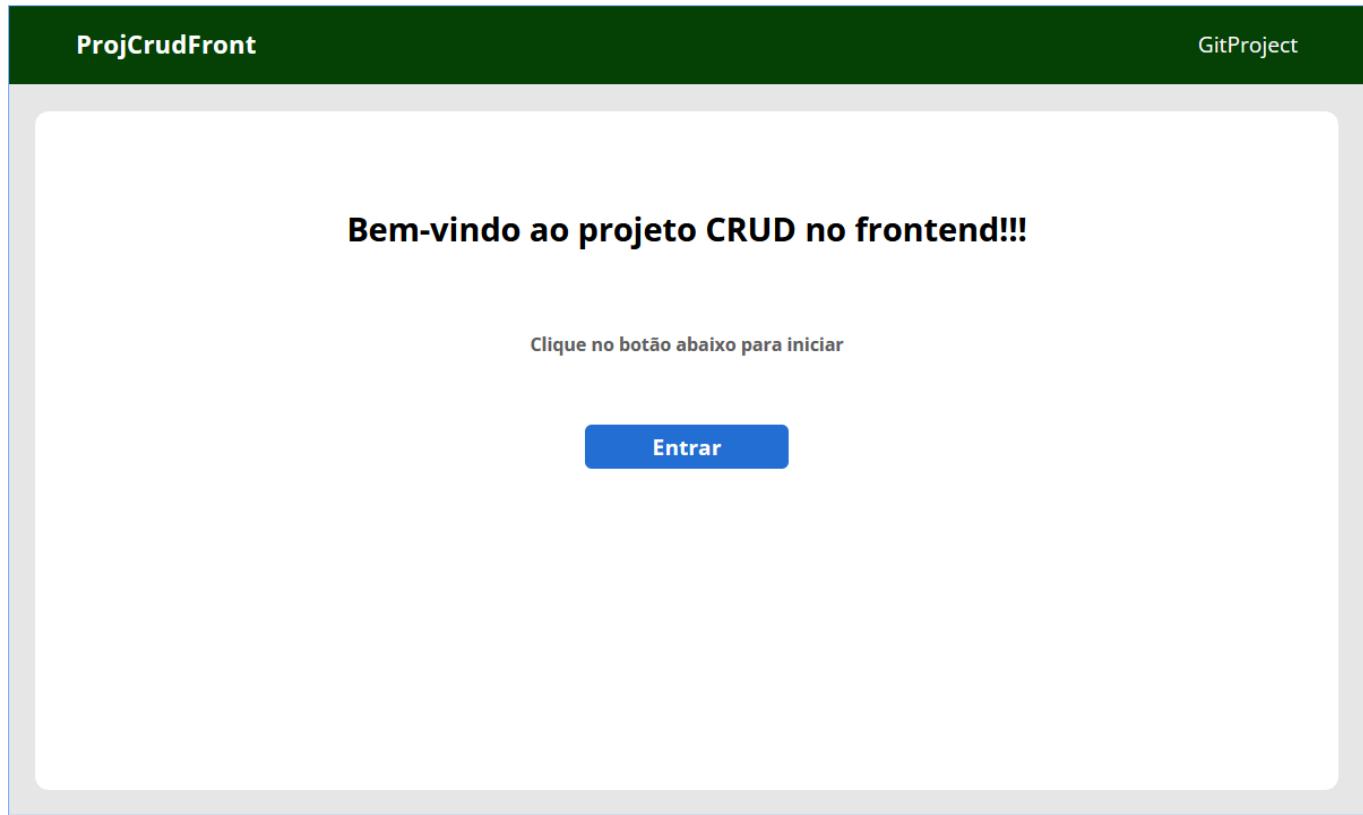
CHAMADO O COMPONENTE BUTTON_PRIMARY

```
src > pages > Home > index.tsx > ...
6
7     return (
8         <main>
9             <section className='proj-container'>
10                <div className='proj-mt20 proj-card'>
11                    <div className="proj-home-content">
12                        <h1>Bem-vindo ao projeto CRUD no frontend!!!</h1>
13                    </div>
14                    <div className="proj-home-content">
15                        <h5>Clique no botão abaixo para iniciar</h5>
16                    </div>
17                    <div className='proj-home-btn'>
18                        <Link to="/listings">
19                            <ButtonPrimary name='Entrar'/>
20                        </Link>
21                    </div>
22                </div>
23            </section>
24        </main>
25    )
26 }
```

ESTILIZAR

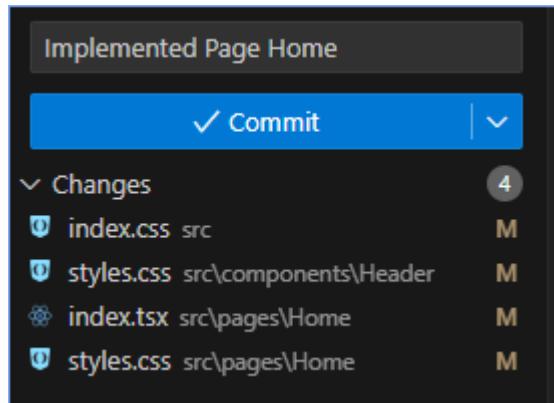
```
src > pages > Home > styles.css > ...
14
15     .proj-home-btn {
16         display: flex;
17         justify-content: center;
18         margin-top: 50px;
19     }
20
```

VISUALIZAÇÃO WEB



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

GitHub-8



PÁGINA LISTING

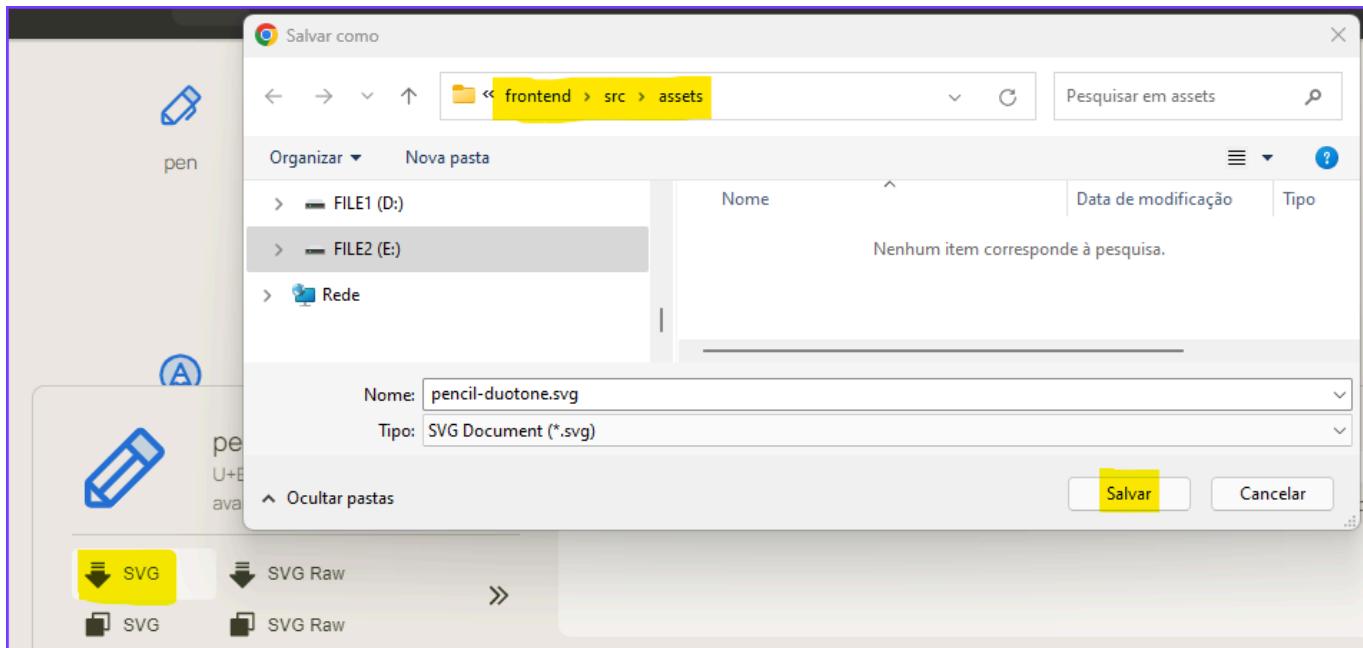
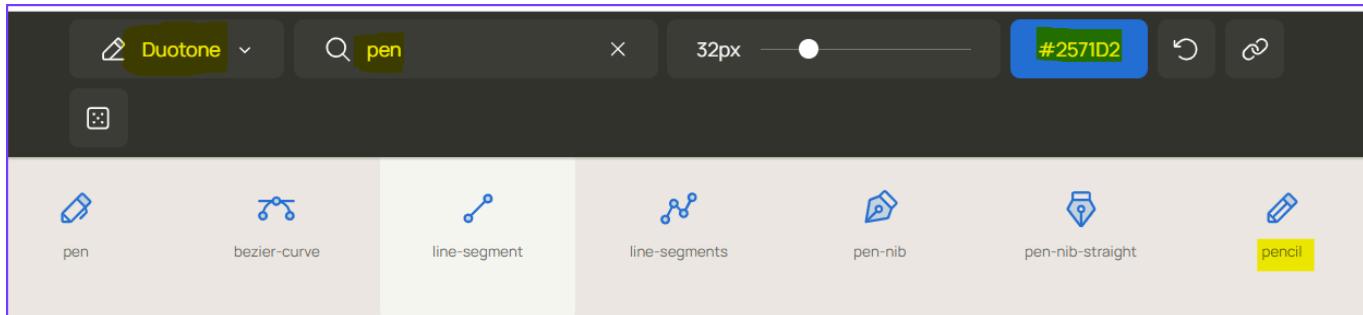
BAIXAR ICONES

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

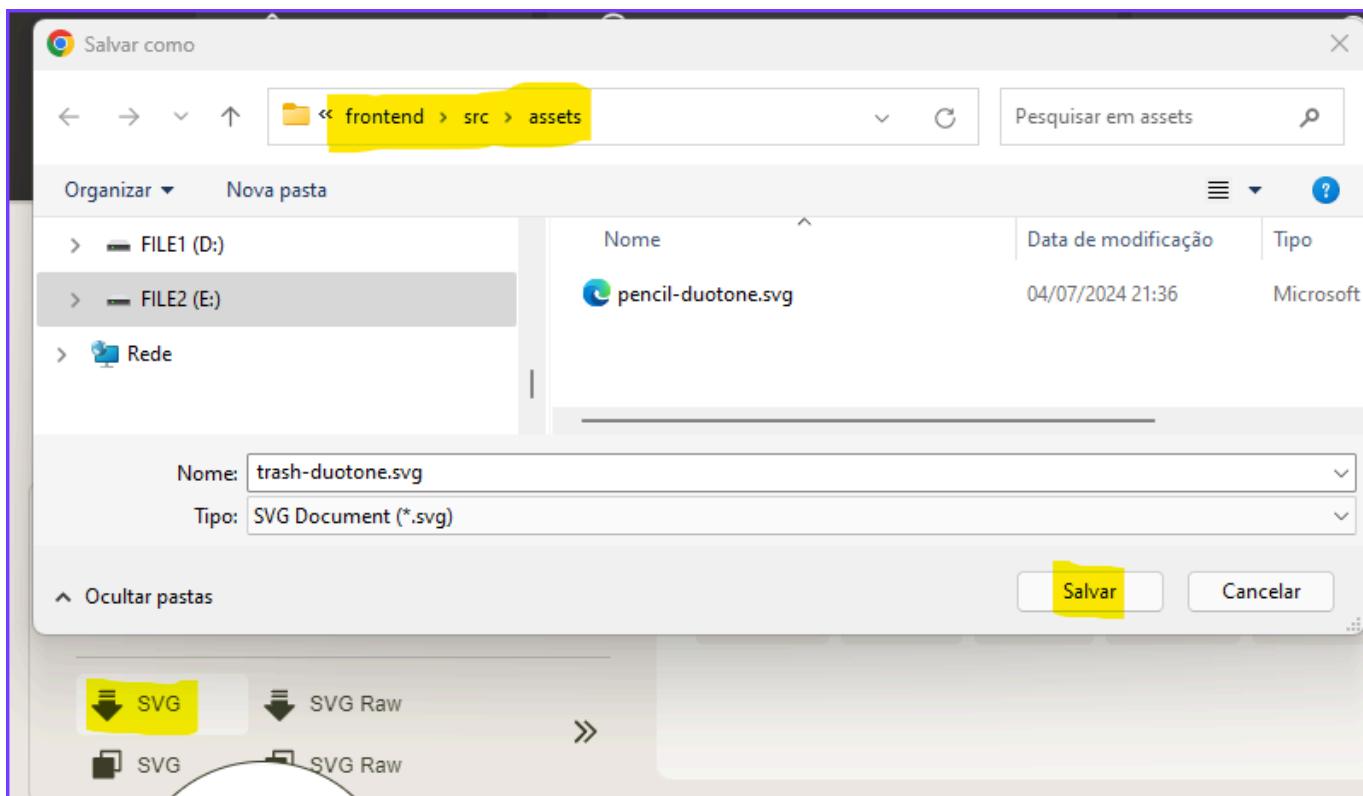
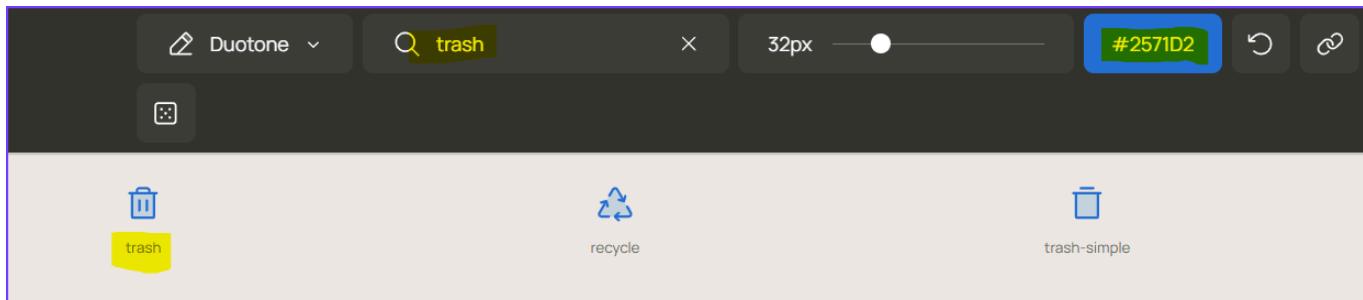
- Link: <https://phosphoricons.com/>

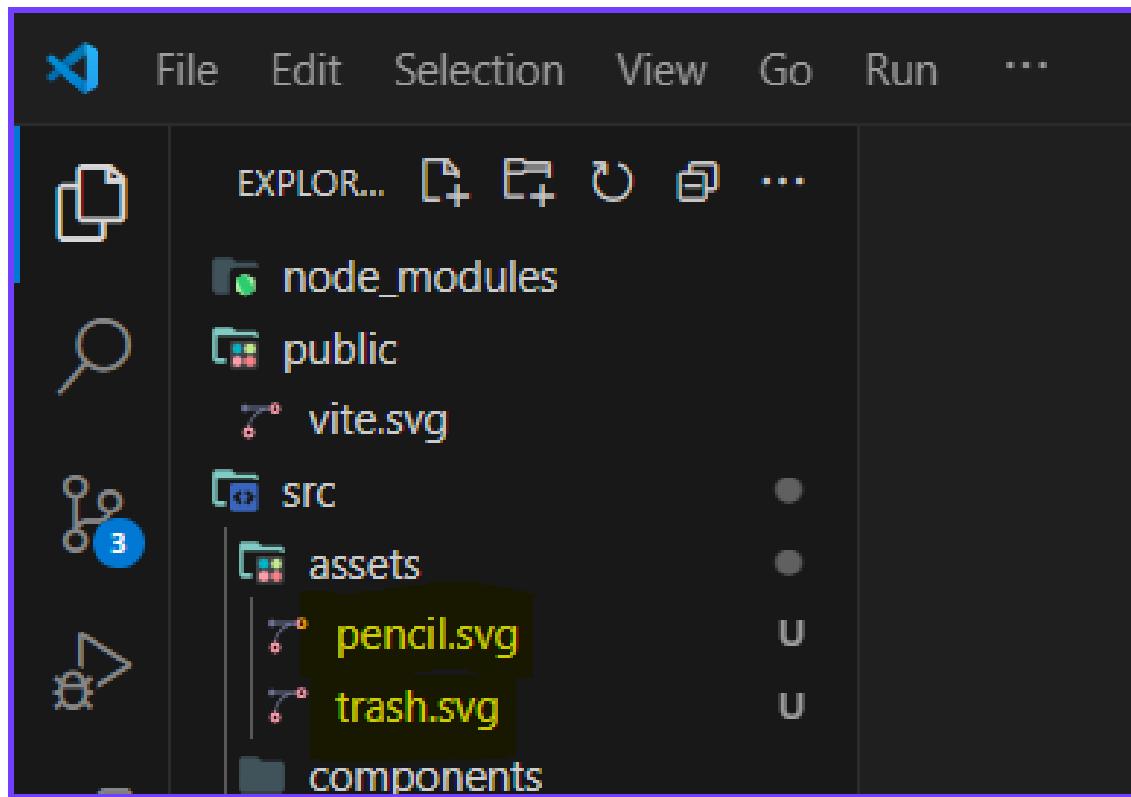
NOTA: Baixar os ícones em formato SVG e salvar no diretório do projeto, em assets.

Lápis

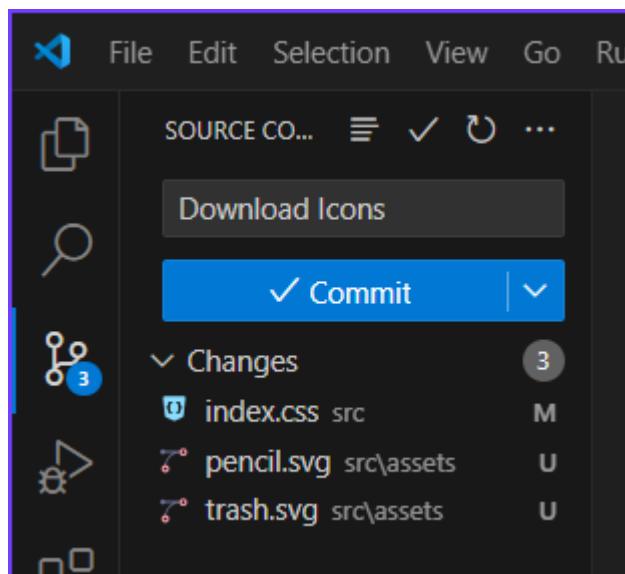


Lixeira





GitHub-9



LAYOUT DA LISTAGEM DE ALUNOS

Implementar o Index

```
src > pages > Listing > index.tsx ...
1  import './styles.css';
2  import editIcon from '../../../../../assets/pencil.svg';
3  import deleteIcon from '../../../../../assets/trash.svg';
4  import ButtonPrimary from '../../../../../components/ButtonPrimary';
5
6  export default function Listing() {
7
8      return (
9          <main>
10             <section id="proj-listing-section" className="proj-container">
11                 <div>
12                     <h2 className="proj-mt20">Listagem de Alunos</h2>
13                 </div>
14
15                 <div className="proj-mt40 proj-mb20">
16                     <div className="proj-listing-btn">
17                         <ButtonPrimary name='Novo' />
18                     </div>
19                 </div>
20
21                 <table className="proj-listing-table proj-mb20 proj-mt20">
22                     <thead>
23                         <tr>
24                             <th className='proj-listing-table-id'>ID</th>
25                             <th>NOME</th>
26                             <th>CPF</th>
27                             <th>NASCIMENTO</th>
28                             <th>RENDAS</th>
29                             <th></th>
30                             <th></th>
31                         </tr>
32                     </thead>
```

```
33 |         <tbody>
34 |             <tr>
35 |                 <td className='proj-listing-table-id'>1</td>
36 |                 <td>Auricelio Freitas</td>
37 |                 <td>123.456.789-00</td>
38 |                 <td>28/08/1982</td>
39 |                 <td>R$ 10999.0</td>
40 |                 <td><img src={editIcon} alt='Editar' /></td>
41 |                 <td><img src={deleteIcon} alt='Deletar' /></td>
42 |             </tr>
43 |             <tr>
44 |                 <td className='proj-listing-table-id'>2</td>
45 |                 <td>Teste1 Sobrenome</td>
46 |                 <td>123.456.789-00</td>
47 |                 <td>08/06/1990</td>
48 |                 <td>R$ 1999.0</td>
49 |                 <td><img src={editIcon} alt='Editar' /></td>
50 |                 <td><img src={deleteIcon} alt='Deletar' /></td>
51 |             </tr>
52 |             <tr>
53 |                 <td className='proj-listing-table-id'>3</td>
54 |                 <td>Teste2 Sobrenome</td>
55 |                 <td>123.456.789-00</td>
56 |                 <td>08/06/1990</td>
57 |                 <td>R$ 12999.0</td>
58 |                 <td><img src={editIcon} alt='Editar' /></td>
59 |                 <td><img src={deleteIcon} alt='Deletar' /></td>
60 |             </tr>
61 |
62 |             <tr>
63 |                 <td className='proj-listing-table-id'>4</td>
64 |                 <td>Teste3 Sobrenome</td>
65 |                 <td>123.456.789-00</td>
66 |                 <td>08/06/1990</td>
67 |                 <td>R$ 8999.0</td>
68 |                 <td><img src={editIcon} alt='Editar' /></td>
69 |                 <td><img src={deleteIcon} alt='Deletar' /></td>
70 |             </tr>
71 |             <tr>
72 |                 <td className='proj-listing-table-id'>5</td>
73 |                 <td>Teste4 Sobrenome</td>
74 |                 <td>123.456.789-00</td>
75 |                 <td>08/06/1990</td>
76 |                 <td>R$ 15999.0</td>
77 |                 <td><img src={editIcon} alt='Editar' /></td>
78 |                 <td><img src={deleteIcon} alt='Deletar' /></td>
79 |             </tr>
80 |         </tbody>
81 |     </table>
82 |     </section>
83 | </main>
84 | }
```

NOTA: Neste momento, criaremos alunos estáticos para testes.

