**Curso de React**

**Matheus Battisti (Hora de Codar)**

# Aula 01 - Introdução

## O que é React?

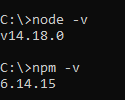
* Uma biblioteca JS para criação de interfaces;
* Utilizado para construir SPAs (Single Page Application);
* Baseado em componentes;
* Utiliza o JSX para renderizar HTML;
* E aplica o Virtual DOM para realizar as alterações de DOM;
* Podemos adicionar a um projeto ou criar um projeto com ele;

# Aula 02 - Instalando o React

## Como instalar o React?

* Para instalar o React vamos utilizar uma ferramenta chamada Create React App;
* Recebemos todos os arquivos da biblioteca e temos como executá-la;
* Para utilizar precisamos do Node e também npm;
* Esta ferramenta também otimiza o app gerado pela produção;
* É possível iniciar a aplicação com npm start

- Instale o Node e o Visual Studio Code



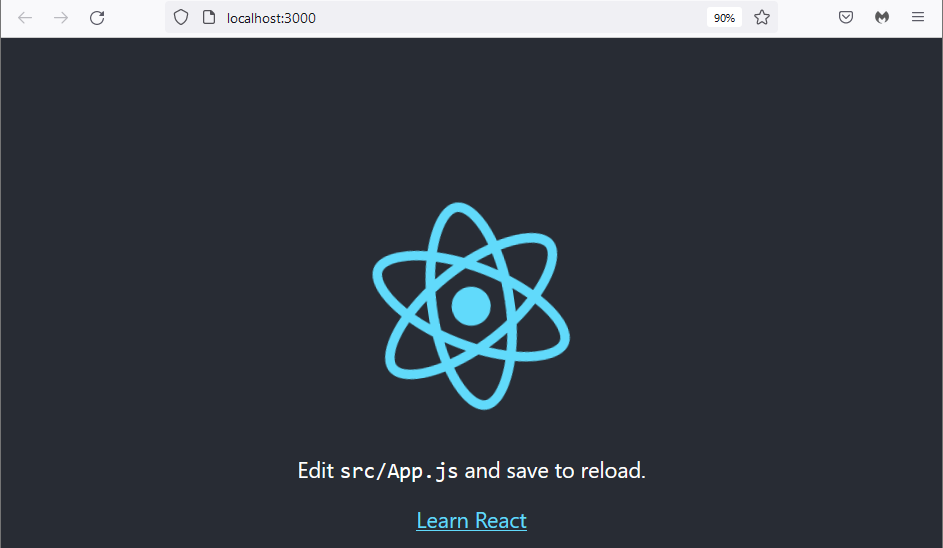
npx create-react-app my-app

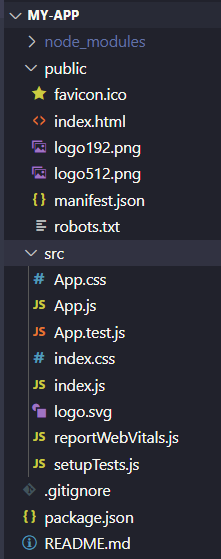
cd my-app

code .

npm start

**http://localhost:3000/**





**src\App.js**

import './App.css';

function App() {

return (

<div className="App">

<h1>Olá React!</h1>

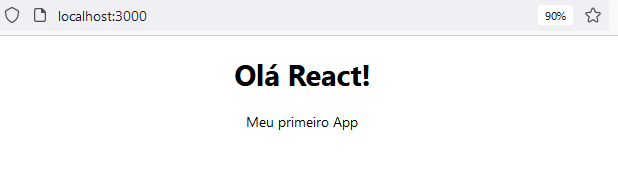
<p>Meu primeiro App</p>

</div>

);

}

export default App;



# Aula 03 - Entendendo o JSX

* O JSX é como um HTML, porém dentro do código JavaScript;
* É a principal maneira de escrever HTML com o React;
* Podemos interpolar variáveis, inserindo ela entre {};
* É possível também executar funções em JSX;
* Inserir valores em atributos de tags também é válido em JSX;

**src\App.js**

import './App.css';

function App() {

const name = "Matheus"

const newName = name.toUpperCase()

function sum(a,b){

return a+b

}

const url = "https://via.placeholder.com/150/0000FF/FFFFFF?Text=Digital.com"

return (

<div className="App">

<h2>Alterando o JSX</h2>

<p>Olá, {newName}!</p>

<p>Soma: {sum(2,3)} </p>

<img src={url} alt="Minha imagem" />

</div>

);

}

export default App;



# Aula 04 - Criando componentes no React

## Componentes

* Permitem dividir a aplicação em partes;
* Os componentes renderizam JSX, assim como App.js (que é um componente);
* Precisamos criar um arquivo de componente;
* E importá-lo onde precisamos utilizar;
* Normalmente ficam em uma pasta chamada components;

**src\components\Frase.js**

function Frase(){

return (

<div>

<p>Este é um componente com uma frase</p>

</div>

)

}

export default Frase

**src\components\HelloWorld.js**

import Frase from './Frase'

function HelloWorld(){

return (

<div>

<Frase />

<h1>Meu primeiro componente</h1>

<Frase />

</div>

)