Estilizar

```
src > pages > Listing > styles.css > ...
1  .proj-listing-btn {
2    cursor: pointer;
3    margin: 5px;
4    width: 18%;
5    font-size: 10px;
6  }
7
8  .proj-listing-table {
9    width: 100%;
10   background-color: var(--proj-color-table-primary);
11   border-spacing: 0;
12   border: 1px solid var(--proj-color-table-border);
13   border-radius: 15px 15px 0 0;
14 }
15
16 .proj-listing-table-id {
17   text-align: end;
18 }
19
20 .proj-listing-table thead {
21   height: 55px;
22   font-size: 14px;
23   color: var(--proj-color-font-quaternary);
24 }
25
26 .proj-listing-table tbody {
27   text-align: center;
28   font-size: 12px;
29   color: var(--proj-color-font-primary);
30   background-color: var(--proj-color-card-bg);
31 }
32
33 .proj-listing-table tbody tr {
34   height: 50px;
35   border-top: 1px solid var(--proj-color-card-border);
36 }
37
38 .proj-listing-table tbody img {
39   height: 20px;
40   cursor: pointer;
41 }
42
43 .proj-listing-table tr:nth-of-type(even) {
44   background-color: var(--proj-color-table-tertiary);
45 }
```

Visualização web

The screenshot shows a web application titled "ProjCrudFront" running at "localhost:5173/listings". The page has a dark green header with the title and a "GitProject" button. Below the header, the title "Listagem de Alunos" is displayed. A blue "Novo" button is located above a table. The table has columns: ID, NOME, CPF, NASCIMENTO, and RENDA. It contains five rows of student data with edit and delete icons.

ID	NOME	CPF	NASCIMENTO	RENDAS
1	Auricelio Freitas	123.456.789-00	28/08/1982	R\$ 10999.0
1	Teste1 Sobrenome	123.456.789-00	08/06/1990	R\$ 1999.0
3	Teste2 Sobrenome	123.456.789-00	08/06/1990	R\$ 12999.0
4	Teste3 Sobrenome	123.456.789-00	08/06/1990	R\$ 8999.0
5	Teste4 Sobrenome	123.456.789-00	08/06/1990	R\$ 15999.0

GitHub-10

The screenshot shows a GitHub commit interface. The title is "Implement page Listing Static". The commit message is "✓ Commit". There are 2 changes: "index.tsx" and "styles.css", both marked as modified ("M").

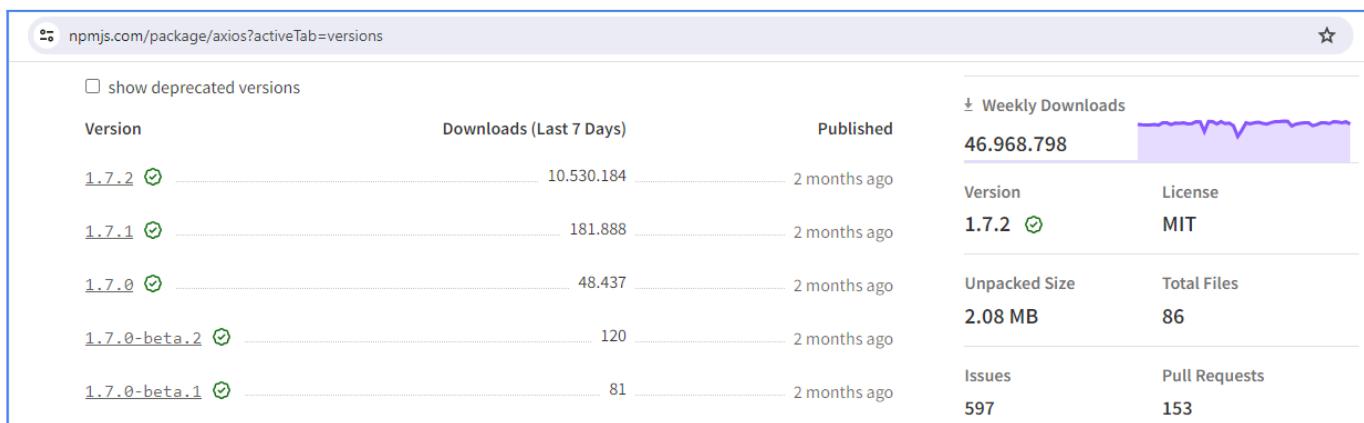
LISTAGEM DINÂMICA

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

```
SAP@DESKTOP-LSDERN6 MINGW64 /c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend (main)
$ yarn add axios@1.7.2
yarn add v1.22.22
[1/4] Resolving packages...
[2/4] Fetching packages...
[3/4] Linking dependencies...
[4/4] Building fresh packages...
success Saved lockfile.
success Saved 9 new dependencies.
info Direct dependencies
└─ axios@1.7.2
info All dependencies
├─ asynckit@0.4.0
└─ axios@1.7.2
├─ combined-stream@1.0.8
├─ delayed-stream@1.0.0
├─ follow-redirects@1.15.6
├─ form-data@4.0.0
├─ mime-db@1.52.0
└─ mime-types@2.1.35
└─ proxy-from-env@1.1.0
Done in 2.98s.
```

Verificar instalação

```
package.json > ...
12  "dependencies": {
13    "@types/react-router-dom": "5.3.3",
14    "axios": "1.7.2",
15    "react": "^18.3.1",
16    "react-dom": "^18.3.1",
17    "react-router-dom": "6.4.1"
18  },
```

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 }
15
```

NOTA: Local onde armazenaremos as variáveis de configuração do Axios

Criar o format

```
src > utils > format.ts > ...
1  // Função para formatar data e hora no formato Brazil "dd/mm/yyyy"
2  export function formatDateBR(dataHora: any) {
3      const data = new Date(dataHora);
4      const ano = data.getFullYear();
5      const mes = String(data.getMonth() + 1).padStart(2, '0');
6      const dia = String(data.getDate() + 1).padStart(2, '0');
7
8      const dataBR = `${dia}/${mes}/${ano}`
9
10     if (dataBR === "31/12/1969") {
11         return ""
12     }
13
14     return dataBR;
15 }
16
```

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 > ...
1  import { AxiosRequestConfig } from "axios";
2  import { BASE_URL } from "../utils/system";
3  import { requestBackend } from "../utils/requests";
4
5  export function findPageRequest(page: number, name: string, size = 12, sort = "id") {
6      const config : AxiosRequestConfig = {
7          method: "GET",
8          baseURL: BASE_URL,
9          url: "/students",
10         params: {
11             page,
12             name,
13             size,
14             sort
15         }
16     }
17
18     return requestBackend(config)
19 }
```

NOTA: Local onde armazenaremos as variáveis de requisições REST.

IMPLEMENTAR O MODELS

Criar o students.ts

```
src > models >  students.ts > ...
1  export type StudentDTO = {
2      id: number;
3      name: string;
4      cpf: string;
5      birthDate: Date;
6      income: number;
7  };
```

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 > ...
1  import './styles.css';
2  import editIcon from '../../assets/pencil.svg';
3  import deleteIcon from '../../assets/trash.svg';
4  import ButtonPrimary from '../../components/ButtonPrimary';
5  import { useEffect, useState } from 'react';
6  import * as studentService from '../../services/student-service';
7  import { StudentDTO } from '../../models/students';
8  import { formatDateBR } from '../../utils/format';
9
10 type QueryParams = {
11     page: number;
12     name: string;
13 }
```

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

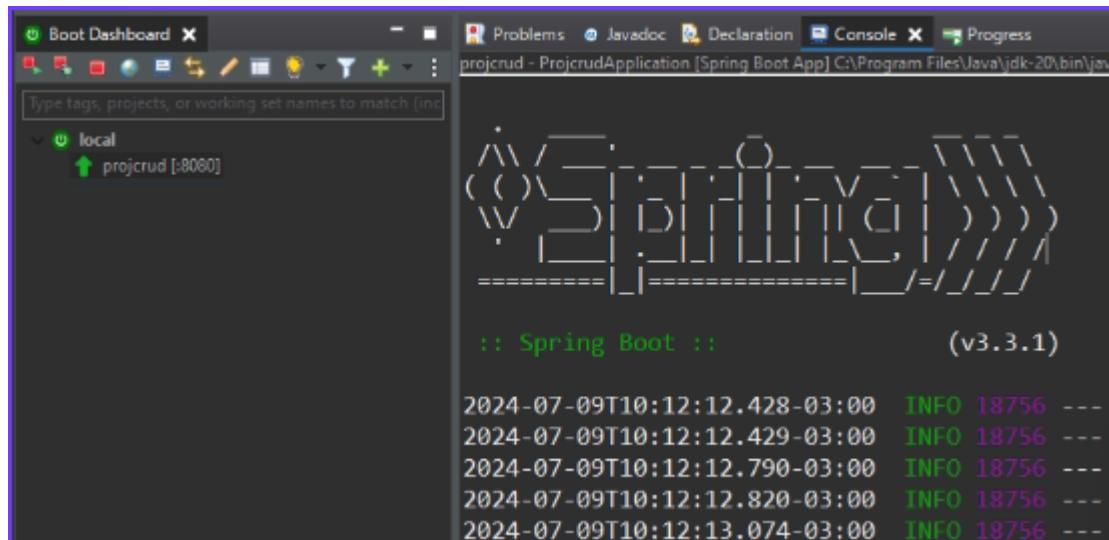
```
src > pages > Listing > ✎ index.tsx > ...
25   useEffect(() => {
26     studentService.findPageRequest(queryParams.page, queryParams.name)
27       .then(response => {
28         setStudents(response.data.content)
29       })
30   }, [])
31
```

Implementar a renderização da tabela dinamicamente

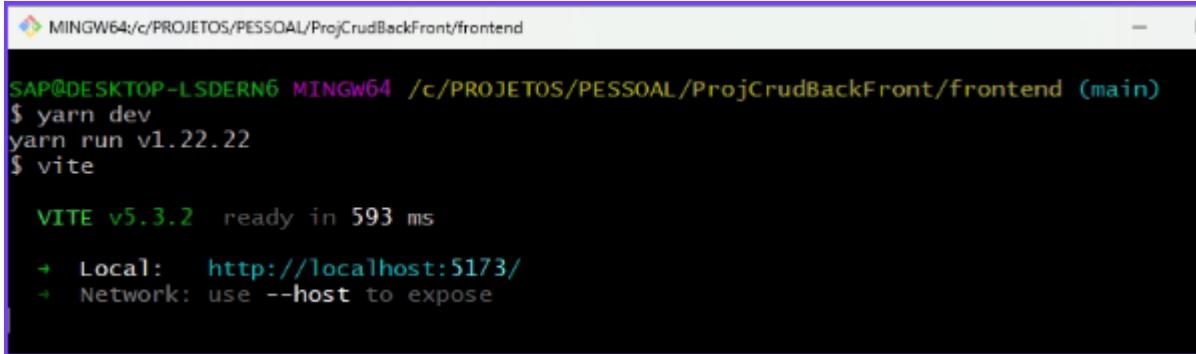
```
src > pages > Listing > index.tsx > ...
56     </thead>
57     <tbody>
58     {
59         students.map(student => (
60             <tr key={student.id}>
61                 <td className='proj-listing-table-id'>{student.id}</td>
62                 <td>{student.name}</td>
63                 <td>{student.cpf}</td>
64                 <td>{formatDateBR(student.birthDate)}</td>
65                 <td>R$ {student.income}</td>
66                 <td><img src={editIcon} alt='Editar' /></td>
67                 <td><img src={deleteIcon} alt='Deletar' /></td>
68             </tr>
69         )));
70     }
71     </tbody>
72     </table>
73     </section>
74     </main>
75   )
76 }
77 }
```

VISUALIZAÇÃO WEB

Iniciar o Backend

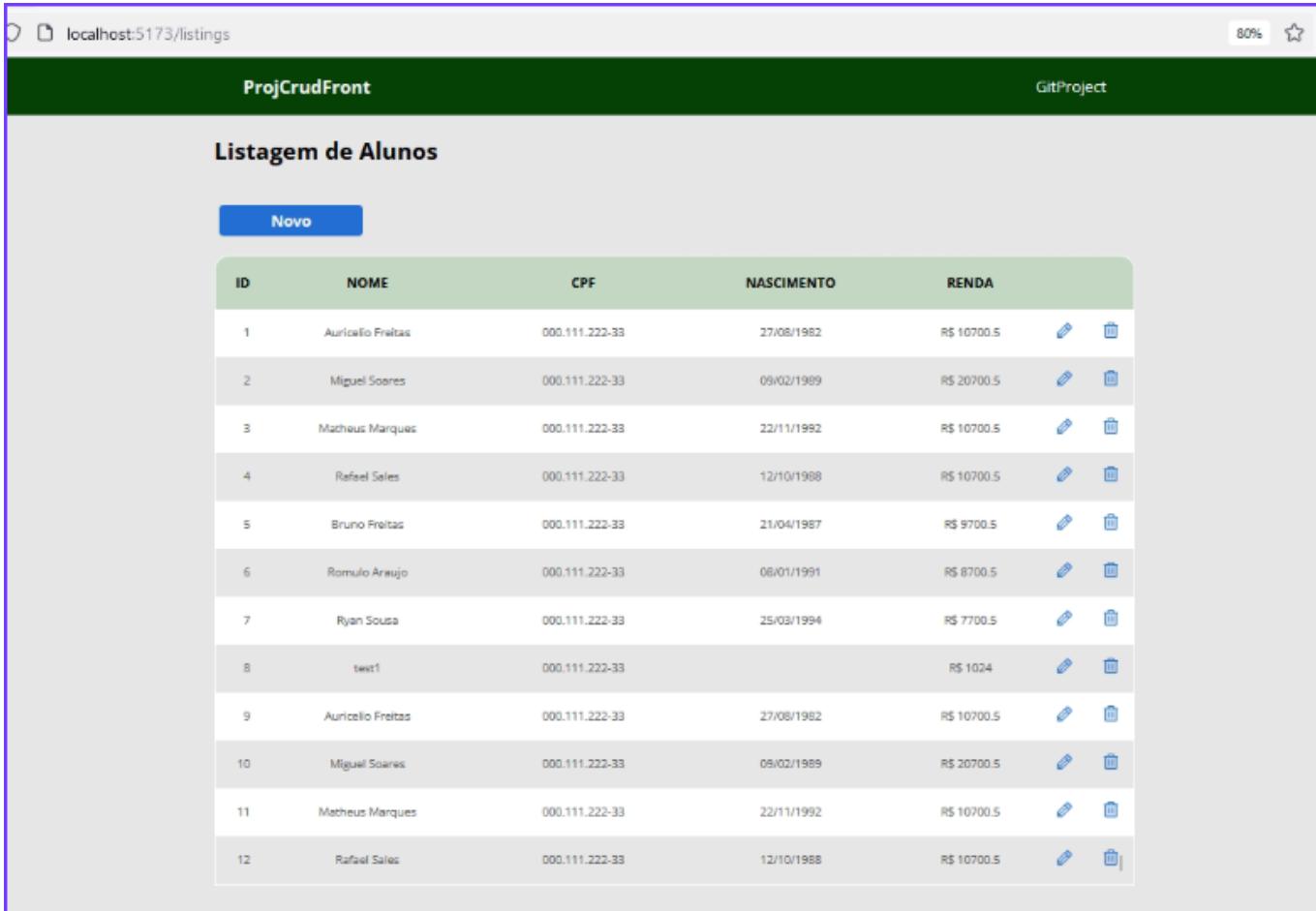


Iniciar o Frontend