}

export default HelloWorld

**src\App.js**

import './App.css'

import HelloWorld from './components/HelloWorld'

function App() {

const name = "Matheus"

const newName = name.toUpperCase()

function sum(a,b){

return a+b

}

const url = "https://via.placeholder.com/150/0000FF/FFFFFF?Text=Digital.com"

return (

<div className="App">

<h2>Alterando o JSX</h2>

<p>Olá, {newName}!</p>

<p>Soma: {sum(2,3)} </p>

<img src={url} alt="Minha imagem" />

<HelloWorld/>

</div>

);

}

export default App;



# Aula 05 - Trabalhando com props

* As props são valores passados para componentes;
* Podemos deixá-los dinâmicos;
* Ou seja, mudando a execução por causa do valor da prop;
* O valor é passado como um atributo na chamada do componente;
* E precisa ser resgatado dentro de uma propriedade/argumento chamada props na função de definição do componente;
* As props são somente de leitura!

**src\components\SayMyName.js**

function SayMyName(props){

return (

<div>

<p>Fala aí, {props.name}, suave?</p>

</div>

)

}

export default SayMyName

**src\App.js**

import './App.css';

import HelloWorld from './components/HelloWorld';

import SayMyName from './components/SayMyName';

function App() {

const name = "Maria"

return (

<div className="App">

<HelloWorld/>

<SayMyName name="Matheus" />

<SayMyName name="João" />

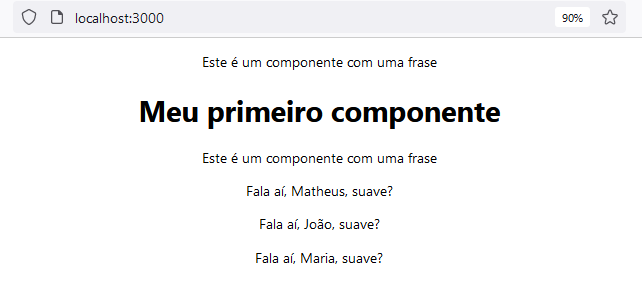
<SayMyName name={name} />

</div>

);

}

export default App;



**src\components\Pessoa.js**

function Pessoa(props){

return(

<div>

<img src={props.foto} alt={props.nome} />

<h2>Nome: {props.nome}</h2>

<p>Idade: {props.idade}</p>

<p>Profissão: {props.profissao}</p>

</div>

)

}

export default Pessoa

**src\App.js**

import './App.css';

import HelloWorld from './components/HelloWorld';

import Pessoa from './components/Pessoa';

import SayMyName from './components/SayMyName';

function App() {

const name = "Maria"

return (

<div className="App">

<HelloWorld/>

<SayMyName name="Matheus" />

<SayMyName name="João" />

<SayMyName name={name} />

<Pessoa

nome="Rodrigo"

idade="28"

profissao="programador"

foto="https://via.placeholder.com/150"

/>

</div>

);

}

export default App;



**src\components\Pessoa.js**

function Pessoa({nome, idade, profissao, foto}){

return(

<div>

<img src={foto} alt={nome} />

<h2>Nome: {nome}</h2>

<p>Idade: {idade}</p>

<p>Profissão: {profissao}</p>

</div>

)

}

export default Pessoa



# Aula 06 - Inserindo CSS no React

* O CSS pode ser adicionado de forma global na aplicação, por meio do arquivo index.css por exemplo;
* Porém é possível estiliizar a nível de componentes;
* Utilizamos o CSS modules para isso;
* Basta criar um arquivo como: Componente.module.css;
* E chamar este CSS no componente;

## Usando CSS de forma global

**src\App.js**

import './App.css';

import HelloWorld from './components/HelloWorld';

import Pessoa from './components/Pessoa';

import SayMyName from './components/SayMyName';

function App() {

const name = "Maria"

return (

<div className="App">

<h1>Testando CSS</h1>

<HelloWorld/>

<SayMyName name="Matheus" />

<SayMyName name="João" />

<SayMyName name={name} />

<Pessoa

nome="Rodrigo"

idade="28"

profissao="programador"

foto="https://via.placeholder.com/150"

/>

</div>

);

}

export default App;

**src\index.css**

body {

padding: 50px;

background-color: #efefef;

}

h1{

color: #333;

}

p{

background-color: #999;

padding: 10px;

}



## Utilizando CSS a nível de componente

**src\components\Frase.module.css**

.fraseContainer{

background-color: #333;

border: 1px solid #111;

}

.fraseContent{

color: #fff;

background-color: #333;

margin: 0;

}

**src\components\Frase.js**

import styles from './Frase.module.css'

function Frase(){

return (

<div className={styles.fraseContainer}>

<p className={styles.fraseContent}>Este é um componente com uma frase</p>

</div>

)

}

export default Frase

**src\App.js**

import './App.css';

import Frase from './components/Frase';

import Pessoa from './components/Pessoa';

import SayMyName from './components/SayMyName';

function App() {

const name = "Maria"

return (

<div className="App">

<h1>Testando CSS</h1>

<Frase />

<Frase />

<SayMyName name="Matheus" />

<SayMyName name="João" />

<SayMyName name={name} />

<Pessoa

nome="Rodrigo"

idade="28"

profissao="programador"

foto="https://via.placeholder.com/150"

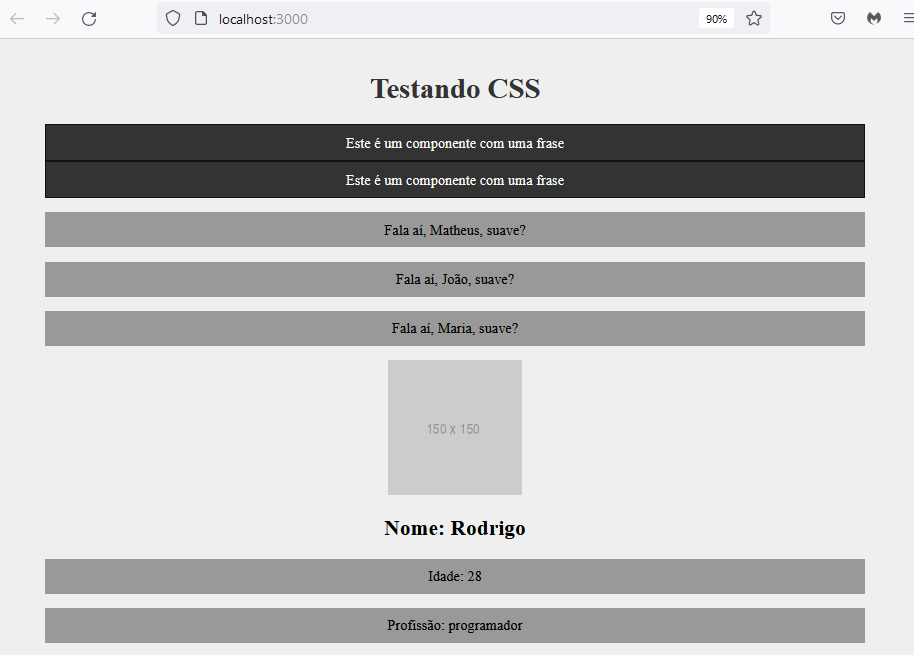
/>

</div>

);

}

export default App;



# Aula 07 - Utilizando React fragments

* Os React Fragments permitem a criação de um componente sem elemento pai;
* O propósito é descomplicar os nós do DOM;
* A sintaxe é <> e </>, não há um nome para a tag;
* Criamos no próprio JSX.

**src\components\Item.js**

function Item(props){

return (

<>

<li>{props.marca}</li>

</>

)

}

export default Item

**src\components\List.js**

import Item from './Item'

function List(){

return (

<>

<h1>Minha lista</h1>

<ul>

<Item marca="Ferrari" />

<Item marca ="Fiat" />

<Item marca="Renault" />

</ul>

</>

)

}

export default List

**src\App.js**

import './App.css';

import Frase from './components/Frase';

import List from './components/List';

import Pessoa from './components/Pessoa';

import SayMyName from './components/SayMyName';

function App() {

const name = "Maria"

return (

<div className="App">

<h1>Testando CSS</h1>

<Frase />

<Frase />

<SayMyName name="Matheus" />

<SayMyName name="João" />

<SayMyName name={name} />

<Pessoa

nome="Rodrigo"

idade="28"

profissao="programador"

foto="https://via.placeholder.com/150"