```
MINGW64:/c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend
$ yarn dev
yarn run v1.22.22
$ vite
VITE v5.3.2 ready in 593 ms
→ Local: http://localhost:5173/
→ Network: use --host to expose
```

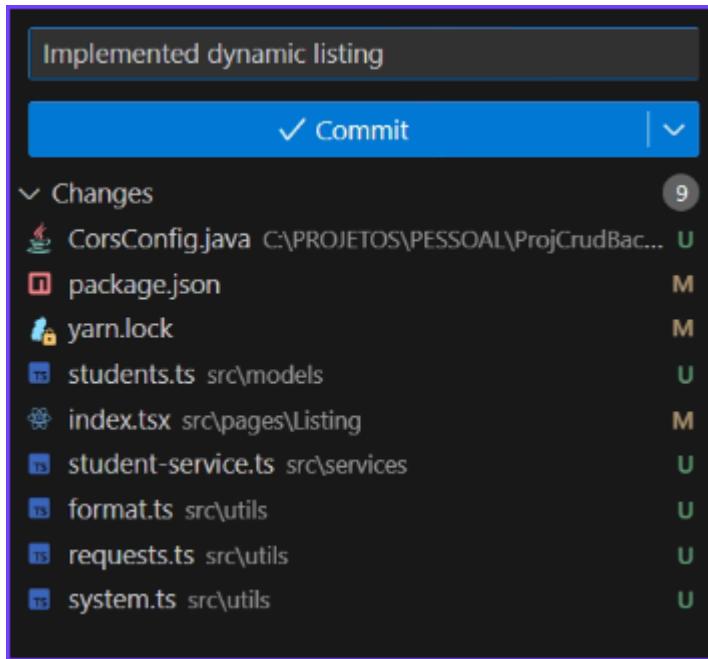
Visualizar



The screenshot shows a web browser window with the URL `localhost:5173/listings`. The title bar says "ProjCrudFront". The main content area has a header "Listagem de Alunos" with a "Novo" button. Below is a table with 12 rows of student data:

ID	NOME	CPF	NASCIMENTO	RENDA	editar	deletar
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	text1	000.111.222-33		R\$ 1024		
9	Auricelio Freitas	000.111.222-33	27/08/1982	R\$ 10700.5		
10	Miguel Soares	000.111.222-33	09/02/1989	R\$ 20700.5		
11	Matheus Marques	000.111.222-33	22/11/1992	R\$ 10700.5		
12	Rafael Sales	000.111.222-33	12/10/1988	R\$ 10700.5		

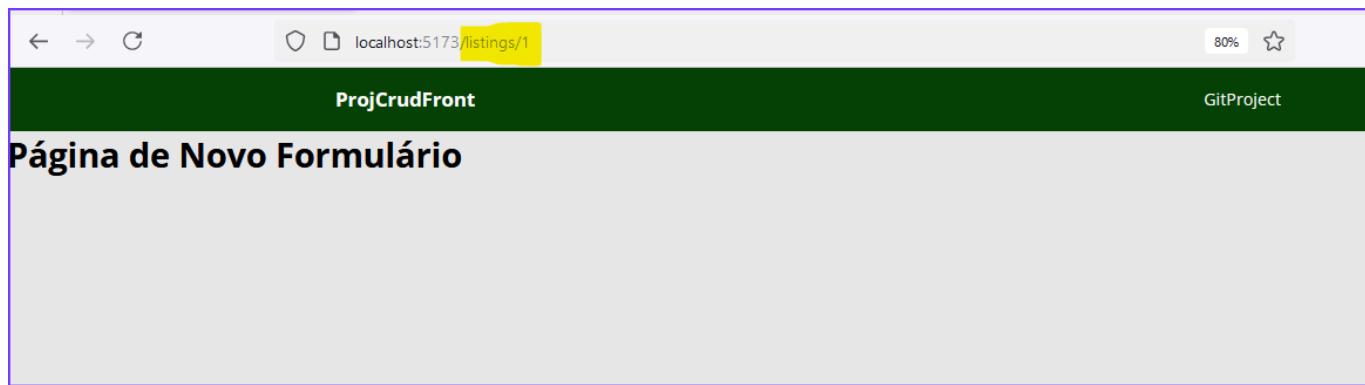
GitHub-11



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 > ...
1  import './styles.css';
2  import { Link } from 'react-router-dom';
3  import ButtonSecondary from '../../../../../components/ButtonSecondary';
4  import ButtonPrimary from '../../../../../components/ButtonPrimary';
5
6  export default function NewForm() {
7
8      return (
9          <main>
10             <section id="proj-form-section" className="proj-container">
11                 <div className="proj-form-container">
12                     <form className="proj-card proj-form">
13                         <h2 className="proj-mb20">Dados do Aluno</h2>
14                         <div className="proj-form-controls-container">
15                             <div>
16                                 <h5>Nome</h5>
17                                 <input className="proj-form-control" type="text" placeholder='Nome' />
18                             </div>
19                             <div>
20                                 <h5>Cpf</h5>
21                                 <input className="proj-form-control" type="text" placeholder='Cpf' />
22                             </div>
23                             <div>
24                                 <h5>Data de Nascimento</h5>
25                                 <input className="proj-form-control" type="text" placeholder='Data de Nascimento' />
26                             </div>
27                             <div>
28                                 <h5>Renda</h5>
29                                 <input className="proj-form-control" type="text" placeholder='Renda' />
30                             </div>
31
32                         </div>
33
34                         <div className="proj-form-buttons">
35                             <Link to="/listings">
36                                 <ButtonSecondary name='Cancelar' />
37                             </Link>
38                             <ButtonPrimary name='Salvar' />
39                         </div>
40
41                     </form>
42                 </div>
43             </section>
44         </main>
45     );
46 }
```

Styles

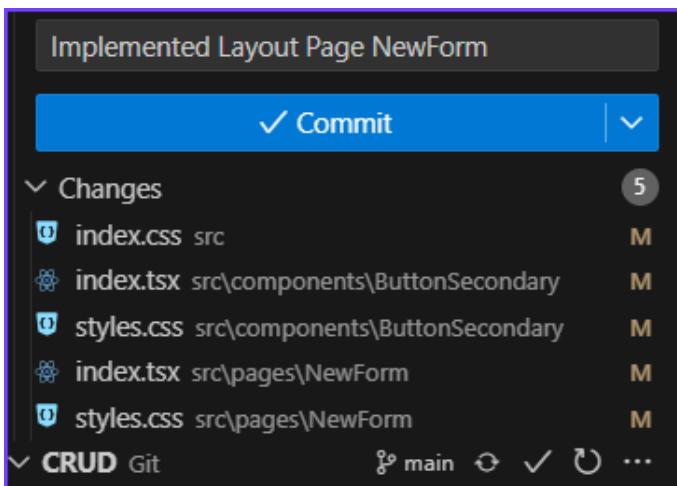
```
src > pages > NewForm > styles.css > ...
1  #proj-form-section {
2    padding: 20px 0;
3  }
4
5  .proj-form-container {
6    width: 100%;
7    max-width: 500px;
8    margin: 0 auto;
9  }
10
11 .proj-form {
12   display: flex;
13   flex-direction: column;
14   align-items: center;
15   padding: 40px 20px;
16 }
17
18 .proj-form h2 {
19   font-size: 20px;
20   font-weight: bolder;
21   color: var(--proj-color-font-quinternary);
22   text-transform: uppercase;
23   font-weight: bold;
24   text-align: center;
25 }
26
27 .proj-form h5 {
28   color: var(--proj-color-font-primary);
29   margin-left: 10px;
30 }
31
32 .proj-form-controls-container {
33   width: 100%;
34   display: grid;
35   grid-gap: 20px;
36   margin: 10px 0;
37   padding: 0 22px;
38 }
```

```
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

A screenshot of a web browser window. The address bar shows 'localhost:5173/listings/1'. The page title is 'ProjCrudFront' and the top right corner says 'GitProject'. A modal dialog box is centered on the page with a white background and a gray border. The title of the dialog is 'DADOS DO ALUNO'. Inside the dialog, there are four input fields: 'Nome' (Name), 'Cpf', 'Data de Nascimento' (Date of Birth), and 'Renda' (Income). Below the input fields are two buttons: a red 'Cancelar' (Cancel) button on the left and a blue 'Salvar' (Save) button on the right.

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 > ...
1  export default function FormInput(props: any) {
2
3    const { validation, invalid, ...inputProps } = props;
4
5    return (
6      <input { ...inputProps } data-invalid={invalid} />
7    );
8 }
```

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 > ...
1  export function update(inputs: any, name: string, newValue: any) {
2    return{ ...inputs, [name]: { ...inputs[name], value: newValue } };
3  }
4
```

IMPLEMENTAR A PAGE NEW_FORM

Implementar os Imports

```
src > pages > NewForm > index.tsx > ...
1  import './styles.css';
2  import { Link } from 'react-router-dom';
3  import ButtonSecondary from '../../components/ButtonSecondary';
4  import ButtonPrimary from '../../components/ButtonPrimary';
5  import FormInput from '../../components/FormInput';
6  import { useState } from 'react';
7  import * as forms from '../../utils/forms';
```

Implementar o formData com useState

```
src > pages > NewForm > index.tsx > ...
28
29     const [formData, setFormData] = useState<any>({
30         name: {
31             value: "",
32             id: "name",
33             name: "name",
34             type: "text",
35             placeholder: "Nome",
36         },
37         cpf: {
38             value: "",
39             id: "cpf",
40             name: "cpf",
41             type: "text",
42             placeholder: "CPF",
43         },
44         birthDate: {
45             value: "",
46             id: "birthDate",
47             name: "birthDate",
48             type: "date",
49             placeholder: "Data de Nascimento",
50         },
51         income: {
52             value: [],
53             id: "income",
54             name: "income",
55             type: "double",
56             placeholder: "Salário",
57         },
58     });
59
```

Implementar o FormInput

```
src > pages > NewForm > index.tsx > ...
46     return (
47         <main>
48             <section id="proj-form-section" className="proj-container">
49                 <div className="proj-form-container">
50                     <form className="proj-card proj-form">
51                         <h2 className="proj-mb20">Dados do Aluno</h2>
52                         <div className="proj-form-controls-container">
53                             <div>
54                                 <h5>Nome</h5>
55                                 <FormInput
56                                     { ...formData.name }
57                                     className="proj-form-control"
58                                     onChange={handleInputChange}
59                                 />
60                             </div>
61                             <div>
62                                 <h5>Cpf</h5>
63                                 <FormInput
64                                     { ...formData.cpf }
65                                     className="proj-form-control"
66                                     onChange={handleInputChange}
67                                 />
68                             </div>
69                             <div>
70                                 <h5>Data de Nascimento</h5>
71                                 <FormInput
72                                     { ...formData.birthDate }
73                                     className="proj-form-control"
74                                     onChange={handleInputChange}
75                                 />
76                             </div>
77                             <div>
78                                 <h5>Renda</h5>
79                                 <FormInput
80                                     { ...formData.income }
81                                     className="proj-form-control"
82                                     onChange={handleInputChange}
83                                 />
84                             </div>
85                         </div>
86                     </form>

```

Implementar a função handleInputChange

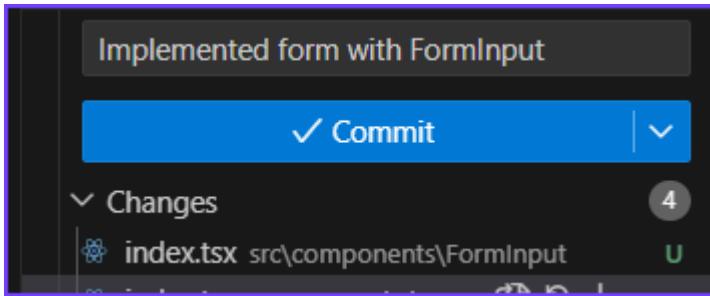
```
50
51     function handleInputChange(event: any) {
52         setFormData(forms.update(formData, event.target.name, event.target.value));
53     }

```

Visualização web

The screenshot shows a web application interface titled "ProjCrudFront" at the top left and "GitProject" at the top right. A modal window titled "DADOS DO ALUNO" (Student Data) is displayed in the center. The form contains four input fields: "Nome" (Name), "Cpf" (CPF), "Data de Nascimento" (Date of Birth), and "Renda" (Income). Below the inputs are two buttons: "Cancelar" (Cancel) in red and "Salvar" (Save) in blue.

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”.

The image consists of two screenshots of a web application interface. Both screenshots have a header bar with a shield icon, a search bar containing 'localhost:5173/listings', and a zoom level of 80%.

Screenshot 1: Listing Page (localhost:5173/listings)

This screenshot shows a table titled "Listagem de Alunos". The columns are labeled ID, NOME, CPF, NASCIMENTO, and RENDA. There is one row of data: ID 1, Name Auricelio Freitas, CPF 000.111.222-33, Birthdate 27/08/1982, and Income R\$ 10700.5. To the right of the table are edit and delete icons.

ID	NOME	CPF	NASCIMENTO	RENDAS
1	Auricelio Freitas	000.111.222-33	27/08/1982	R\$ 10700.5

Screenshot 2: Create Form (localhost:5173/listings/create)

This screenshot shows a modal dialog titled "DADOS DO ALUNO". It contains four input fields: "Nome" (Name), "Cpf" (CPF), "Data de Nascimento" (Birthdate), and "Renda" (Income). Below the inputs are two buttons: "Cancelar" (Cancel) and "Salvar" (Save).

DADOS DO ALUNO

Nome

Cpf

Data de Nascimento
 dd / mm / aaaa

Renda

Cancelar **Salvar**

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 }
```

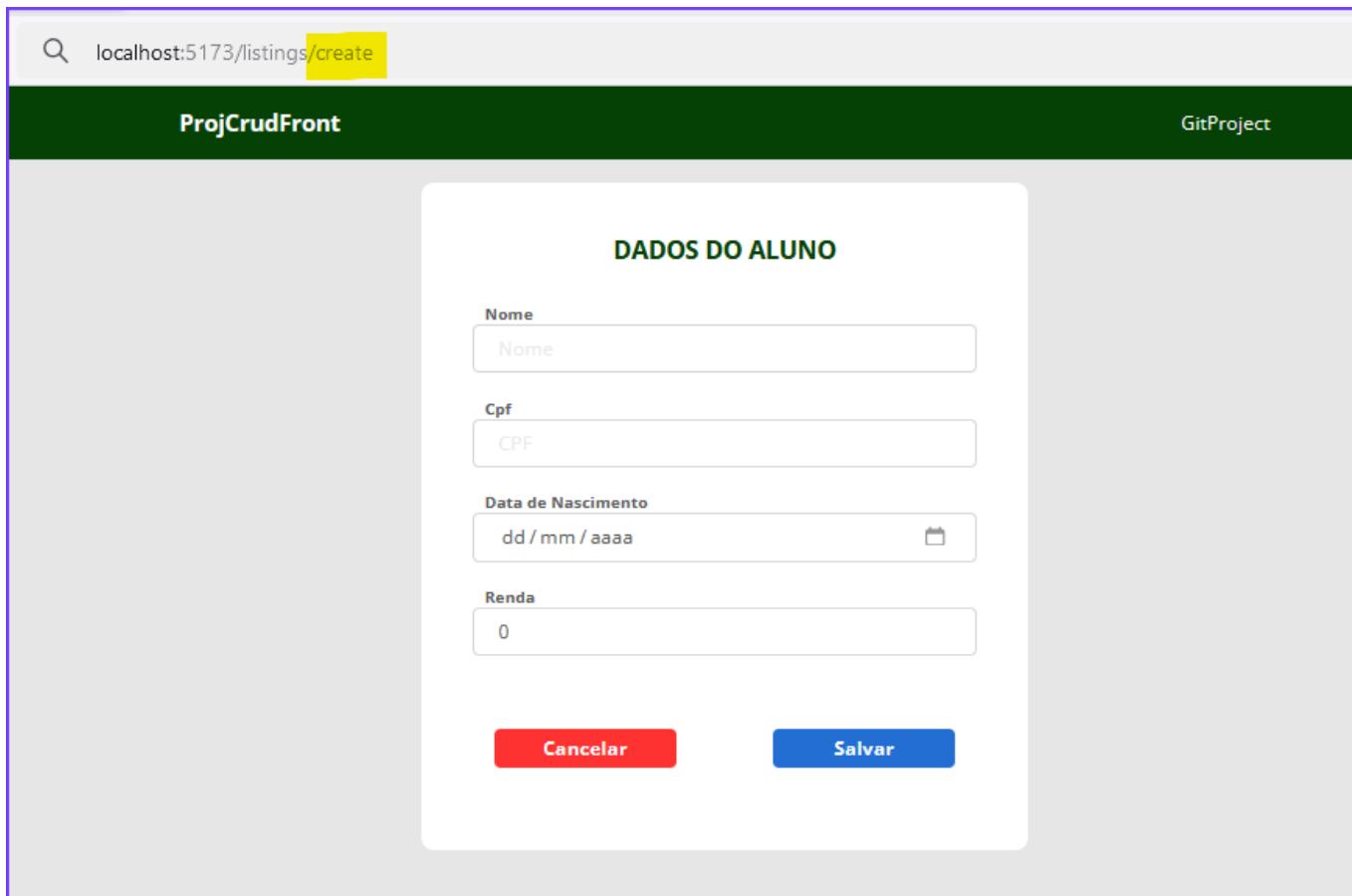
Implementar a função handleNewProduct

```
src > pages > Listing > index.tsx > ...
35 |     function handleNewProduct() {
36 |         navigate("/listings/create")
37 |     };
38 | 
```

Visualização web

The screenshot shows a web application interface. At the top, there is a header bar with the URL "localhost:5173/listings/create". Below the header, the title "ProjCrudFront" is displayed, along with a "GitProject" link. The main content area is titled "Listagem de Alunos". A blue button labeled "Novo" is visible. Below it is a table with the following data:

ID	NOME	CPF	NASCIMENTO	RENDA	ACTIONS
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	



IMPLEMENTAR A ESTRATÉGIA PARA O FORMULÁRIO DE EDIÇÃO ALUNOS

Implementar o `findById` no `student-service` para requisição no backend

```
src > services > student-service.ts > ...
21 | export function findById(id: number) {
22 |   return requestBackend( { url: `/students/${id}` } );
23 |
24 | }
```

Implementar useParams no NewForm

```
src > pages > NewForm > index.tsx > ...
  9  export default function NewForm() {
10
11    const params = useParams();
12
13    const isEditing = params.studentId !== 'create';
14
```

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 > ...
  1  import './styles.css';
  2  import { Link, useParams } from 'react-router-dom';
  3  import ButtonSecondary from '../../components/ButtonSecondary';
  4  import ButtonPrimary from '../../components/ButtonPrimary';
  5  import * as studentService from '../../services/student-service';
  6
```

```
src > pages > NewForm > index.tsx > ...
15
14    useEffect(() => {
15      if (isEditing) {
16        studentService.findById(Number(params.studentId))
17          .then(response => {
18            console.log(response.data);
19          })
20      }
21    }, []);
22
```

Visualização web

The screenshot shows a web browser window with a green header bar labeled "ProjCrudFront". Below it is a modal dialog titled "DADOS DO ALUNO" (Student Data). The form contains four input fields: "Nome" (Name) with placeholder "Nome", "Cpf" (CPF) with placeholder "CPF", "Data de Nascimento" (Birth Date) with placeholder "dd / mm /aaaa" and a calendar icon, and "Renda" (Income) with placeholder "0". At the bottom are two buttons: "Cancelar" (Cancel) in red and "Salvar" (Save) in blue. To the right of the browser is a developer tools window titled "Console". It shows the command "[vite] connecting..." followed by "[vite] connected.". A message from the React DevTools indicates to "Download the React DevTools for a better development experience: <https://reactis.org/link/react-devtools>". Below this, a JSON object is displayed:

```
Object { id: 1, name: "Auricelio Freitas", cpf: "000.111.222-33", birthDate: "1982-08-28", income: 10700.5 }
  birthDate: "1982-08-28"
  cpf: "000.111.222-33"
  id: 1
  income: 10700.5
  name: "Auricelio Freitas"
  <prototype>: Object { ... }
```

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

GitHub-14

The screenshot shows a GitHub commit history for the repository "ProjCrudBackFr...". The commit message is "Diff create and editing". Below the message is a large blue button with a checkmark and the text "Commit". Under the commit message, there is a section titled "Changes" with a count of 2. It lists two files: "index.tsx" and "student-service.ts", both marked with a "M" indicating they were modified. At the bottom of the commit history, there is a summary: "2 changes, 1 addition & 1 deletion" and a link to "View diff".

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 > ...
4
5  export function updateAll(inputs: any, newValue: any) {
6    const newInputs: any = {};
7    for (var name in inputs) {
8      newInputs[name] = { ...inputs[name], value: newValue[name] };
9    }
10
11   return newInputs;
12 }
```

CHAMAR O UPDATE_ALL NO USE_EFFECT DO NEW_FORM

```
src > pages > NewForm > index.tsx > ...
15
16  useEffect(() => {
17    if (isEditing) {
18      studentService.findById(Number(params.studentId))
19        .then(response => {
20          console.log(forms.updateAll(formData, response.data));
21        })
22    }
23  }, [])
24
```

VISUALIZAÇÃO WEB

```

Object { name: {...}, cpf: {...}, birthDate: {...}, income: {...} }
  ↘ birthDate: Object { value: "1982-08-28", id: "birthDate", name: "birthDate", ... }
    id: "birthDate"
    name: "birthDate"
    placeholder: "Data de Nascimento"
    type: "date"
    value: "1982-08-28"
  ↗ <prototype>: Object { ... }
  ↘ cpf: Object { value: "000.111.222-33", id: "cpf", name: "cpf", ... }
    id: "cpf"
    name: "cpf"
    placeholder: "CPF"
    type: "text"
    value: "000.111.222-33"
  ↗ <prototype>: Object { ... }
  ↘ income: Object { value: 10700.5, id: "income", name: "income", ... }
    id: "income"
    name: "income"
    placeholder: "Salário"
    type: "double"
    value: 10700.5
  ↗ <prototype>: Object { ... }
  ↘ name: Object { value: "Auricelio Freitas", id: "name", name: "name", ... }
    id: "name"
    name: "name"
    placeholder: "Nome"
    type: "text"
    value: "Auricelio Freitas"
  ↗ <prototype>: Object { ... }
  ↗ <prototype>: Object { ... }

```

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

```

src > pages > NewForm > index.tsx > ...
15
16     useEffect(() => {
17       if (isEditing) {
18         studentService.findById(Number(params.studentId))
19           .then(response => {
20             const newFormData = forms.updateAll(formData, response.data);
21             setFormData(newFormData);
22           })
23       }
24     }, [])
25

```

VISUALIZAÇÃO WEB

The image displays two screenshots of a web application interface, likely a CRUD (Create, Read, Update, Delete) system for managing student data. Both screenshots show a modal dialog box titled "DADOS DO ALUNO" (Student Data) with fields for Nome (Name), Cpf (CPF), Data de Nascimento (Date of Birth), and Renda (Income). Each screenshot shows a different student record.