/>

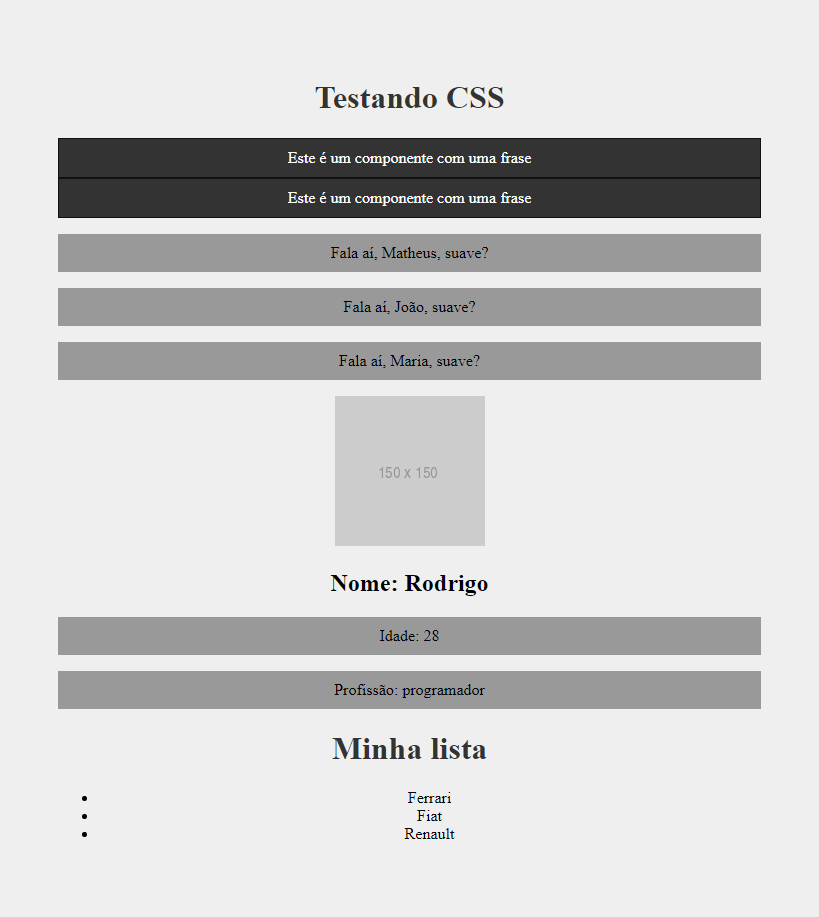
<List />

</div>

);

}

export default App;



# Aula 08 - Avançando em props

* Podemos definir tipos para as props, realizando uma espécie de validação;
* Definimos em um objeto chamado propTypes no próprio componente;
* E ainda há possibilidade de definir um valor padrão;
* Neste caso utilizamos o objeto defaultProps;

**src\components\List.js**

import Item from './Item'

function List(){

return (

<>

<h1>Minha lista</h1>

<ul>

<Item marca="Ferrari" ano\_lancamento={1985} />

<Item marca ="Fiat" ano\_lancamento={1964} />

<Item marca="Renault" />

<Item marca="Chevrolet" ano\_lancamento={1999} />

<Item />

</ul>

</>

)

}

export default List

**src\components\Item.js**

import PropTypes from "prop-types"

function Item({marca, ano\_lancamento}){

return (

<>

<li>{marca} - {ano\_lancamento}</li>

</>

)