Screenshot 1 (Top):

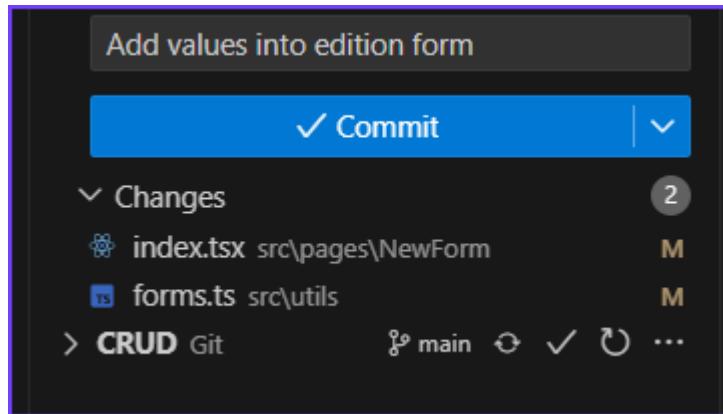
- Nome: Auricelio Freitas
- Cpf: 000.111.222-33
- Data de Nascimento: 28 / 08 / 1982
- Renda: 10700.5

Screenshot 2 (Bottom):

- Nome: Miguel Soares
- Cpf: 000.111.222-33
- Data de Nascimento: 10 / 02 / 1989
- Renda: 20700.5

Both screenshots include "Cancelar" (Cancel) and "Salvar" (Save) buttons at the bottom of the modal.

GitHub-15



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.

ID	NOME	CPF	NASCIMENTO	RENDAS		
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		

IMPLEMENTAR O ONCLICK DO BOTÃO EDITAR, NO LISTING

```
src > pages > Listing > index.tsx > ...
64   <tbody>
65     {
66       students.map(student => (
67         <tr key={student.id}>
68           <td className='proj-listing-table-id'{student.id}</td>
69           <td>{student.name}</td>
70           <td>{student.cpf}</td>
71           <td>{formatDateBR(student.birthDate)}</td>
72           <td>R$ {student.income}</td>
73           <td><img src={editIcon} alt='Editar' onClick={() => {handleUpdate(student.id)}}/></td>
74           <td><img src={deleteIcon} alt='Deletar' /></td>
75         </tr>
76       )
77     )
78   </tbody>
```

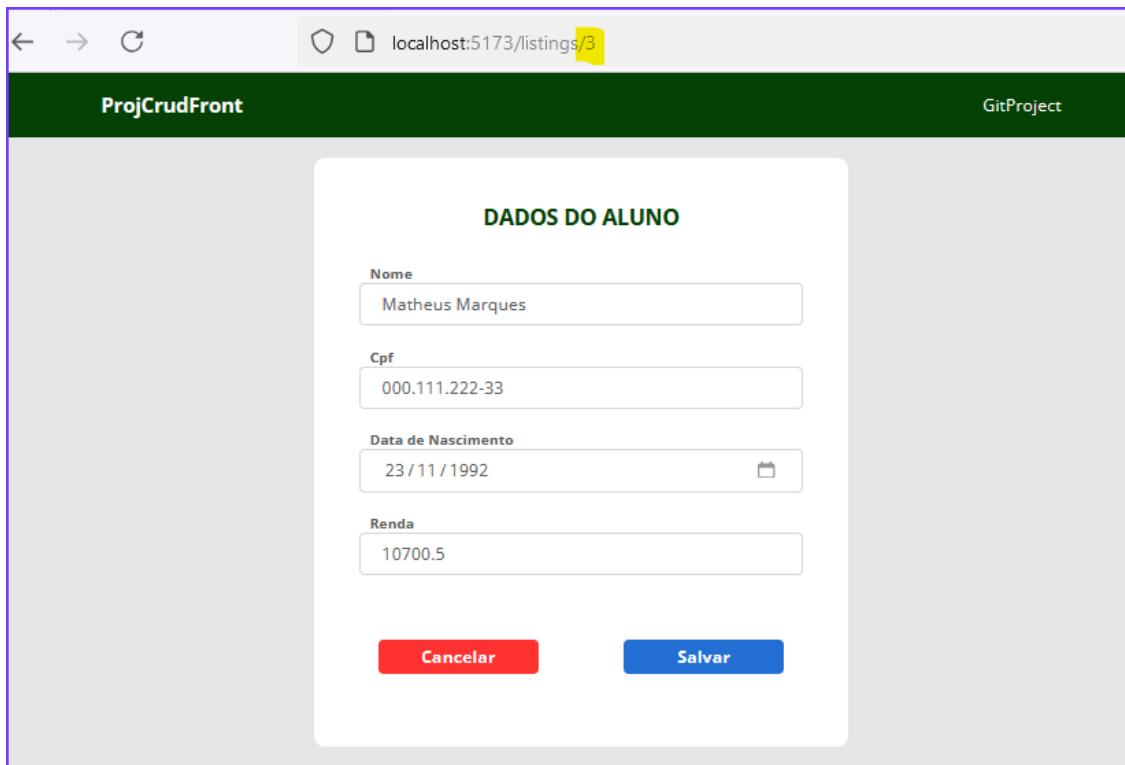
IMPLEMENTAR A FUNÇÃO PARA O HANDLEUPDATE

```
src > pages > Listing > index.tsx > ...
39 |     function handleUpdate(studentId: number) {
40 |         navigate(` /listings/${studentId}`);
41 |
42 |     }
```

VISUALIZAÇÃO WEB

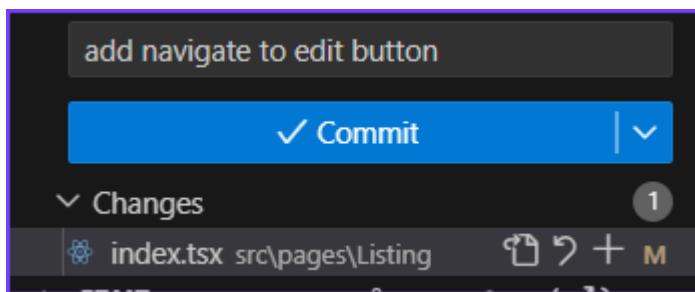
The screenshot shows a web application running at `localhost:5173/listings`. The title bar says "ProjCrudFront" and "GitProject". The main content area is titled "Listagem de Alunos" and features a "Novo" button. Below it is a table with columns: ID, NOME, CPF, NASCIMENTO, and RENDA. The table contains four rows of student data. The third row, for Matheus Marques, has its edit icon highlighted with a yellow box.

ID	NOME	CPF	NASCIMENTO	RENDAS
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



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

The screenshot shows the Postman interface for a PUT request to `localhost:8080/students/1`. The request method is selected in the top left, and the URL is in the top center. The "Body" tab is active, showing a JSON payload:

```
1 {  
2   "name": "Estudante atualizado",  
3   "cpf": "123.456.789-00",  
4   "birthDate": "1994-07-20T10:30:00Z",  
5   "income": 6500.0  
6 }
```

Below the body, the "Test Results" section shows a successful response with status code 200 OK, 613 ms duration, and 355 B size. The response body is identical to the request body:

```
1 {  
2   "id": 1,  
3   "name": "Estudante atualizado",  
4   "cpf": "123.456.789-00",  
5   "birthDate": "1994-07-20",  
6   "income": 6500.0  
7 }
```

Visualização web

ID	NOME	CPF	NASCIMENTO	RENDAS		
1	Estudante atualizado	123.456.789-00	19/07/1994	R\$ 6500		
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		

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  }
22
```

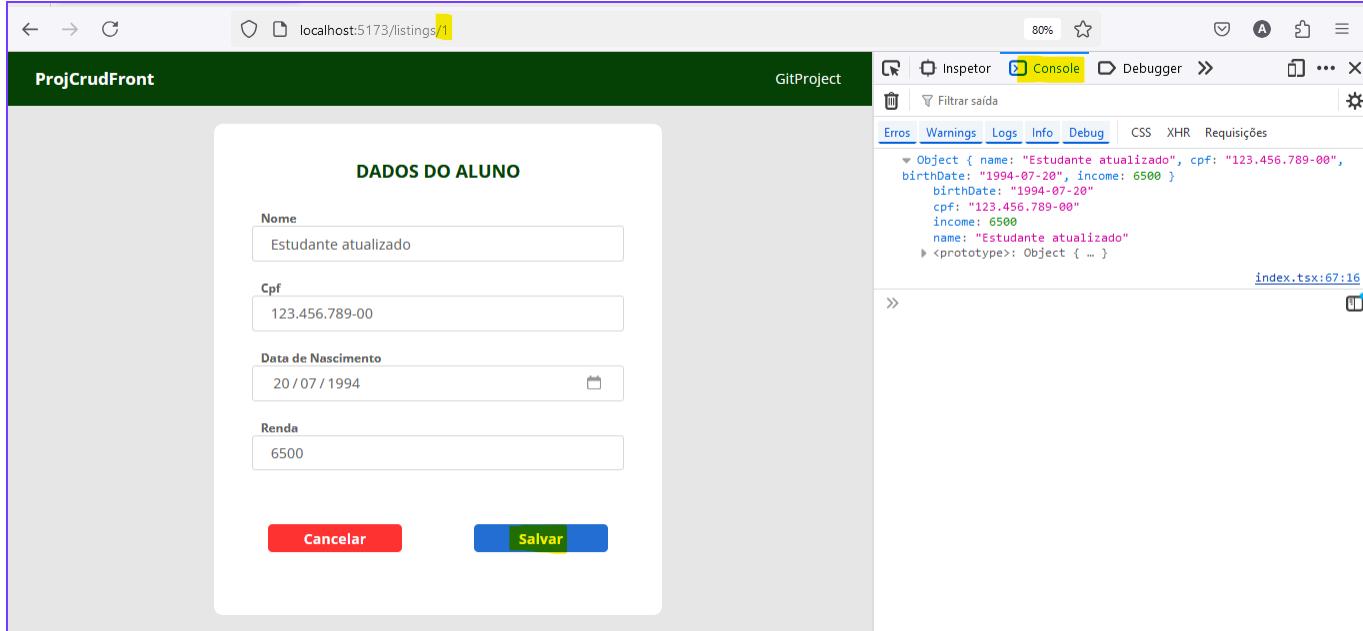
Implementar o handleSubmit, no NewForm

```
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                 <div className="proj-form-container">
71                     <form className="proj-card proj-form" onSubmit={handleSubmit}>
72                         <h2 className="proj-mb20">Dados do Aluno</h2>
73                         <div className="proj-form-controls-container">
```

Implementar um teste

```
src > pages > NewForm > index.tsx > ...
62
63     function handleSubmit(event: any) {
64         event.preventDefault();
65
66         const requestBody = forms.toValues(formData);
67         console.log(requestBody);
68     }
69
```

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 > ...
1  import './styles.css';
2  import { Link, useNavigate, useParams } from 'react-router-dom';
3  import ButtonSecondary from '../../components/ButtonSecondary';
4  import ButtonPrimary from '../../components/ButtonPrimary';
5  import FormInput from '../../components/FormInput';
6  import { useEffect, useState } from 'react';
7  import * as forms from '../../utils/forms';
8  import * as studentService from '../../services/student-service';
9
10 export default function NewForm() {
11
12     const params = useParams();
13
14     const navigate = useNavigate();
15
16     const isEditing = params.studentId !== 'create';
```

Implementar o updateRequest, no NewForm

```
src > pages > NewForm > index.tsx > ...
62
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         studentService.updateRequest(requestBody)
72             .then(() => {
73                 navigate("/listings");
74             });
75     }
76 }
```

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

Visualização web

DADOS DO ALUNO

Nome
Estudante atualizado e EDITADO

Cpf
123.456.789-00

Data de Nascimento
20 / 07 / 1994

Renda
6500

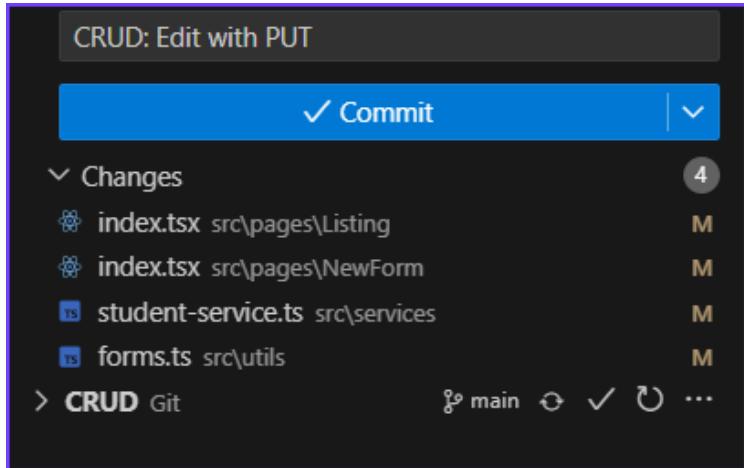
Cancelar **Salvar**

Listagem de Alunos

Novo

ID	NOME	CPF	NASCIMENTO	RENDAS	
1	Estudante atualizado e EDITADO	123.456.789-00	19/07/1994	R\$ 6500	
2	Miguel Soares	000.111.222-33	09/02/1989	R\$ 20700.5	

GitHub-17



INSERIR COM O POST

Testar resposta no postman

The screenshot shows a Postman request for a POST method to "localhost:8080/students". The "Body" tab is selected, showing a JSON payload:

```
1 {  
2   "name": "Novo estudante",  
3   "cpf": "123.456.789-00",  
4   "birthDate": "1982-08-28T10:30:00Z",  
5   "income": 1400.0  
6 }
```

The response section shows a green "201 Created" status, 473 ms duration, and 405 B size. The response body is displayed in "Pretty" format:

```
1 {  
2   "id": 9,  
3   "name": "Novo estudante",  
4   "cpf": "123.456.789-00",  
5   "birthDate": "1982-08-28",  
6   "income": 1400.0  
7 }
```

Visualização web

ID	NOME	CPF	NASCIMENTO	RENDAS
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

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

The screenshot shows a web application interface. At the top, there's a header bar with the title "ProjCrudFront" and a "GitProject" link. Below the header is a dark green navigation bar with a "Novo" button. The main content area is titled "Listagem de Alunos". A table displays two student records:

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		

A screenshot of a web browser window titled "ProjCrudFront". The address bar shows "localhost:5173/listings/create". The main content area has a dark green header with the title "ProjCrudFront" and a "Git" link. Below the header is a white card with a rounded rectangle containing the form fields. The form is titled "DADOS DO ALUNO" and contains four input fields: "Nome" (Nome: teste POST), "Cpf" (Cpf: 12345678900), "Data de Nascimento" (Data de Nascimento: 28 / 08 / 1982), and "Renda" (Renda: 1000.9). At the bottom are two buttons: a red "Cancelar" button and a blue "Salvar" button.

The screenshot shows a web application titled "ProjCrudFront" running at "localhost:5173/listings". The page has a dark header with the title and a "GitProject" link. Below the header is a section titled "Listagem de Alunos" with a "Novo" button. A table displays 10 student records with columns: ID, NOME, CPF, NASCIMENTO, and RENDA. Each row includes edit and delete icons. The last record in the table is highlighted with yellow boxes around its ID (10), name ("teste POST"), and CPF ("12345678900").

ID	NOME	CPF	NASCIMENTO	RENDAS
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

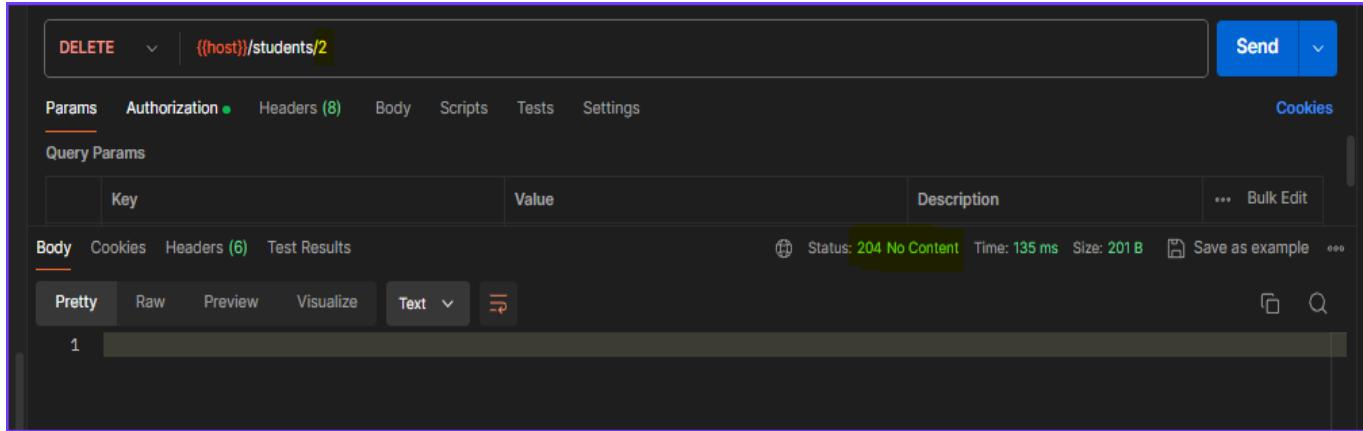
GitHub-18

The screenshot shows a GitHub commit history for a project. The top bar says "CRUD: Insert with POST". Below it is a "Commit" button with a checkmark icon. The main area shows a list of changes:

- import.sql E:\PROJETOS\PESSOAL\ProjCrud... M (3)
- index.tsx src\pages\NewForm M
- student-service.ts src\services M

IMPLEMENTAR O DELETE COM CONFIRMAÇÃO

Testar resposta no postman



The screenshot shows a Postman interface with a DELETE request to `http://{{host}}/students/2`. The response status is `204 No Content`, indicating the record was successfully deleted.