}

Item.propTypes = {

marca: PropTypes.string.isRequired,

ano\_lancamento: PropTypes.number,

}

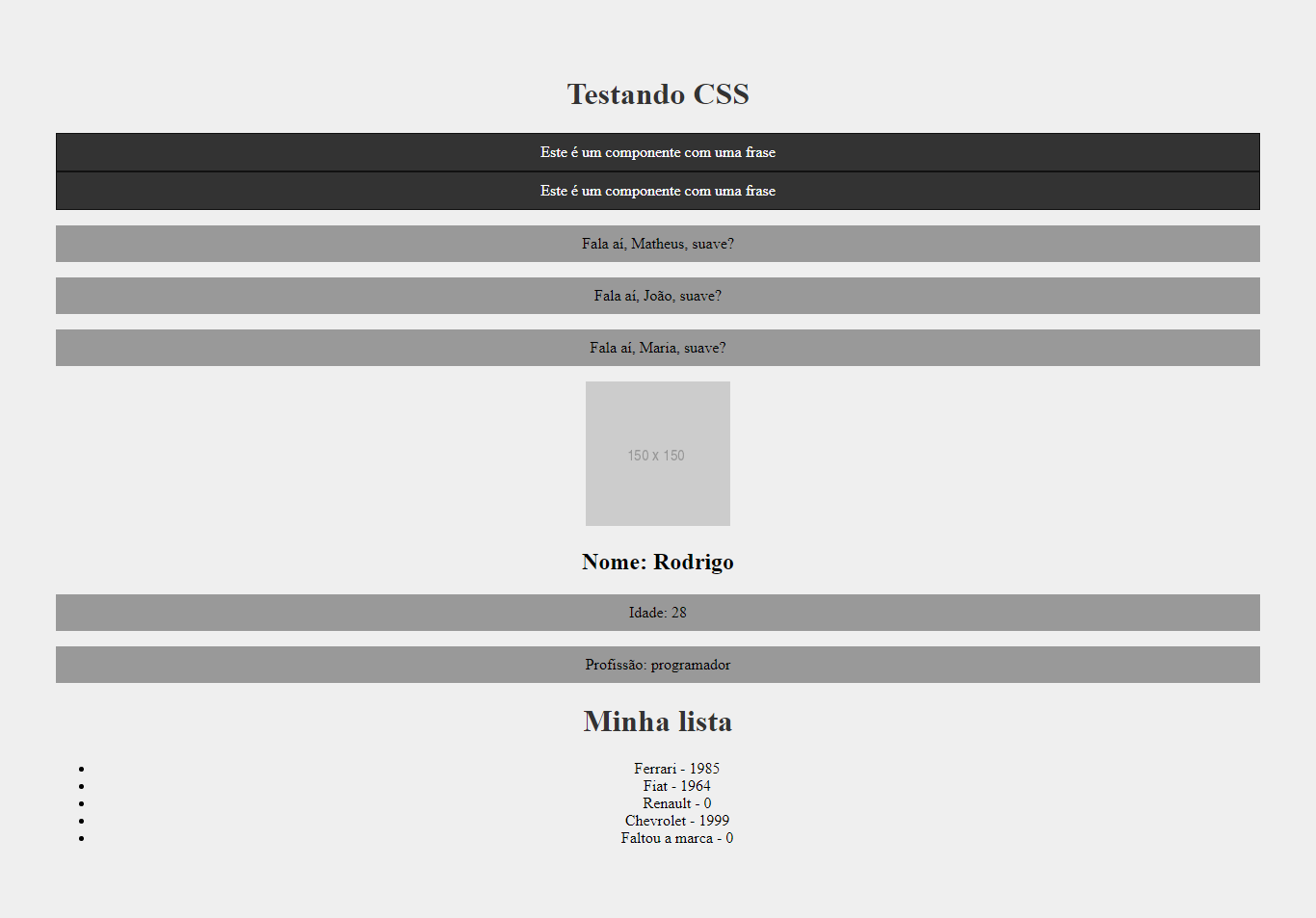
Item.defaultProps = {

marca: 'Faltou a marca',

ano\_lancamento: 0,

}

export default Item



# Aula 09 - Eventos no React

* Os eventos de React são os mesmos eventos do DOM;
* Ou seja, temos eventos para responder a um click;
* O evento é atrelado a uma tag que irá executá-lo;
* Geralmente um método é atribuído ao evento;
* Este método deve ser criado no componente;

**src\components\Evento.js**

function Evento({numero}) {

function meuEvento() {

console.log(`Opa, fui ativado! ${numero}`)

}

return (

<div>

<p>Clique para disparar o evento:</p>

<button onClick={meuEvento}>Ativar!</button>

</div>

)

}

export default Evento

**src\App.js**

import './App.css';

import Evento from './components/Evento';

function App() {

return (

<div className="App">

<h1>Testando Eventos</h1>

<Evento numero="1" />

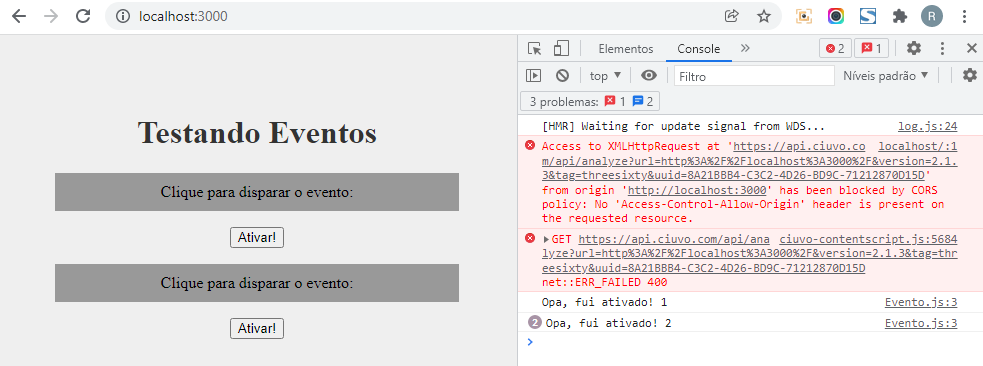
<Evento numero="2" />

</div>

);

}

export default App;



**src\components\Form.js**

function Form(){

function cadastrarUsuario(e){

e.preventDefault()

console.log("Cadastrou o usuário!")

}

return(

<div>

<h1>Meu cadastro</h1>

<form onSubmit={cadastrarUsuario}>

<div>

<input type="text" placeholder="Digite o seu nome" />

<input type="submit" value="Cadastrar" />

</div>

</form>

</div>

)

}

export default Form

**src\App.js**

import './App.css';

import Evento from './components/Evento';

import Form from './components/Form';

function App() {

return (

<div className="App">

<h1>Testando Eventos</h1>

<Evento numero="1" />

<Evento numero="2" />

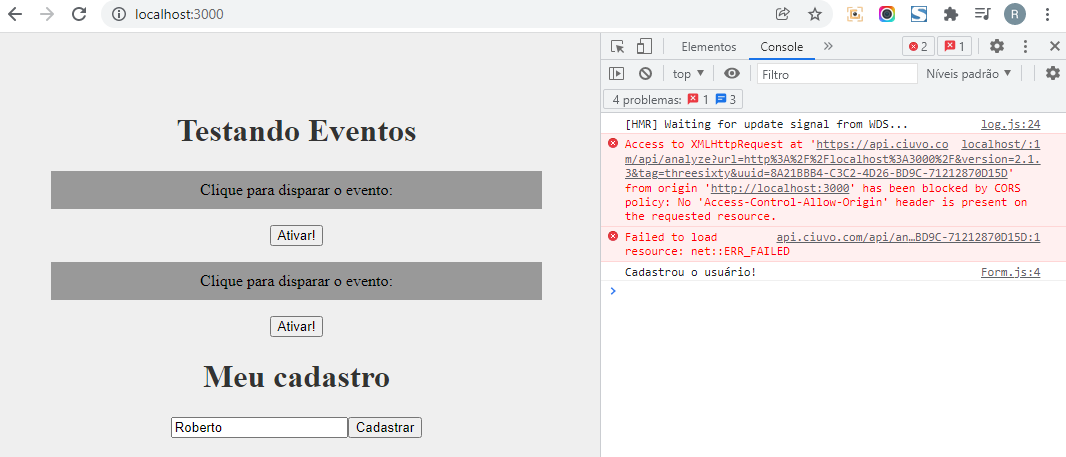
<Form />

</div>

);

}

export default App;



# Aula 10 - useState na prática

* O useState é um hook do React;
* Com ele conseguimos manusear o estado de um componente de forma simples;
* Este hook funciona muito bem com eventos;
* Podemos atrelar um evento a mudança de state;

**src\components\Form.js**

import { useState } from 'react';

function Form(){

function cadastrarUsuario(e){

e.preventDefault()

console.log(name)

console.log(`O usuário ${name} foi cadastrado com a senha ${password}`)

}

const [name, setName] = useState();

const [password, setPassword] = useState();

return(

<div>

<h1>Meu cadastro</h1>

<form onSubmit={cadastrarUsuario}>

<div>

<div>

<label htmlFor="name">Nome: </label>

<input

type="text"

id="name"

name="name"

placeholder="Digite o seu nome"

onChange={(e) => setName(e.target.value)}

/>

</div>

<div>

<label htmlFor="password">Senha: </label>

<input

type="password"

id="password"

name="password"

placeholder="Digite a sua senha"

onChange={(e) => setPassword(e.target.value)}

/>

</div>

<input type="submit" value="Cadastrar" />

</div>

</form>

</div>

)

}

export default Form

**src\App.js**

import './App.css';

import Evento from './components/Evento';

import Form from './components/Form';

function App() {

return (

<div className="App">

<h1>Testando Eventos</h1>

<Evento numero="1" />

<Evento numero="2" />

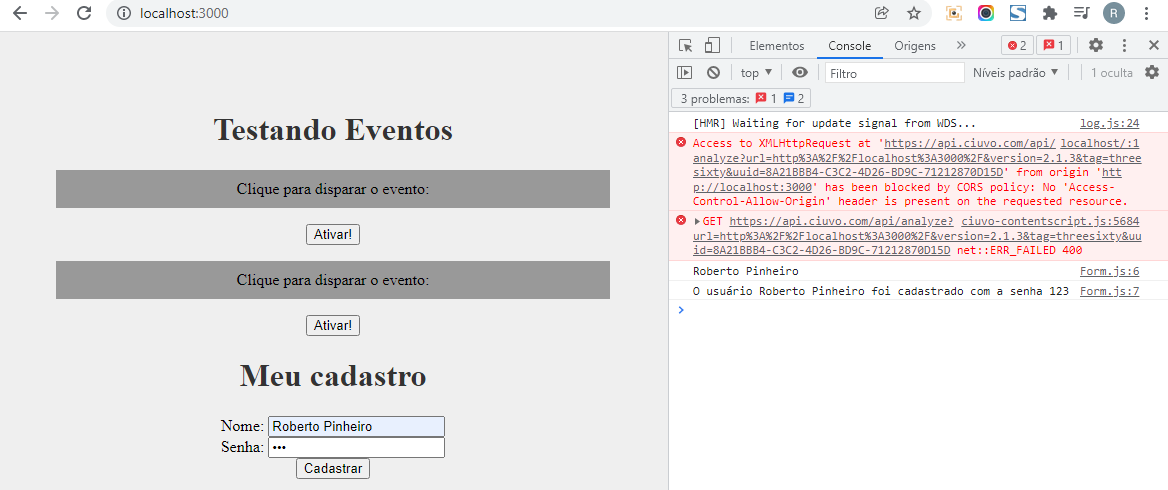
<Form />

</div>

);

}

export default App;



# Aula 11 - Passar eventos por props

* Os métodos também podem ser passados por props;
* Ou seja, um componente filho pode ativar o método do seu ancestral;
* Vamos acessar o método por meio de um evento;
* A sintaxe é a mesma de uma props de dados: props.meuEvento;

**src\components\evento\Button.js**

function Button(props){

return(

<button onClick={props.event}>{props.text}</button>

)

}

export default Button

**src\components\Evento.js**

import Button from './evento/Button'

function Evento() {

function meuEvento() {

console.log(`Ativando primeiro evento!`)

}

function segundoEvento(){

console.log("Ativando o segundo evento!")

}

return (

<div>

<p>Clique para disparar o evento:</p>

<Button event={meuEvento} text="Primeiro evento" />

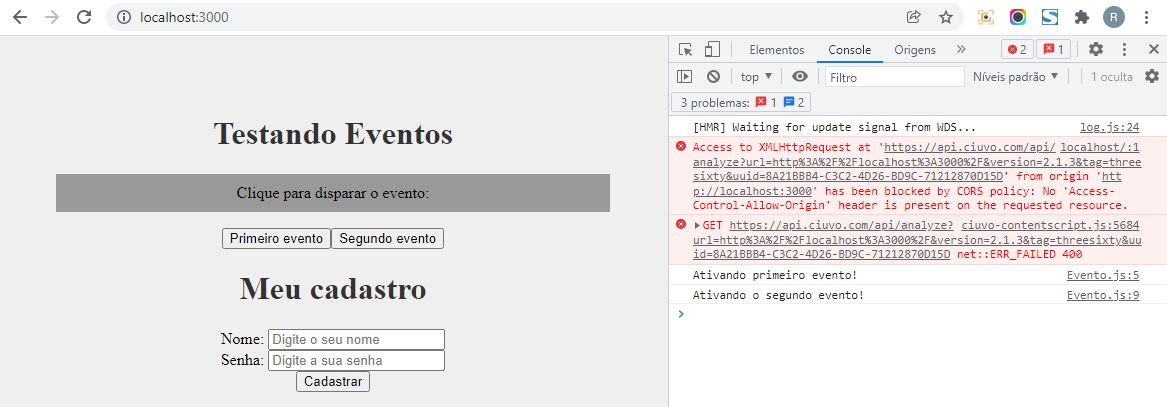
<Button event={segundoEvento} text="Segundo evento" />