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



The screenshot shows a web application titled "ProjCrudFront" with a "Listagem de Alunos" (Student List) page. The page displays three student records in a table:

ID	NOME	CPF	NASCIMENTO	RENDAS	ACTION
1	Auricelio Freitas	000.111.222-33	27/08/1982	R\$ 10700.5	
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	

Implementar o componente Modal

```
src > components > DialogConfirmation > index.tsx > ...
1  import ButtonPrimary from '../ButtonPrimary';
2  import ButtonSecondary from '../ButtonSecondary';
3  import './styles.css';
4
5  type Props = {
6    id: number;
7    message: string;
8    onDialogAnswer: Function;
9  }
10 export default function DialogConfirmation( {id, message, onDialogAnswer} : Props) {
11
12   return (
13     <div onClick={() => onDialogAnswer(false, id)}>
14       <div onClick={(event) => event.stopPropagation}>
15         <h2>{message}</h2>
16
17         <div>
18           <div onClick={() => onDialogAnswer(false, id)}>
19             <ButtonSecondary name="Não"/>
20           </div>
21           <div onClick={() => onDialogAnswer(true, id)}>
22             <ButtonPrimary name="Sim"/>
23           </div>
24         </div>
25       </div>
26     );
27   }
28 }
29 }
```

Estilizar o Modal

```
src > components > DialogConfirmation > index.tsx > ...
10  export default function DialogConfirmation({ id, message, onDialogAnswer }: Props) {
11
12    return (
13      <div className="proj-dialog-background" onClick={() => onDialogAnswer(false, id)}>
14        <div className="proj-dialog-box" onClick={(event) => event.stopPropagation}>
15          <h2>{message}</h2>
16
17          <div className="proj-dialog-btn">
18            <div onClick={() => onDialogAnswer(false, id)}>
19              <ButtonSecondary name="Não"/>
20            </div>
21            <div onClick={() => onDialogAnswer(true, id)}>
22              <ButtonPrimary name="Sim"/>
23            </div>
24          </div>
25        </div>
26      </div>
27    )
28  )
29
30  </div>
31
32  <div>
33    <h1>Projekt</h1>
34    <h2>Projekt</h2>
35  </div>
36
```

```
src > components > DialogConfirmation > styles.css > ...
1  .proj-dialog-background {
2    position: fixed;
3    top: 0;
4    left: 0;
5    right: 0;
6    bottom: 0;
7    background-color: rgba(0, 0, 0, 0.5);
8  }
9
10 .proj-dialog-box {
11   display: flex;
12   flex-direction: column;
13   align-items: center;
14   justify-content: center;
15   border-radius: 5px;
16   background-color: var(--proj-color-card-bg);
17   padding: 20px;
18   position: absolute;
19   top: 50%;
20   left: 50%;
21   transform: translate(-50%, -50%);
22 }
23
24 .proj-dialog-box h2 {
25   font-size: 20px;
26   color: var(--proj-color-font-primary);
27   margin-bottom: 20px;
28 }
29
30 .proj-dialog-btn {
31   width: 100%;
32   display: grid;
33   grid-template-columns: 1fr 1fr;
34   grid-gap: 20px;
35 }
36
```

Implementar a chamado do Dialog no Listing

```
src > pages > Listing > index.tsx > ...
103           </tbody>
104       </table>
105   </section>
106   {
107     dialogConfirmationData.visible &&
108     <DialogConfirmation
109       id={dialogConfirmationData.id}
110       message={dialogConfirmationData.message}
111       onDialogAnswer={handleDialogConfirmationAnswer}
112     />
113   }
114   </main>
115
116 }
117
```

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

Implementar o “onClick” do botão Delete

```
src > pages > Listing > index.tsx > ...
89           <tbody>
90         {
91           students.map(student => (
92             <tr key={student.id}>
93               <td className='proj-listing-table-id'>{student.id}</td>
94               <td>{student.name}</td>
95               <td>{student.cpf}</td>
96               <td>{formatDateBR(student.birthDate)}</td>
97               <td>R$ {student.income}</td>
98               <td><img src={editIcon} alt='Editar' onClick={() => {handleUpdate(student.id)}}/></td>
99               <td><img src={deleteIcon} alt='Deletar' onClick={() => handleDelete(student.id)}/></td>
100             </tr>
101           ))
102         )
103       )
104     )
105   )
106 }
```

Implementar a função handleDelete

```
src > pages > Listing > index.tsx > ...
48
49   function handleDelete(studentId: number) {
50     setDialogConfirmationData({ ...dialogConfirmationData, id: studentId, visible: true});
51   }
52 }
```

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

Listagem de Alunos						Novo
ID	NOME	CPF	NASCIMENTO	RENDAS		
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		

localhost:5173/listings 70%

ID	NOME	CPF	NASCIMENTO	RENDAS		
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			R\$ 10700.5		
4	Rafael Sales			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		

NOTA: Ao clicar em “Não” a ação será cancelada.

localhost:5173/listings

ProjCrudFront

GitProject

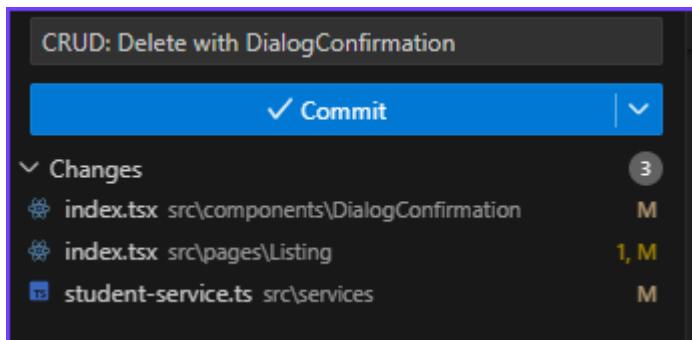
Listagem de Alunos

Novo

ID	NOME	CPF	NASCIMENTO	RENDAS
1	Auricelio Freitas	000.111.222-33	27/08/1982	R\$ 10700.5
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

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 |
28 | }
```

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 }, [ ])
31
```

Visualização web

The screenshot shows a web application interface. At the top, there's a header with 'ProjCrudFront' and 'GitProject'. Below it is a form titled 'DADOS DO ALUNO' containing fields for 'Nome' (Auricelio Freitas), 'Cpf' (000.111.222-33), 'Data de Nascimento' (28/08/1982), and 'Renda' (10700.5). At the bottom of the form are 'Cancelar' and 'Salvar' buttons. To the right, the browser's developer tools are open, specifically the 'Console' tab, which displays an error message for the 'income' field: 'Favor informar um valor positivo'.

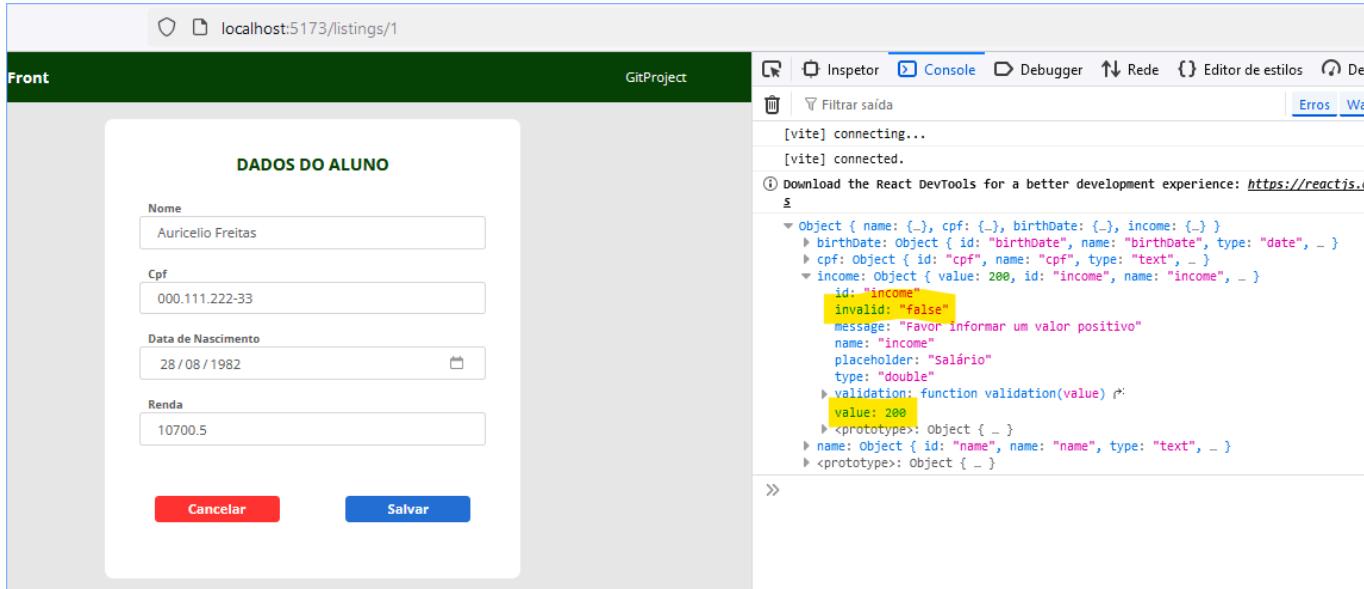
NOTA: Pode clicar para editar qualquer aluno. Basta abrir o “Inspecionar” e ir em “console”.

Realizar teste com valor “positivo”, no ProductForm

```
src > pages > NewForm > index.tsx > ...
54     income: {
55       value: 200,
56       id: "income",
57       name: "income",
58       type: "double",
59       placeholder: "Salário",
60       validation: function(value: any) {
61         return Number(value) > 0;
62       },
63     },
64   },
65 );
```

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 isValid = !inputs[name].validation(inputs[name].value);
30 |     return { ...inputs, [name]: { ...inputs[name], invalid: isValid.toString() }};
31 |   }

```

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

Visualização web

```

localhost:5173/listings/1
GitProject
Inspecor Console Debugger Rede Editor de estilos Erros
[vite] connecting...
[vite] connected.
① Download the React DevTools for a better development experience: https://reactjs.org/config/
Object { name: {}, cpf: {}, birthDate: {}, income: {} }
  ► birthDate: Object { id: "birthdate", name: "birthdate", type: "date", ... }
  ► cpf: Object { id: "cpf", name: "cpf", type: "text", ... }
  ► income: Object { value: 200, id: "income", name: "income", ... }
    id: "income"
      invalid: "false"
      message: "Favor informar um valor positivo"
      name: "income"
      placeholder: "Salário"
      type: "double"
    + validation: function validation(value) {
      length: 1
      name: "validation"
      prototype: Object { ... }
      <prototype>: function ()
      value: 200
    } <prototype>: Object { ... }
    name: Object { id: "name", name: "name", type: "text", ... }
    <prototype>: Object { ... }
```

```

## Testar com um campo sem o validation

```

src > pages > NewForm > index.tsx > ...
18 useEffect(() => {
19
20 const obj = forms.validate(formData, "name");
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

```

Nome: "Auricelio Freitas"
Cpf: "000.111.222-33"
Data de Nascimento: "28/08/1982"
Renda: "10700.5"

Cancelar Salvar
  
```

```

[{"id": "name", "name": "name", "type": "text", "value": ""}, {"id": "birthDate", "name": "birthDate", "type": "date", "value": "2023-08-28"}, {"id": "cpf", "name": "cpf", "type": "text", "value": "000.111.222-33"}, {"id": "income", "name": "income", "type": "text", "value": "10700.5"}]
  
```

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

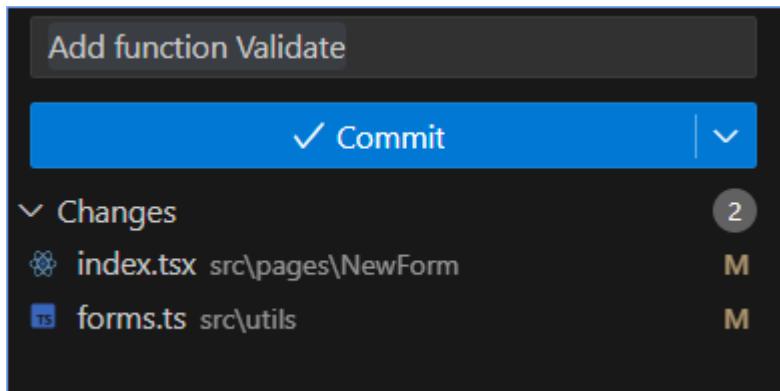
Remover os testes anteriores

```

src > pages > NewForm > index.tsx ...
17
18     useEffect(() => {
19
20         const obj = forms.validate(formData, "name");
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     }, [])
  
```

```
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 > ...
1  export default function FormInput(props: any) {
2
3      const { validation, invalid, ...inputProps } = props;
4
5      return (
6          <input { ...inputProps } data-invalid={invalid} />
7      );
8 }
```

NOTA: Já havíamos deixada implementada em momentos anteriores.

Refatorar a função handleInputChange, no NewForm

```
src > pages > NewForm > ✎ index.tsx > ...
00
67     function handleInputChange(event: any) {
68         const dataUpdated = forms.update(formData, event.target.name, event.target.value);
69         setFormData(dataUpdated);
70     }
```

Validar os dados atualizados

```
src > pages > NewForm > ✎ index.tsx > ...
03
64     function handleInputChange(event: any) {
65         const dataUpdated = forms.update(formData, event.target.name, event.target.value);
66         const dataValidated = forms.validate(dataUpdated, event.target.name);
67         setFormData(dataValidated);
68     }
69
```

Visualização web

DADOS DO ALUNO

Nome

Cpf

Data de Nascimento

Renda

[Cancelar](#) [Salvar](#)

```
<!DOCTYPE html>
<html lang="en"> [event]
  <head> [event] </head>
  <body>
    <div id="root"> [event]
      <header class="proj-header"> [event] </header> [flex]
      <main>
        <section id="proj-form-section" class="proj-container">
          <div class="proj-form-container">
            <form class="proj-card proj-form"> [flex]
              <h2 class="proj-mb20">Dados do Aluno</h2>
              <div class="proj-form-controls-container" style="grid">
                <div> [event] </div>
                <div> [event] </div>
                <div> [event] </div>
                <div> [event] </div>
              </div>
              <div> [event]
                <h5>Renda</h5>
                <input id="income" class="proj-form-control" name="income" type="double" placeholder="Salário" message="Favor informar um valor positivo" value="-1200" data-invalid="true" event>
              </div>
            </form>
            <div class="proj-form-buttons" style="grid">
              <a href="#" style="background-color: red; color: white; padding: 5px 10px;">Cancelar
              <a href="#" style="background-color: #007bff; color: white; padding: 5px 10px;">Salvar
            </div>
          </div>
        </section>
      </main>
    </div>
  </body>
</html>
```

DADOS DO ALUNO

Nome

Cpf

Data de Nascimento

Renda

[Cancelar](#) [Salvar](#)

```
<!DOCTYPE html>
<html lang="en"> [event]
  <head> [event] </head>
  <body>
    <div id="root"> [event]
      <header class="proj-header"> [event] </header> [flex]
      <main>
        <section id="proj-form-section" class="proj-container">
          <div class="proj-form-container">
            <form class="proj-card proj-form"> [flex]
              <h2 class="proj-mb20">Dados do Aluno</h2>
              <div class="proj-form-controls-container" style="grid">
                <div> [event] </div>
                <div> [event] </div>
                <div> [event] </div>
                <div> [event] </div>
              </div>
              <div> [event]
                <h5>Renda</h5>
                <input id="income" class="proj-form-control" name="income" type="double" placeholder="Salário" message="Favor informar um valor positivo" value="200" data-invalid="false" event>
              </div>
            </form>
            <div class="proj-form-buttons" style="grid">
              <a href="#" style="background-color: red; color: white; padding: 5px 10px;">Cancelar
              <a href="#" style="background-color: #007bff; color: white; padding: 5px 10px;">Salvar
            </div>
          </div>
        </section>
      </main>
    </div>
  </body>
</html>
```

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 |
61 | }
```

Visualização web

The image displays two side-by-side screenshots of a web application interface titled "DADOS DO ALUNO". Both screenshots show a form with four fields: Nome, Cpf, Data de Nascimento, and Renda. The "Nome" and "Cpf" fields are marked as valid (white background). The "Data de Nascimento" field has a calendar icon next to it. The "Renda" field contains the value "-1" and is highlighted with a red border, indicating it is invalid. At the bottom of each screenshot are two buttons: "Cancelar" (red) and "Salvar" (blue).

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 > ...
29     const [formData, setFormData] = useState<any>({
30         name: {
31             value: "",
32             id: "name",
33             name: "name",
34             type: "text",
35             placeholder: "Nome",
36             validation: function(value: any) {
37                 return value.length >= 3 && value.length <= 50;
38             },
39             message: "Favor informar um nome entre 3 e 50 caracteres"
40         },
41         cpf: {
42             value: "",
43             id: "cpf",
44             name: "cpf",
45             type: "text",
46             placeholder: "CPF",
47             validation: function(value: any) {
48                 return /^[([0-9]{3}.[0-9]{3}.[0-9]{3}-[0-9]{2})$/.test(value);
49             },
50             message: "Favor informar um CPF válido"
51         },
52         birthDate: {
53             value: "",
54             id: "birthDate",
55             name: "birthDate",
56             type: "date",
57             placeholder: "Data de Nascimento",
58             validation: function(value: any) {
59                 return /^[([0-9]{4}.[0-9]{2}.[0-9]{2})$/.test(value);
60             },
61             message: "Favor informar uma data válida"
62         },
63         income: {
64             value: 0,
65             id: "income",
66             name: "income",
67             type: "double",
68             placeholder: "Salário",
69             validation: function(value: any) {
70                 return Number(value) >= 0;
71             },
72             message: "Favor informar um valor positivo"
73         },
74     });
75 
```

Implementar a mensagem de erro, no FormInput

```
src > pages > NewForm > index.tsx > ...
111
112
113
114
115
116 |           <FormInput
117 |             { ...formData.name }
118 |             className="proj-form-control"
119 |             onChange={handleInputChange}
120 |           />
121 |           <div className="proj-form-error">{formData.name.message}</div>
122 </div>
123 <div>
124 |             <h5>Cpf</h5>
125 |             <FormInput
126 |               { ...formData.cpf }
127 |               className="proj-form-control"
128 |               onChange={handleInputChange}
129 |             />
130 |             <div className="proj-form-error">{formData.cpf.message}</div>
131 </div>
132 <div>
133 |             <h5>Data de Nascimento</h5>
134 |             <FormInput
135 |               { ...formData.birthDate }
136 |               className="proj-form-control"
137 |               onChange={handleInputChange}
138 |             />
139 |             <div className="proj-form-error">{formData.birthDate.message}</div>
140 </div>
141 <div>
142 |             <h5>Renda</h5>
143 |             <FormInput
144 |               { ...formData.income }
145 |               className="proj-form-control"
146 |               onChange={handleInputChange}
147 |             />
148 |             <div className="proj-form-error">{formData.income.message}</div>
149 </div>
150 </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 }  
72
```

Visualização web

DADOS DO ALUNO

Nome

Favor informar um nome entre 3 e 50 caracteres

Cpf

Favor informar um CPF válido

Data de Nascimento
 CALENDAR

Favor informar uma data válida

Renda

Favor informar um valor positivo

Cancelar **Salvar**

DADOS DO ALUNO

Nome

Favor informar um nome entre 3 e 50 caracteres

Cpf

Favor informar um CPF válido

Data de Nascimento
 CALENDAR

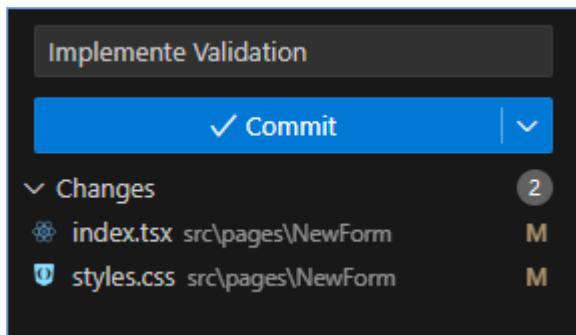
Favor informar uma data válida

Renda

Favor informar um valor positivo

Cancelar **Salvar**

GitHub-21



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

DADOS DO ALUNO

Nome
 Favor informar um nome entre 3 e 50 caracteres

Cpf

Data de Nascimento

Renda

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 > ...
1  export default function FormInput(props: any) {
2
3    const { validation, invalid, onTurnDirty, ...inputProps } = props;
4
5    return (
6      <input { ...inputProps } data-invalid={invalid} />
7    );
8  }
9
```

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 > ...
32
33  export function toDirty(inputs: any, name: string) {
34    return { ...inputs, [name]: { ...inputs[name], dirty: "true" } }
35  }
36
```

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

The screenshot shows a web browser window with the URL `localhost:5173/listings/create`. The page title is `ProjCrudFront`. On the left, there is a green header bar with the text `GitProject`. The main content area contains a form titled `DADOS DO ALUNO` with fields for `Nome`, `Cpf`, `Data de Nascimento` (with a date input field), and `Renda` (with a numeric input field containing the value `0`). Below the form are two buttons: `Cancelar` (red) and `Salvar` (blue). To the right of the form is a developer tools console window. The console shows the following output:

```
[vite] connecting...
[vite] connected.
① Download the React DevTools for a better development experience: https://reactjs.org
Object { name: {}, cpf: {}, birthDate: {}, income: {} }
  birthDate: Object { id: "birthDate", name: "birthdate", type: "date", ... }
  cpf: Object { id: "cpf", name: "cpf", type: "text", ... }
  income: Object { value: 0, id: "income", name: "income", ... }
    dirty: "true"
    id: "income"
    message: "Favor informar um valor positivo"
    name: "income"
    placeholder: "Salário"
    type: "double"
    validation: function validation(value) {
      value: 0
      <prototype>: Object { ... }
      name: Object { id: "name", name: "name", type: "text", ... }
      <prototype>: Object { ... }
```

Implementar o evento onBlur, no componente FormInput

```
src > components > FormInput > index.tsx > ...
1  export default function FormInput(props: any) {
2
3      const {
4          validation,
5          invalid = "false",
6          dirty = "false",
7          onTurnDirty,
8          ...inputProps
9      } = props;
10
11     function handleBlur() {
12         onTurnDirty(props.name);
13     }
14
15     return (
16         <input
17             { ...inputProps }
18             onBlur={handleBlur}
19             data-invalid={invalid}
20             data-dirty={dirty}
21         />
22     );
23 }
```

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 (
111         <div>
```

Visualização web

The screenshot shows a browser window with developer tools open. On the left is a form titled "DADOS DO ALUNO" with fields for Nome, CPF, Data de Nascimento, and Renda. On the right, the browser's DOM inspector highlights a specific input field for "Renda". The highlighted code snippet shows an element with the ID "income" and the attribute "data-invalid=true". The browser's status bar at the bottom indicates "data-invalid: true".

```
<!DOCTYPE html>
<html lang="en">[event]
<head>[event]</head>
<body>
  <div id="root">[event]
    <header class="proj-header">[...]</header>[flex]
    <main>
      <section id="proj-form-section" class="proj-container">
        <div class="proj-form-container">
          <form class="proj-card proj-form">[flex]
            <h2 class="proj-mb20">Dados do Aluno</h2>
            <div class="proj-form-controls-container">[grid]
              <div>[...]
              <div>[...]
              <div>[...]
            </div>
            <div>
              <h5>Renda</h5>
              <input id="income" class="proj-form-control" name="income" type="double" placeholder="Salário" message="Favor informar um valor positivo" data-invalid="true" data-dirty="false" value="0">[event]
              <div class="proj-form-error">Favor informar um valor positivo</div>
            </div>
          </div>
          <div class="proj-form-buttons">[flex]
            <a href="/listings">[...]</a>[event]
            <button class="proj-btn-primary">[...]</button>[flex]
          </div>
        </form>
      </div>
    </main>
  </div>
</body>
</html>
```

NOTA: Por padrão o “data-invalid” e o “data-dirty” são falsos.

Simulação

The screenshot shows a browser's developer tools with the 'Inspector' tab selected. The page content is a form titled 'DADOS DO ALUNO'. The 'Nome' field contains 'Nome' and has a placeholder 'Nome'. The 'Cpf' field contains 'CPF' and has a placeholder 'CPF'. The 'Data de Nascimento' field contains 'dd / mm / aaaa' and has a date picker icon. The 'Renda' field contains '0' and has a placeholder 'Renda'. Below the form are two buttons: 'Cancelar' (red) and 'Salvar' (blue). The right side of the screen shows the DOM tree. A yellow box highlights the 'data-dirty=true' and 'data-invalid=true' attributes on the 'Renda' input field. Another yellow box highlights the error message 'Favor informar um valor positivo' located in a sibling div.

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

DADOS DO ALUNO

Nome: Nome

Cpf: CPF

Data de Nascimento: dd / mm /aaaa

Renda: 0

Cancel **Salvar**

```
<!DOCTYPE html>
<html lang="en"> [event]
  <head> [::]</head>
  <body>
    <div id="root"> [event]
      <header class="proj-header"> [::] </header> flex
      <main>
        <section id="proj-form-section" class="proj-container">
          <div class="proj-form-container">
            <form class="proj-card proj-form"> flex
              <h2 class="proj-mb20">Dados do Aluno</h2>
              <div class="proj-form-controls-container"> grid
                <div>
                  <div>
                    <h5>Renda</h5>
                    <input id="income" class="proj-form-control" name="income" type="double" placeholder="Salário" message="Favor informar um valor positivo" data-invalid="false" data-dirty="false" value="0"> [event]
                    <div class="proj-form-error">Favor informar um valor positivo</div>
                  </div>
                </div>
                <div class="proj-form-buttons"> [::] </div> flex
              </form>
            </div>
          </section>
        </main>
      </div>
```

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

Digitar um valor inválido no campo

DADOS DO ALUNO

Nome: Nome

Cpf: CPF

Data de Nascimento: dd / mm /aaaa

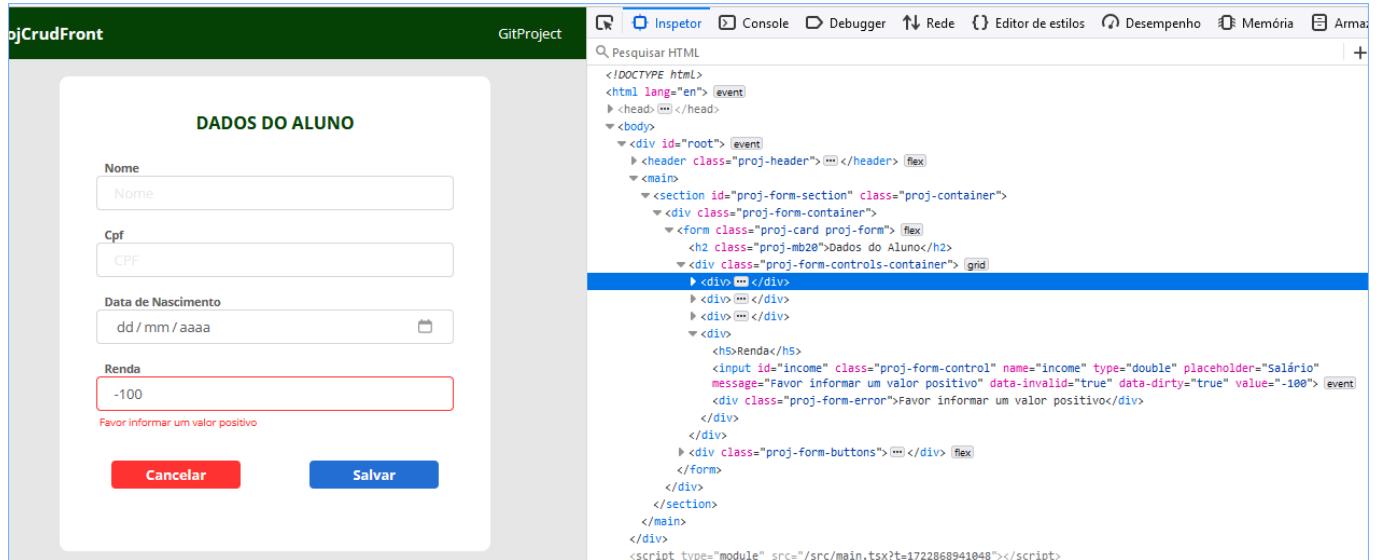
Renda: -100

Cancel **Salvar**

```
<!DOCTYPE html>
<html lang="en"> [event]
  <head> [::]</head>
  <body>
    <div id="root"> [event]
      <header class="proj-header"> [::] </header> flex
      <main>
        <section id="proj-form-section" class="proj-container">
          <div class="proj-form-container">
            <form class="proj-card proj-form"> flex
              <h2 class="proj-mb20">Dados do Aluno</h2>
              <div class="proj-form-controls-container"> grid
                <div>
                  <div>
                    <h5>Renda</h5>
                    <input id="income" class="proj-form-control" name="income" type="double" placeholder="Salário" message="Favor informar um valor positivo" data-invalid="true" data-dirty="false" value="-100"> [event]
                    <div class="proj-form-error">Favor informar um valor positivo</div>
                  </div>
                </div>
                <div class="proj-form-buttons"> [::] </div> flex
              </form>
            </div>
          </section>
        </main>
      </div>
      <script type="module" src="/src/main.tsx">+1722868841048</script>
```

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 inválido e sair do campo



The screenshot shows a browser window titled "ProjCrudFront" with a sub-header "GitProject". Inside, a modal dialog is displayed with the title "DADOS DO ALUNO". The form contains fields for "Nome" (Name), "Cpf" (CPF), "Data de Nascimento" (Date of Birth), and "Renda" (Income). The "Renda" field has a value of "-100" and is highlighted with a red border, indicating it is invalid. Below the field, a message says "Favor informar um valor positivo" (Please enter a positive value). At the bottom of the modal are two buttons: "Cancelar" (Cancel) in red and "Salvar" (Save) in blue.

The right side of the screen shows the browser's developer tools with the "Inspector" tab selected. It displays the HTML structure of the page, focusing on the "Renda" field. The code snippet includes attributes like "data-invalid=true" and "data-dirty=true" on the input field, and a corresponding error message is shown in the DOM.

```

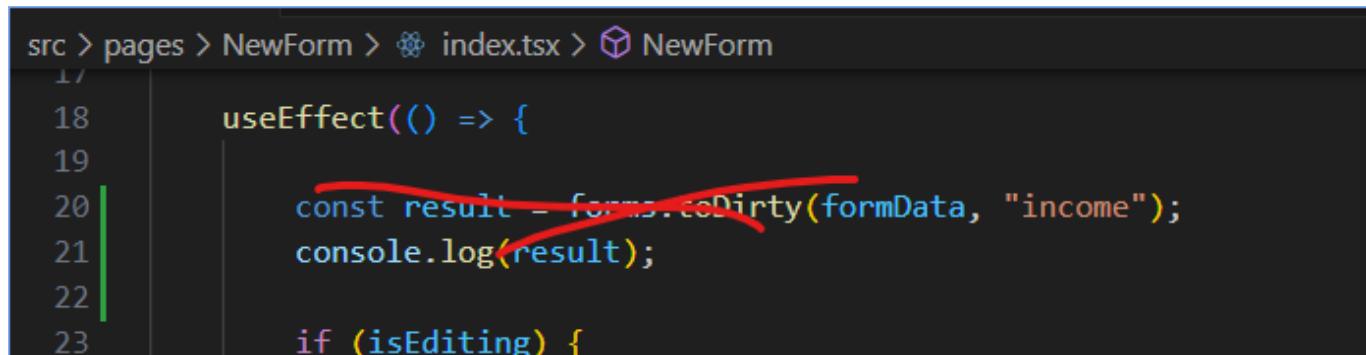
<!DOCTYPE html>
<html lang="en"> [event]
  <head> [event]
    <body>
      <div id="root"> [event]
        <header class="proj-header"> [event]
          <main>
            <section id="proj-form-section" class="proj-container">
              <div class="proj-form-container">
                <form class="proj-card proj-form"> [flex]
                  <h2 class="proj-mb28">Dados do Aluno</h2>
                  <div class="proj-form-controls-container"> [grid]
                    <div> [div]
                      <div> [div]
                        <div> [div]
                          <div> [div]
                            <div> [div]
                              <div> [div]
                                <div> [div]
                                  <div> [div]
                                    <div> [div]
                                      <div> [div]
                                        <div> [div]
                                          <div> [div]
                                            <div> [div]
                                              <div> [div]
                                                <div> [div]
                                                  <div> [div]
                                                    <div> [div]
                                                      <div> [div]
                                                        <div> [div]
                                                          <div> [div]
                                                            <div> [div]
                                                              <div> [div]
                                                                <div> [div]
                                                                  <div> [div]
                                                                    <div> [div]
                                                                      <div> [div]
                                                                        <div> [div]
                                                                          <div> [div]
                                                                            <div> [div]
                                                                              <div> [div]
                                                                                <div> [div]
                                                                                  <div> [div]
                                                                                    <div> [div]
                                                                                      <div> [div]
                                                                                        <div> [div]
              </div>
            </section>
          </main>
        </div>
        <script type="module" src="/src/main.tsx?t=1722868941048"></script>
      </div>
    </body>
  </html>

```

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



The screenshot shows a code editor with the file path "src > pages > NewForm > index.tsx". The code is as follows:

```

src > pages > NewForm > index.tsx > NewForm
1/
18  useEffect(() => {
19
20    const result = formsIsDirty(formData, "income");
21    console.log(result);
22
23    if (isEditing) {

```

A portion of the code between lines 20 and 23 is crossed out with a large red mark, indicating it has been removed.

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

```
src > utils > forms.ts > ...
50
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 > ...
41
42  export function dirtyAndValidate(inputs: any, name: string) {
43    const dataDirty = toDirty(inputs, name);
44    return validate(dataDirty, name);
45 }
46
```

Refatorar o handleInputChange, do NewForm

```
src > pages > NewForm > index.tsx > ...
95
96  function handleInputChange(event: any) {
97    setFormData(forms.updateAndValidate(formData, event.target.name, event.target.value));
98 }
```

Refatorar o handleTurnDirty, do NewForm

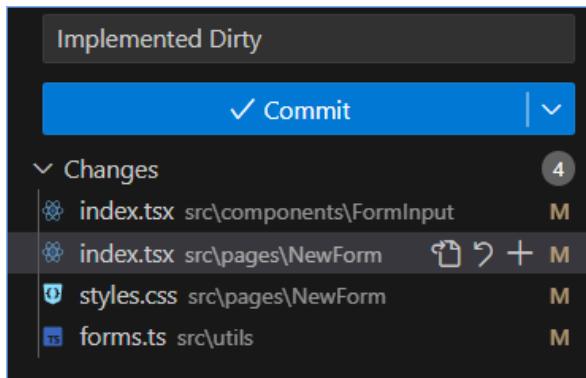
```
src > pages > NewForm > index.tsx > ...
100
101  function handleTurnDirty(name: string) {
102    setFormData(forms.dirtyAndValidate(formData, name));
103 }
```

Visualização web

The screenshot shows a web application interface with a dark green header bar containing the text 'ProjCrudFront' on the left and 'GitProject' on the right. Below the header is a white form card with a title 'DADOS DO ALUNO' in bold capital letters. The form contains four input fields: 'Nome' (Name), 'Cpf' (CPF), 'Data de Nascimento' (Date of Birth), and 'Renda' (Income). Each field has a red border and a placeholder text. Below each field is a validation message in red. At the bottom of the form are two buttons: 'Cancelar' (Cancel) in red and 'Salvar' (Save) in blue.

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

ProjCrudFront

GitProject

DADOS DO ALUNO

Nome

Cpf

Data de Nascimento

Renda

dd / mm / aaaa

Cancelar

Salvar

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

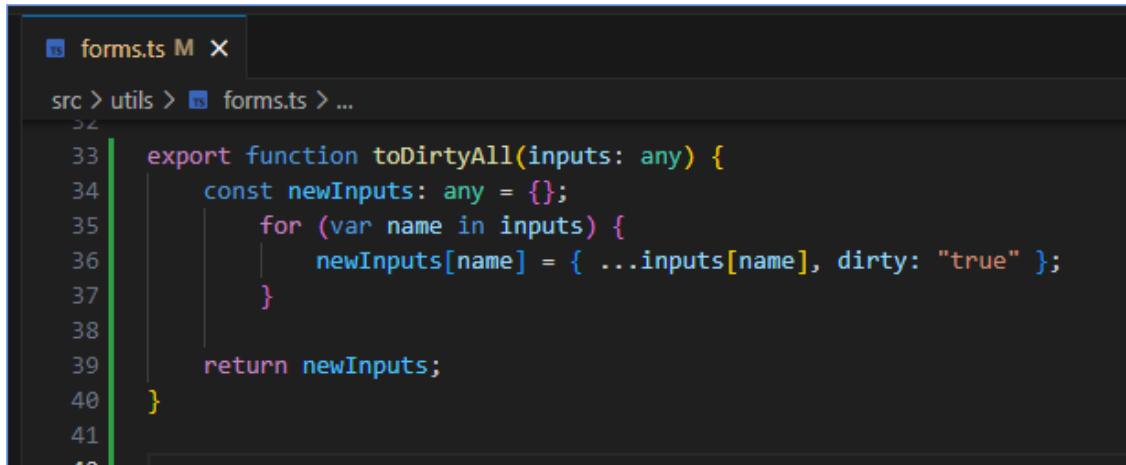
The screenshot shows a web application titled "ProjCrudFront" running at "localhost:5173/listings". The page displays a table of student records with columns: ID, NOME, CPF, NASCIMENTO, and RENDA. Each row includes edit and delete icons. A new record (ID 9) is shown with empty fields for NOME, CPF, and NASCIMENTO, while RENDA is set to R\$ 0. The entire screenshot is framed by a purple border.

ID	NOME	CPF	NASCIMENTO	RENDAS
1	Auricelio Freitas	000.111.222-33	28/08/1982	R\$ 10700.5
2	Miguel Soares	000.111.222-33	10/02/1989	R\$ 20700.5
3	Matheus Marques	000.111.222-33	23/11/1992	R\$ 10700.5
4	Rafael Sales	000.111.222-33	13/10/1988	R\$ 10700.5
5	Bruno Freitas	000.111.222-33	22/04/1987	R\$ 9700.5
6	Romulo Araujo	000.111.222-33	09/01/1991	R\$ 8700.5
7	Ryan Sousa	000.111.222-33	26/03/1994	R\$ 7700.5
8	test1	000.111.222-33	32/12/1969	R\$ 1024
9			32/12/1969	R\$ 0

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

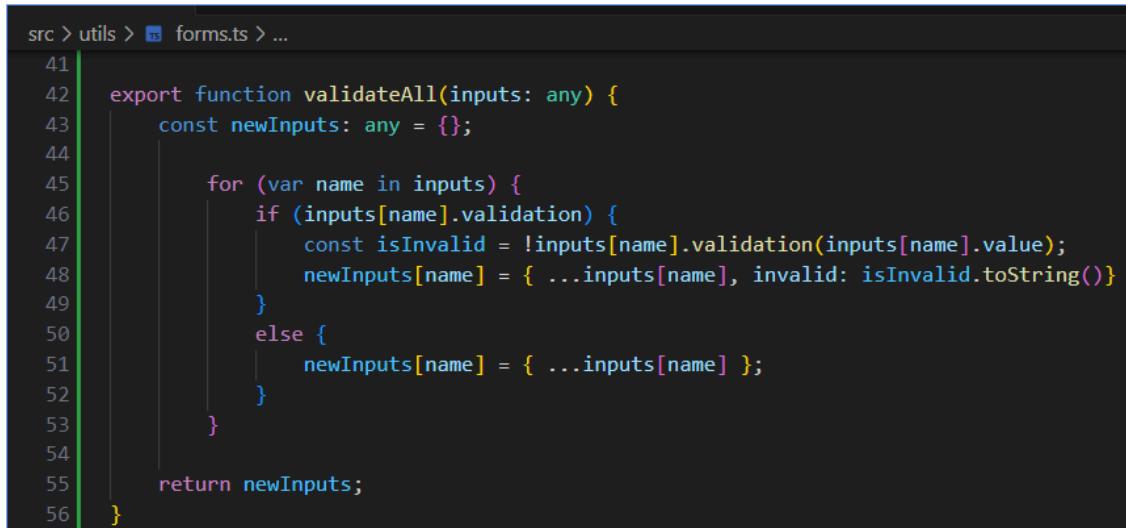
OBJETIVO: *Sujar todos os campos.*



```
src > utils > forms.ts > ...
33 | export function toDirtyAll(inputs: any) {
34 |   const newInputs: any = {};
35 |   for (var name in inputs) {
36 |     newInputs[name] = { ...inputs[name], dirty: "true" };
37 |   }
38 |
39 |   return newInputs;
40 |
41 }
```

Implementar a função `validateAll`, no utilitário forms

OBJETIVO: *Validar apenas se existir o atributo “validation”.*



```
src > utils > forms.ts > ...
41 |
42 | export function validateAll(inputs: any) {
43 |   const newInputs: any = {};
44 |
45 |   for (var name in inputs) {
46 |     if (inputs[name].validation) {
47 |       const isValid = !inputs[name].validation(inputs[name].value);
48 |       newInputs[name] = { ...inputs[name], invalid: isValid.toString() };
49 |     }
50 |     else {
51 |       newInputs[name] = { ...inputs[name] };
52 |     }
53 |   }
54 |
55 |   return newInputs;
56 }
```

Implementar a função dirtyAndValidateAll, no forms do utils

OBJETIVO: *Sujar e Validar todos os campos.*

```
src > utils > forms.ts > ...
72
73 | export function dirtyAndValidateAll(inputs: any) {
74 |   return validateAll(toDirtyAll(inputs));
75 |
76 }
```

Implementar a função hasAnyInvalid, no forms do utils

OBJETIVO: *Testar se há algum campo inválido.*

```
src > utils > forms.ts > ...
70
71 | export function hasAnyInvalid(inputs: any) {
72 |
73 |   for (var name in inputs) {
74 |     if (inputs[name].dirty === "true" && inputs[name].invalid === "true") {
75 |       return true;
76 |     }
77 |   }
78 |   return false;
79 |
80 | }
```

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

The screenshot shows a web application interface titled "ProjCrudFront" at the top left and "GitProject" at the top right. A central modal window is displayed with the title "DADOS DO ALUNO". The form contains four fields: "Nome" (Name), "Cpf", "Data de Nascimento" (Date of Birth), and "Renda" (Income). Each field has a red border and a placeholder text. Below each field is a red error message indicating validation rules. At the bottom of the modal are two buttons: "Cancelar" (Cancel) in red and "Salvar" (Save) in blue.