</div>

)

}

export default Evento



# Aula 12 - Renderização condicional (if)

* Podemos atrelar a exibição de algum elemento a um if;
* Esta ação é chamada de renderização condicional;
* Envolvemos as tags em chaves {};
* Como as chaves executam JavaScript, criamos nossa condição;
* É possível usar o state para criar as condições;

**src\components\Condicional.js**

import { useState } from 'react'

function Condicional(){

const [email, setEmail] = useState()

const [userEmail, setUserEmail] = useState()

function enviarEmail(e){

e.preventDefault()

setUserEmail(email)

console.log(userEmail)

}

function limparEmail() {

setUserEmail('')

}

return (

<div>

<h2>Cadastre o seu e-mail:</h2>

<form>

<input type="email" placeholder="Digite o seu e-mail..." onChange={(e) => setEmail(e.target.value)} />

<button type="submit" onClick={enviarEmail}>Enviar</button>

{userEmail && (

<div>

<p>O e-mail do usuário é: {userEmail}</p>

<button onClick={limparEmail}>Limpar e-mail</button>

</div>

)}

</form>

</div>

)

}

export default Condicional

**src\App.js**

import './App.css';

import Condicional from './components/Condicional';

function App() {

return (

<div className="App">

<h1>Renderização condicional</h1>

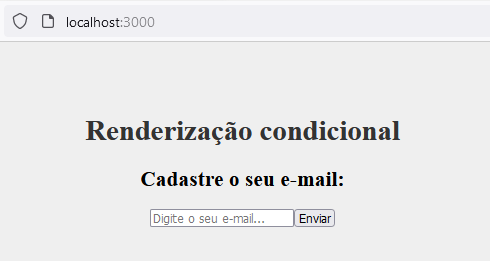
<Condicional />

</div>

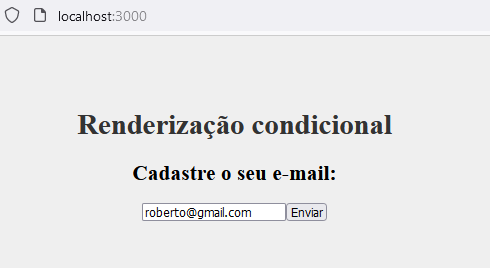
);

}

export default App;







# Aula 13 - Renderização de listas

* Para renderizar uma lista vamos primeiramente precisar de um array;
* Depois utilizamos a função map, para percorrer cada um dos itens;
* Podendo assim renderizar algo na tela;
* É possível unir operadores condicionais com a renderização de listas;

**src\components\OutraLista.js**

function OutraLista({itens}){

return (

<>

<h3>Lista de bibliotecas front-end</h3>

{

itens.length > 0 ? (

itens.map((item, index) => (

<p key={index}>{item}</p>

))

) : (

<p>Não há itens na lista!</p>

)

}

</>

)

}

export default OutraLista

**src\App.js**

import './App.css';

import OutraLista from './components/OutraLista';

function App() {

const meusItens = ['React', 'Vue', 'Angular']

return (

<div className="App">

<h1>Renderização de listas</h1>

<OutraLista itens={meusItens} />

</div>

);

}

export default App;



# Aula 14 - State Lift

* State Lift é uma técnica utilizada para compartilhar o state;
* É normal vários componentes dependerem do mesmo estado;
* Então precisamos elevar o nível do mesmo a um componente pai;
* Então centralizamos o state no pai, e definimos quem usa e quem define (setState);

**src\components\Saudacao.js**

function Saudacao({nome}){

function gerarSaudacao(algumNome){

return `Olá ${algumNome}! Tudo bem?`

}

return (

<>

{nome && <p>{gerarSaudacao(nome)}</p>}

</>

)

}

export default Saudacao

**src\components\SeuNome.js**

function SeuNome({setNome}){

return(

<div>

<p>Digite o seu nome:</p>

<input type="text" placeholder="Qual é o seu nome?" onChange={(e) => setNome(e.target.value)} />

</div>

)

}

export default SeuNome

**src\App.js**

import { useState } from 'react';

import './App.css';

import Saudacao from './components/Saudacao';

import SeuNome from './components/SeuNome';

function App() {

const [nome, setNome] = useState()

return (

<div className="App">

<h1>State Lift</h1>

<SeuNome setNome={setNome} />

<Saudacao nome={nome} />

</div>

);