DADOS DO ALUNO

Nome
Nome

Favor informar um nome entre 3 e 50 caracteres

Cpf
CPF

Favor informar um CPF válido

Data de Nascimento
dd / mm / aaaa

Favor informar uma data válida

Renda
Salário

Favor informar um valor positivo

Cancelar **Salvar**

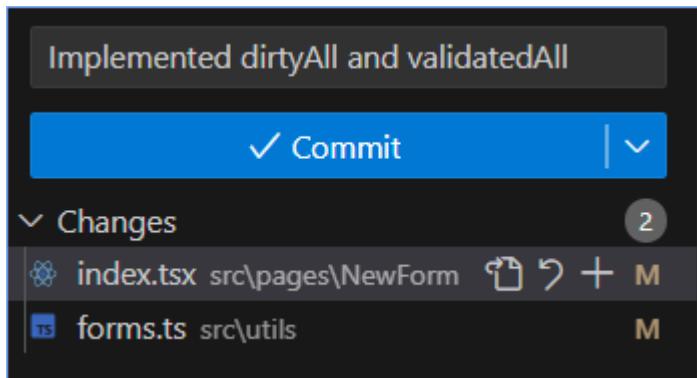
NOTA: Se o usuário tentar salvar um formulário em branco, todas as dependências serão acionadas.

Enviar formulário com alguma pendência

The screenshot shows a web application interface titled "ProjCrudFront" at the top left and "GitProject" at the top right. A modal window titled "DADOS DO ALUNO" is displayed in the center. The form contains four fields: "Nome" (Name) with value "teste", "Cpf" (CPF) with value "CPF" (which is highlighted in red), "Data de Nascimento" (Date of Birth) with value "02 / 08 / 2001", and "Renda" (Income) with value "100". Below the form are two buttons: "Cancelar" (Cancel) in red and "Salvar" (Save) in blue. A red error message "Favor informar um CPF válido" (Please enter a valid CPF) is displayed next to the CPF input field.

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

GitHub-23



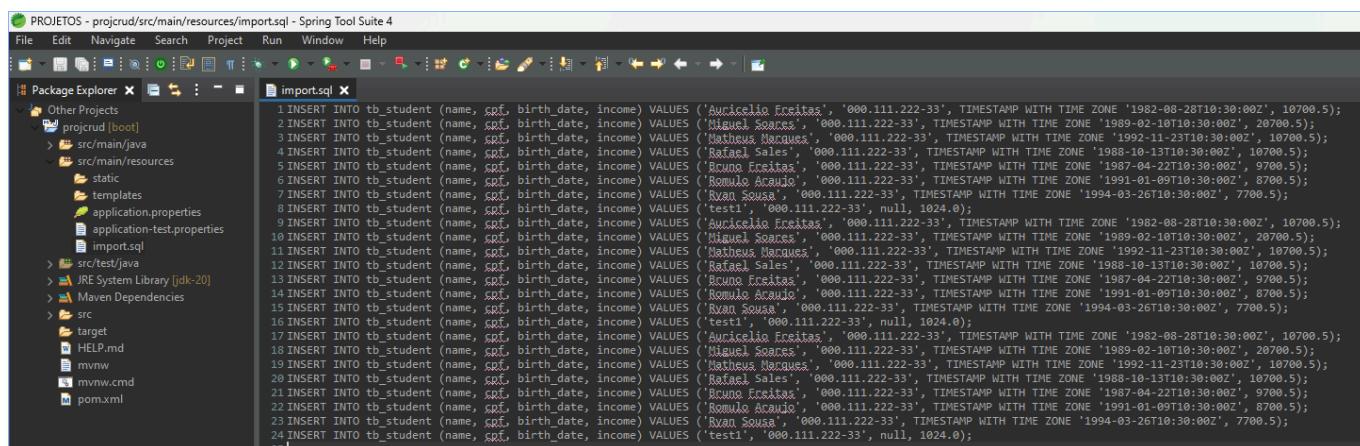
PAGINAÇÃO

IMPLEMENTAR O SEED DO BACKEND

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

Implementar o import.sql

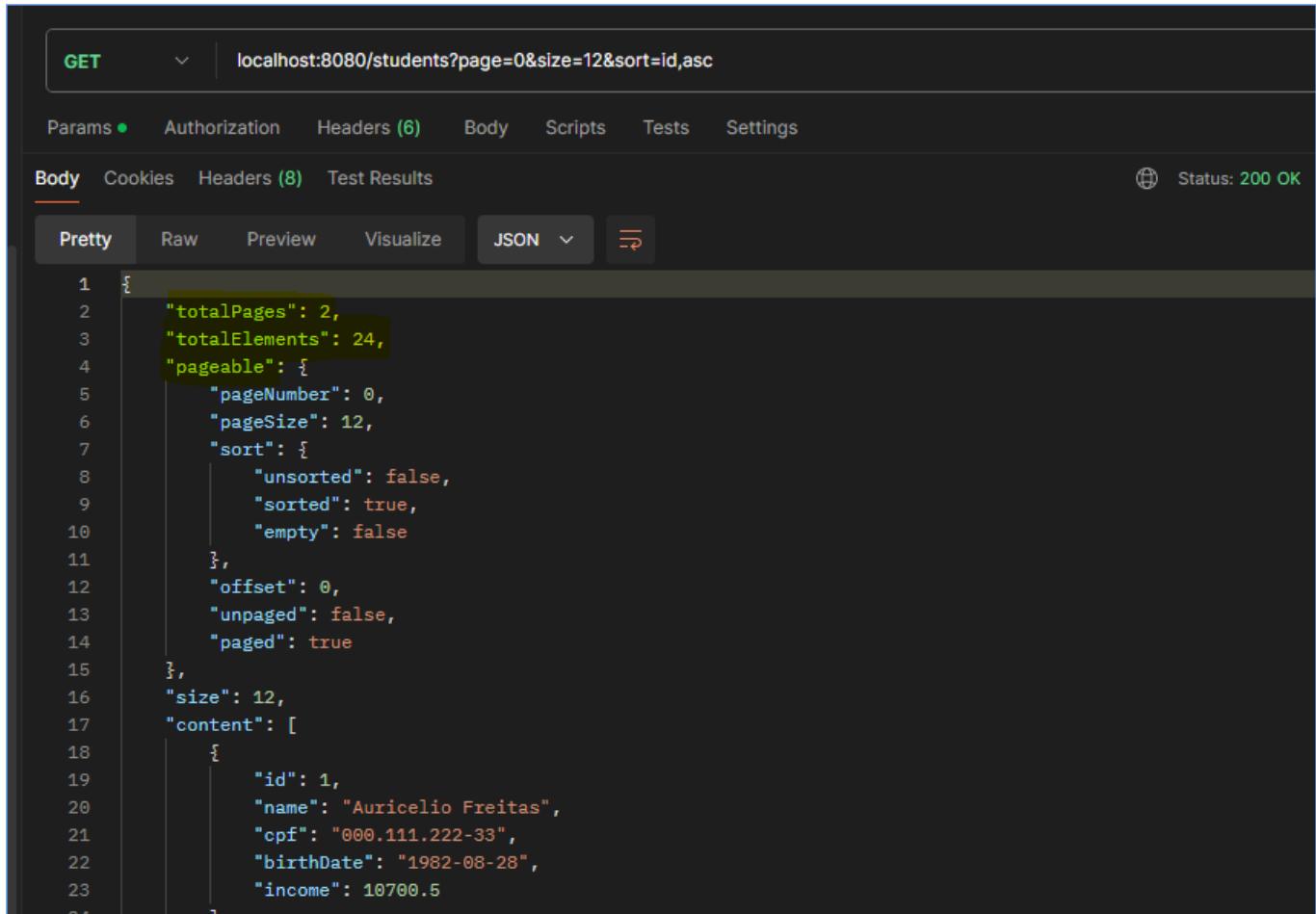
```
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Auricelio Freitas', '000.111.222-33', 'TIMESTAMP WITH TIME ZONE '1982-08-28T10:30:00Z', 10700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Miguel Soares', '000.111.222-33', 'TIMESTAMP WITH TIME ZONE '1989-02-10T10:30:00Z', 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.



The screenshot shows a Postman request for the endpoint `localhost:8080/students?page=0&size=12&sort=id,asc`. The response status is `200 OK`. The response body is a paginated JSON object:

```
1 {  
2     "totalPages": 2,  
3     "totalElements": 24,  
4     "pageable": {  
5         "pageNumber": 0,  
6         "pageSize": 12,  
7         "sort": {  
8             "unsorted": false,  
9             "sorted": true,  
10            "empty": false  
11        },  
12        "offset": 0,  
13        "unpaged": false,  
14        "paged": true  
15    },  
16    "size": 12,  
17    "content": [  
18        {  
19            "id": 1,  
20            "name": "Auricelio Freitas",  
21            "cpf": "000.111.222-33",  
22            "birthDate": "1982-08-28",  
23            "income": 10700.5  
24        }  
25    ]  
26}
```

VISUALIZAR A RENDERIZAÇÃO NO FRONTEND

ProjCrudFront					GitProject
Listagem de Alunos					
Novo					
ID	NOME	CPF	NASCIMENTO	RENDA	
1	Auricelio Freitas	000.111.222-33	28/08/1982	R\$ 10700,5	 
2	Miguel Soares	000.111.222-33	10/02/1989	R\$ 20700,5	 
3	Matheus Marques	000.111.222-33	23/11/1992	R\$ 10700,5	 
4	Rafael Sales	000.111.222-33	13/10/1988	R\$ 10700,5	 
5	Bruno Freitas	000.111.222-33	22/04/1987	R\$ 9700,5	 
6	Romulo Araujo	000.111.222-33	09/01/1991	R\$ 8700,5	 
7	Ryan Sousa	000.111.222-33	26/03/1994	R\$ 7700,5	 
8	test11	000.111.222-33	32/12/1969	R\$ 1024	 
9	Auricelio Freitas	000.111.222-33	28/08/1982	R\$ 10700,5	 
10	Miguel Soares	000.111.222-33	10/02/1989	R\$ 20700,5	 
11	Matheus Marques	000.111.222-33	23/11/1992	R\$ 10700,5	 
12	Rafael Sales	000.111.222-33	13/10/1988	R\$ 10700,5	 
13	Bruno Freitas	000.111.222-33	22/04/1987	R\$ 9700,5	 
14	Romulo Araujo	000.111.222-33	09/01/1991	R\$ 8700,5	 
15	Ryan Sousa	000.111.222-33	26/03/1994	R\$ 7700,5	 
16	test11	000.111.222-33	32/12/1969	R\$ 1024	 
17	Auricelio Freitas	000.111.222-33	28/08/1982	R\$ 10700,5	 
18	Miguel Soares	000.111.222-33	10/02/1989	R\$ 20700,5	 
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	test11	000.111.222-33	32/12/1969	R\$ 1024	 

INSTALAR O PLUGIN DE PAGINAÇÃO

URL do react paginate:

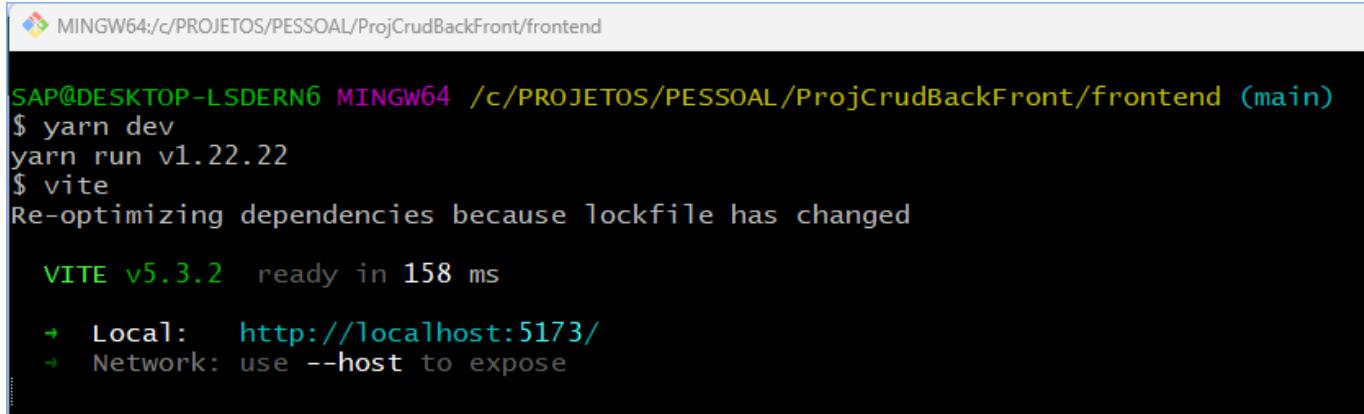
- <https://github.com/AdeleD/react-paginate>

Instalação

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

```
MINGW64:/c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend (main)
$ yarn add react-paginate @types/react-paginate
yarn add v1.22.22
[1/4] Resolving packages...
[2/4] Fetching packages...
[3/4] Linking dependencies...
[4/4] Building fresh packages...
success Saved lockfile.
success Saved 5 new dependencies.
info Direct dependencies
└─ @types/react-paginate@7.1.4
   └─ react-paginate@8.2.0
info All dependencies
└─ @types/react-paginate@7.1.4
   └─ object-assign@4.1.1
   └─ prop-types@15.8.1
   └─ react-is@16.13.1
   └─ react-paginate@8.2.0
Done in 2.16s.
```

Rodar o projeto

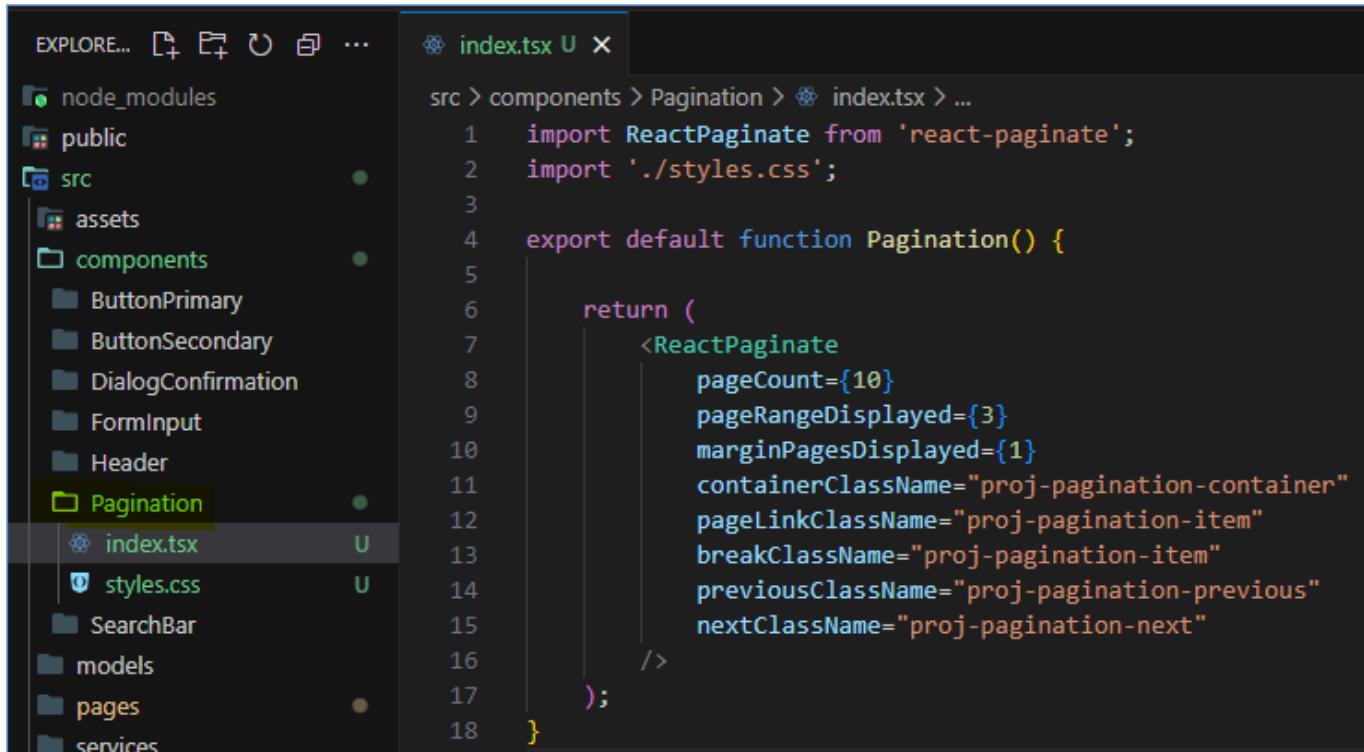


```
MINGW64:/c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend
SAP@DESKTOP-LSDERN6 MINGW64 /c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend (main)
$ yarn dev
yarn run v1.22.22
$ vite
Re-optimizing dependencies because lockfile has changed
VITE v5.3.2 ready in 158 ms

→ Local: http://localhost:5173/
→ Network: use --host to expose
```

IMPLEMENTAR O PAGINATION

Criar o componente de paginação



The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows the project structure. The `Pagination` folder is selected, revealing its contents: `index.tsx`, `styles.css`, and `SearchBar`.
- Code Editor:** The `index.tsx` file is open, displaying the following code:

```
src > components > Pagination > index.tsx > ...
1 import ReactPaginate from 'react-paginate';
2 import './styles.css';
3
4 export default function Pagination() {
5
6     return (
7         <ReactPaginate
8             pageCount={10}
9             pageRangeDisplayed={3}
10            marginPagesDisplayed={1}
11            containerClassName="proj-pagination-container"
12            pageLinkClassName="proj-pagination-item"
13            breakClassName="proj-pagination-item"
14            previousClassName="proj-pagination-previous"
15            nextClassName="proj-pagination-next"
16        />
17    );
18}
```

Implementar o pagination na página Listing

```
src > pages > Listing > index.tsx ...
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
```

90						
91						
92						
93						
94						
95						
96						
97						
98						
99						
100						
101						
102						
103						
104						
105						
106						
107						
108						
109						
110						
111						

```

    <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>
        ))
      }
    </tbody>
  </table>

  <div className="proj-listing-pagination">
    <Pagination />
  </div>
</section>
```

Visualização web

19	Mátheus 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

22	Romulo Araujo	000.111.222-33	09/01/1991	R\$ 6700.5		
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

Retirar os marcadores no index.css

```
src > index.css > ...
54
55  ul {
56    list-style-type: none;
57    margin: 0;
58    padding: 0;
59  }
60
61  /*-----*/
```

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

Visualização web

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

Aplicar estilo de cor ao número da página atual

```
src > components > Pagination > index.tsx > ...
11   ...
12   |   pageLinkClassName="proj-pagination-item"
13   |   breakClassName="proj-pagination-item"
14   |   previousClassName="proj-pagination-previous"
15   |   nextClassName="proj-pagination-next"
16   |   activeLinkClassName="proj-pagination-link-active"
17   |   />
18   );
19 }
20
```

Estilizar

```
src > components > Pagination > styles.css > ...
31
32   .proj-pagination-link-active {
33     background-color: var(--proj-color-bg-secondary);
34   }
35
```

Visualização web

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						... Next

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

```
src > components > Pagination > index.tsx > ...
4  export default function Pagination() {
5
6    return (
7      <ReactPaginate
8        pageCount={10}
9        pageRangeDisplayed={3}
10       marginPagesDisplayed={1}
11       containerClassName="proj-pagination-container"
12       pageLinkClassName="proj-pagination-item"
13       breakClassName="proj-pagination-item"
14       previousClassName="proj-pagination-previous"
15       nextClassName="proj-pagination-next"
16       activeClassName="proj-pagination-link-active"
17       disableInitialCallback
18       activeClassName="proj-pagination-inactive"
19     />
20   );
21 }
22
```

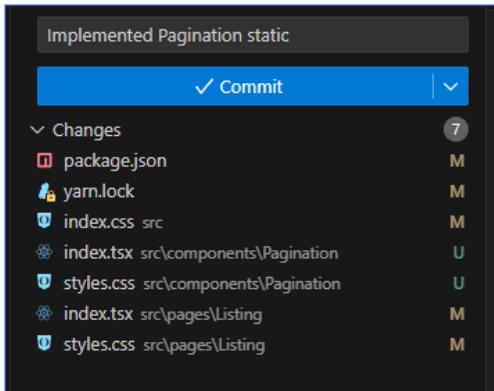
Visualização web

Previous **1** 2 3 ... 10 Next

Previous 1 **2** 3 4 ... 10 Next

Previous 1 ... 8 9 **10** Next

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 > ...
1 import ReactPaginate from 'react-paginate';
2 import './styles.css';
3
4 type Props = {
5   pageCount: number;
6   range: number;
7 }
8
9 export default function Pagination( {pageCount, range} : Props ) {
10
11   return (
12     <ReactPaginate
13       pageCount={pageCount}
14       pageRangeDisplayed={range}
15       marginPagesDisplayed={1}
16       containerClassName="proj-pagination-container"
17       pageLinkClassName="proj-pagination-item"
18       breakClassName="proj-pagination-item"
19       previousClassName="proj-pagination-previous"
20       nextClassName="proj-pagination-next"
21       activeClassName="proj-pagination-link-active"
22       disableInitialCallback
23       disabledClassName="proj-pagination-inactive"
24     />
25   );
26 }
```

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  }, [])
```

Renderizar pageCounts no Pagination, do Listing

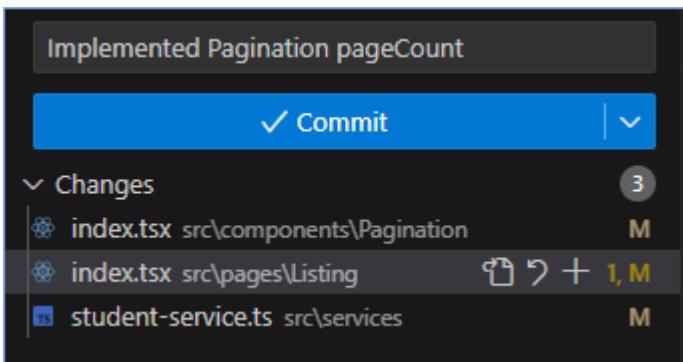
```
src > pages > Listing > index.tsx > ...
109
110      <div className="proj-listing-pagination">
111        <Pagination
112          pageCount={Number((pageCounts) ? pageCounts : 0)}
113          range={3}
114        />
115      </div>
```

Visualização web

9	Auricelio Freitas	000.111.222-33	28/08/1982	R\$ 10700.5		
10	Miguel Soares	000.111.222-33	10/02/1989	R\$ 20700.5		

Previous 1 2 3 Next

GitHub-25



Renderizar a paginação na URL

Instalar o pacote query-string

- yarn add query-string

```
SAP@DESKTOP-LSDERN6 MINGW64 /c/PROJETOS/PESSOAL/ProjCrudBackFront/frontend (main)
$ yarn add query-string
yarn add v1.22.22
[1/4] Resolving packages...
[2/4] Fetching packages...
[3/4] Linking dependencies...
[4/4] Building fresh packages...
success Saved lockfile.
success Saved 4 new dependencies.
info Direct dependencies
└─ query-string@9.1.0
info All dependencies
├─ decode-uri-component@0.4.1
├─ filter-obj@5.1.0
└─ query-string@9.1.0
└─ split-on-first@3.0.0
Done in 1.67s.
```

Nota: startar o projeto em seguida

```
auric@NOTE-SAMSUNG MINGW64 /e/DEV/PESSOAL/ProjCrudBackFront/frontend (main)
$ yarn dev
yarn run v1.22.22
$ vite

VITE v5.3.2 ready in 1836 ms

→ Local: http://localhost:5173/
→ Network: use --host to expose
```

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 > ...
+u
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  }
48
```

Implementar o useEffect para monitorar a mudança de página

```
src > pages > Listing > index.tsx > ...
51
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]);
70
```

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

A screenshot of a web browser window displaying a list of students. The URL in the address bar is `localhost:5173/listings?page=0`. The page title is **ProjCrudFront** and the section title is **Listagem de Alunos**. A blue button labeled **Novo** is visible. The table has columns: ID, NOME, CPF, NASCIMENTO, and RENDA. The data is as follows:

ID	NOME	CPF	NASCIMENTO	RENDAS
1	Auricelio Freitas	000.111.222-33	28/08/1982	R\$ 10700.5
2	Miguel Soares	000.111.222-33	10/02/1989	R\$ 20700.5
3	Matheus Marques	000.111.222-33	23/11/1992	R\$ 10700.5
4	Rafael Sales	000.111.222-33	13/10/1988	R\$ 10700.5

A screenshot of a web browser window displaying a list of students. The URL in the address bar is `localhost:5173/listings?page=1`. The page title is **ProjCrudFront** and the section title is **Listagem de Alunos**. A blue button labeled **Novo** is visible. The table has columns: ID, NOME, CPF, NASCIMENTO, and RENDA. The data is as follows:

ID	NOME	CPF	NASCIMENTO	RENDAS
11	Matheus Marques	000.111.222-33	23/11/1992	R\$ 10700.5
12	Rafael Sales	000.111.222-33	13/10/1988	R\$ 10700.5
13	Bruno Freitas	000.111.222-33	22/04/1987	R\$ 9700.5
14	Romulo Araujo	000.111.222-33	09/01/1991	R\$ 8700.5

localhost:5173/listings?page=2

ProjCrudFront

GitProject

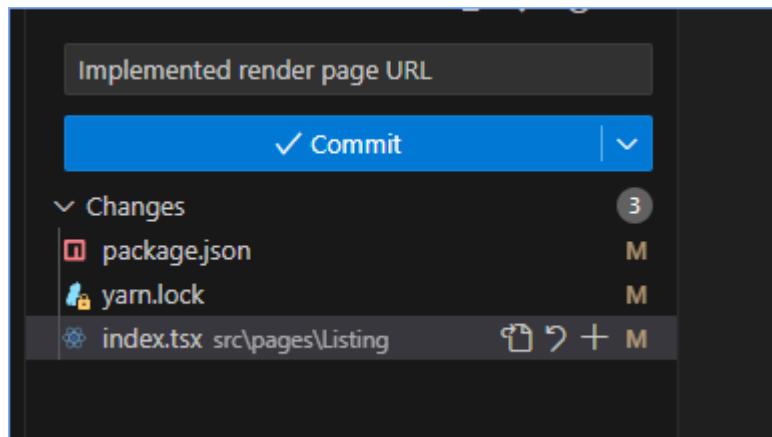
Listagem de Alunos

Novo

ID	NOME	CPF	NASCIMENTO	RENDAS		
21	Bruno Freitas	000.111.222-33	22/04/1987	R\$ 9700.5		
22	Romulo Araujo EDITADO	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		

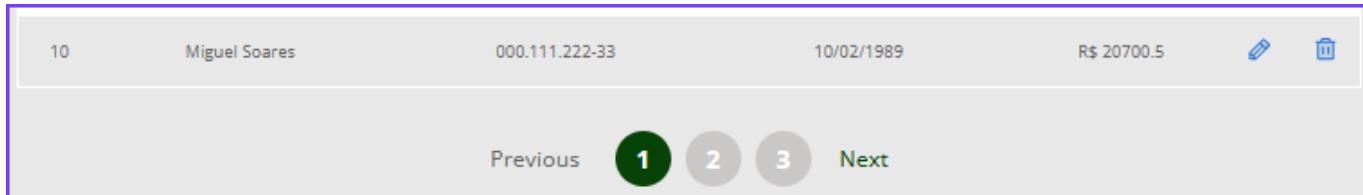
Previous 1 2 3 Next

GitHub-26



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 > ...
1  import ReactPaginate from 'react-paginate';
2  import './styles.css';
3
4  type Props = {
5      pageCount: number;
6      range: number;
7      onChange: (pageNumber: number) => void;
8  }
9
10 export default function Pagination({ pageCount, range, onChange }: Props) {
11     return (
12         <ReactPaginate
13             pageCount={pageCount}
14             pageRangeDisplayed={range}
15             marginPagesDisplayed={1}
16             containerClassName="proj-pagination-container"
17             pageLinkClassName="proj-pagination-item"
18             breakClassName="proj-pagination-item"
19             previousClassName="proj-pagination-previous"
20             nextClassName="proj-pagination-next"
21             activeClassName="proj-pagination-link-active"
22             disableInitialCallback
23             disabledClassName="proj-pagination-inactive"
24             onPageChange={(items) => onChange(items.selected)}>
25         />
26     );
27 }
```

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  }, [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

localhost:5173/listings?page=0

ProjCrudFront GitProject

Listagem de Alunos

[Novo](#)

ID	NOME	CPF	NASCIMENTO	RENDA	ACTION	ACTION
1	Auricelio Freitas	000.111.222-33	28/08/1982	R\$ 10700.5	Edit	Delete
2	Miguel Soares	000.111.222-33	10/02/1989	R\$ 20700.5	Edit	Delete
3	Matheus Marques	000.111.222-33	23/11/1992	R\$ 10700.5	Edit	Delete
4	Rafael Sales	000.111.222-33	13/10/1988	R\$ 10700.5	Edit	Delete
5	Bruno Freitas	000.111.222-33	22/04/1987	R\$ 9700.5	Edit	Delete
6	Romulo Araujo	000.111.222-33	09/01/1991	R\$ 8700.5	Edit	Delete
7	Ryan Sousa	000.111.222-33	26/03/1994	R\$ 7700.5	Edit	Delete
8	test1	000.111.222-33	32/12/1969	R\$ 1024	Edit	Delete
9	Auricelio Freitas	000.111.222-33	28/08/1982	R\$ 10700.5	Edit	Delete
10	Miguel Soares	000.111.222-33	10/02/1989	R\$ 20700.5	Edit	Delete

Previous 1 2 3 Next

localhost:5173/listings?page=1

ProjCrudFront GitProject

Listagem de Alunos

[Novo](#)

ID	NOME	CPF	NASCIMENTO	RENDA	ACTION	ACTION
11	Matheus Marques	000.111.222-33	23/11/1992	R\$ 10700.5	Edit	Delete
12	Rafael Sales	000.111.222-33	13/10/1988	R\$ 10700.5	Edit	Delete
13	Bruno Freitas	000.111.222-33	22/04/1987	R\$ 9700.5	Edit	Delete
14	Romulo Araujo	000.111.222-33	09/01/1991	R\$ 8700.5	Edit	Delete
15	Ryan Sousa	000.111.222-33	26/03/1994	R\$ 7700.5	Edit	Delete
16	test1	000.111.222-33	32/12/1969	R\$ 1024	Edit	Delete
17	Auricelio Freitas	000.111.222-33	28/08/1982	R\$ 10700.5	Edit	Delete
18	Miguel Soares	000.111.222-33	10/02/1989	R\$ 20700.5	Edit	Delete
19	Matheus Marques	000.111.222-33	23/11/1992	R\$ 10700.5	Edit	Delete
20	Rafael Sales	000.111.222-33	13/10/1988	R\$ 10700.5	Edit	Delete

Previous 1 2 3 Next

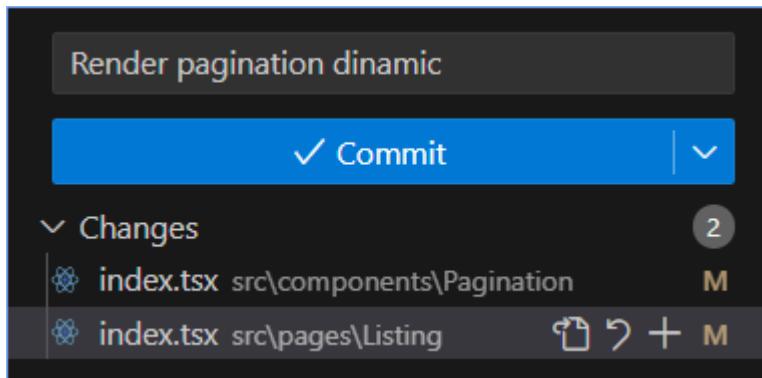
A screenshot of a web application titled "ProjCrudFront" at the URL "localhost:5173/listings?page=2". The page displays a table of student data with columns: ID, NOME, CPF, NASCIMENTO, and RENDA. Each row includes edit and delete icons. The table data is as follows:

ID	NOME	CPF	NASCIMENTO	RENDAS
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

Pagination controls at the bottom show "Previous" and "Next" buttons, with the number "3" indicating the current page.

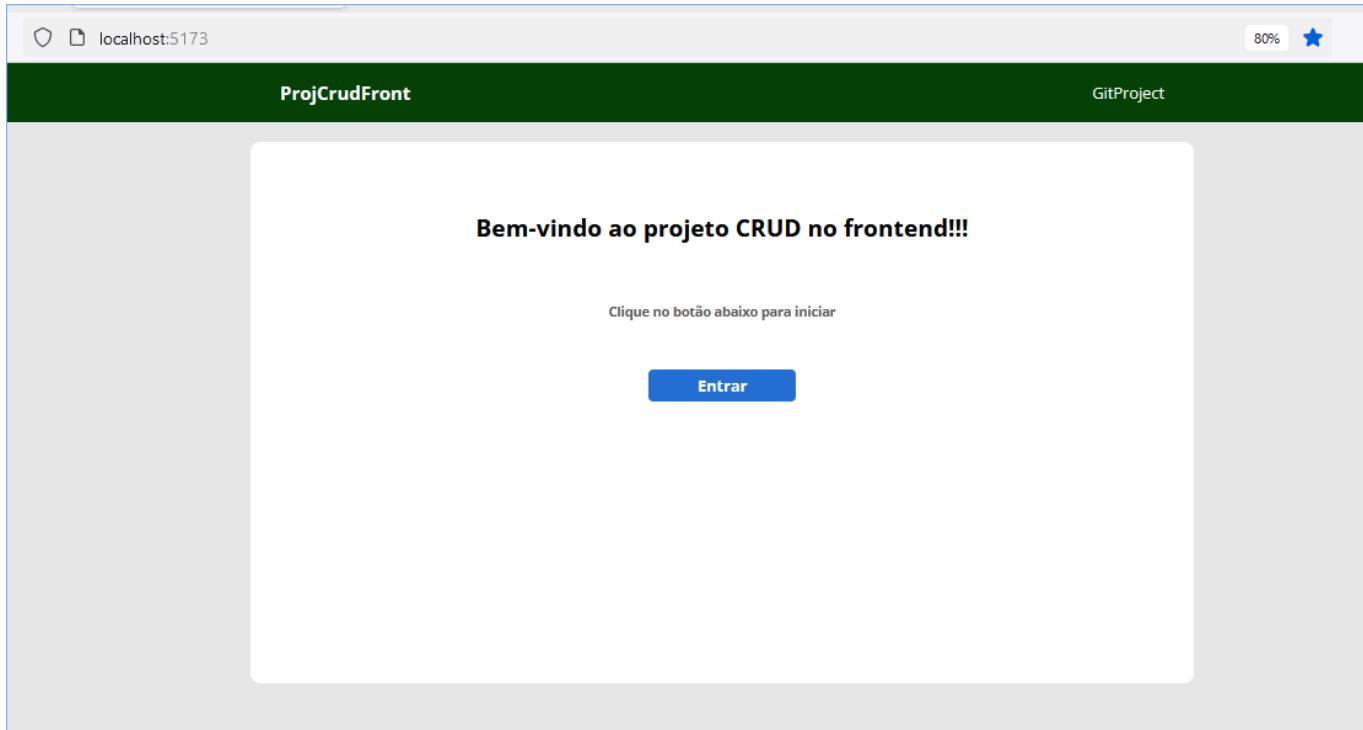
NOTA: Neste momento o nosso componente já está passando as páginas ao clicar.

GitHub-27



SISTEMA FINALIZADO

Home



Listagem de Alunos

localhost:5173/listings

ProjCrudFront

GitProject

Listagem de Alunos

Novo

ID	NOME	CPF	NASCIMENTO	RENDAS
1	Auricelio Freitas	000.111.222-33	28/08/1982	R\$ 10700.5
2	Miguel Soares	000.111.222-33	10/02/1989	R\$ 20700.5
3	Matheus Marques	000.111.222-33	23/11/1992	R\$ 10700.5
4	Rafael Sales	000.111.222-33	13/10/1988	R\$ 10700.5
5	Bruno Freitas	000.111.222-33	22/04/1987	R\$ 9700.5
6	Romulo Araujo	000.111.222-33	09/01/1991	R\$ 8700.5
7	Ryan Sousa	000.111.222-33	26/03/1994	R\$ 7700.5
8	test1	000.111.222-33	32/12/1969	R\$ 1024
9	Auricelio Freitas	000.111.222-33	28/08/1982	R\$ 10700.5
10	Miguel Soares	000.111.222-33	10/02/1989	R\$ 20700.5

Previous 1 2 3 Next

Cadastro de alunos

The screenshot shows a web application interface for student registration. At the top, a header bar displays the title "ProjCrudFront" and a link "GitProject". The main content area is titled "DADOS DO ALUNO". It contains four input fields with validation messages:

- Nome:** A red-bordered input field containing "Nome". Below it, a message says "Favor informar um nome entre 3 e 50 caracteres".
- Cpf:** A red-bordered input field containing "CPF". Below it, a message says "Favor informar um CPF válido".
- Data de Nascimento:** A red-bordered input field containing "dd / mm / aaaa". To its right is a calendar icon. Below it, a message says "Favor informar uma data válida".
- Renda:** A red-bordered input field containing "Salário". Below it, a message says "Favor informar um valor positivo".

At the bottom of the form are two buttons: a red "Cancelar" button and a blue "Salvar" button.

NOTA: Formulário implementado com validação em todos os campos.

Deletar Aluno

The screenshot shows a web browser window with the URL `localhost:5173/listings`. The title bar says "ProjCrudFront" and there's a "GitProject" button. The main content is titled "Listagem de Alunos". A "Novo" button is at the top left of the table. The table has columns: ID, NOME, CPF, NASCIMENTO, and RENDA. It contains 10 rows of student data. In the 5th row, a delete icon triggers a modal dialog box. The dialog box has a white background and a dark border. It contains the text "Tem certeza?" (Are you sure?) in bold. There are two buttons: a red "Não" (No) button and a blue "Sim" (Yes) button. The "Sim" button is highlighted with a darker shade.

ID	NOME	CPF	NASCIMENTO	RENDAS
1	Auricelio Freitas	000.111.222-33	28/08/1982	R\$ 10700.5
2	Miguel Soares	000.111.222-33	10/02/1989	R\$ 20700.5
3	Matheus Marques	000.111.222-33	23/11/1992	R\$ 10700.5
4	Rafael Sales			R\$ 10700.5
5	Bruno Freitas			R\$ 9700.5
6	Romulo Araujo			R\$ 8700.5
7	Ryan Sousa	000.111.222-33	26/03/1994	R\$ 7700.5
8	test1	000.111.222-33	32/12/1969	R\$ 1024
9	Auricelio Freitas	000.111.222-33	28/08/1982	R\$ 10700.5
10	Miguel Soares	000.111.222-33	10/02/1989	R\$ 20700.5

Previous 1 2 3 Next

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

FIM