}

export default App;





# Aula 15 - Implementando o React Router

* O React Router é um pacote para mudança de URLs da aplicação;
* Podemos assim acessar outras views, sem o page reload;
* Trocando apenas uma parte do layout da aplicação, ou seja, o que muda de view para view;
* Precisamos instalar este pacote no projeto;
* E também realizar algumas mudanças em como o App é estruturado;

## Instalando o pacote React-Router

npm install react-router-dom

**src\pages\Home.js**

function Home(){

return(

<div>

<h1>Home</h1>

<p>Conteúdo da página</p>

</div>

)

}

export default Home

**src\pages\Empresa.js**

function Empresa() {

return (

<div>

<h1>Empresa</h1>

<p>Conteúdo da página</p>

</div>

)

}

export default Empresa

**src\pages\Contato.js**

function Contato() {

return (

<div>

<h1>Contato</h1>

<p>Conteúdo da página</p>

</div>

)

}

export default Contato

**src\App.js**

import { BrowserRouter as Router, Link, Route, Routes } from 'react-router-dom';

import Contato from './pages/Contato';

import Empresa from './pages/Empresa';

import Home from './pages/Home';

function App() {

return (

<Router>

<ul>

<li><Link to="/">Home</Link></li>

<li><Link to="/empresa">Empresa</Link></li>

<li><Link to="/contato">Contato</Link></li>

</ul>

<Routes>

<Route exact path="/" element={<Home />} />

<Route path="/contato" element={<Contato />} />

<Route path="/empresa" element={<Empresa />} />

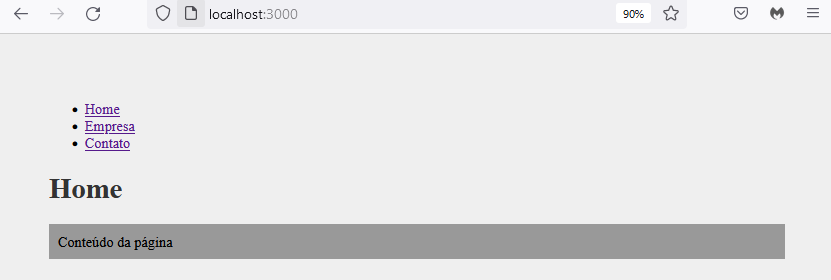
</Routes>

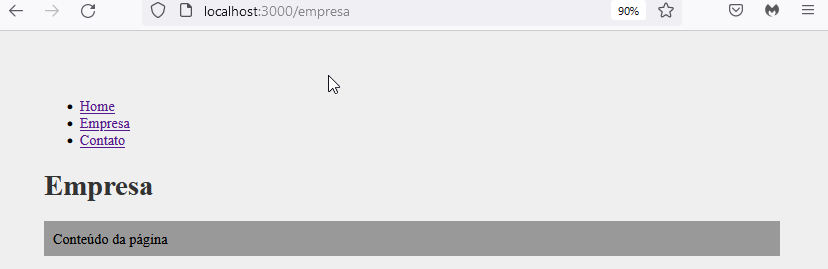
</Router>

)

}

export default App;





**src\components\layout\Navbar.js**

import { Link } from 'react-router-dom'

import styles from './Navbar.module.css'

function Navbar(){

return(

<div>

<ul className={styles.list}>

<li className={styles.item}><Link to="/">Home</Link></li>

<li className={styles.item}><Link to="/empresa">Empresa</Link></li>

<li className={styles.item}><Link to="/contato">Contato</Link></li>

</ul>

</div>

)

}

export default Navbar

**src\components\layout\Navbar.module.css**

.list{

display: flex;

list-style: none;

}

.item{

margin-right: 1em;

}

**src\components\layout\Footer.js**

function Footer(){

return(

<footer>

Rodapé

</footer>

)

}

export default Footer

**src\App.js**

import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';

import Footer from './components/layout/Footer';

import Navbar from './components/layout/Navbar';

import Contato from './pages/Contato';

import Empresa from './pages/Empresa';

import Home from './pages/Home';

function App() {

return (

<Router>

<Navbar />

<Routes>

<Route exact path="/" element={<Home />} />

<Route path="/contato" element={<Contato />} />

<Route path="/empresa" element={<Empresa />} />

</Routes>

<Footer />

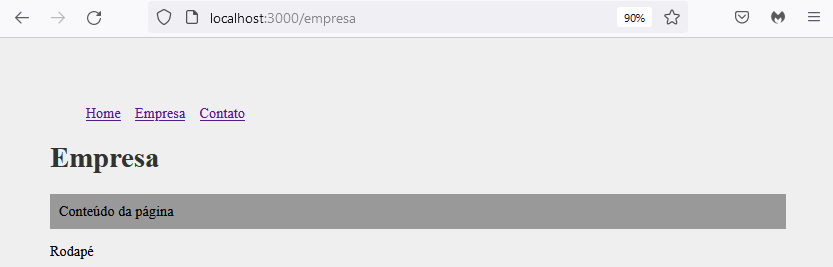
</Router>

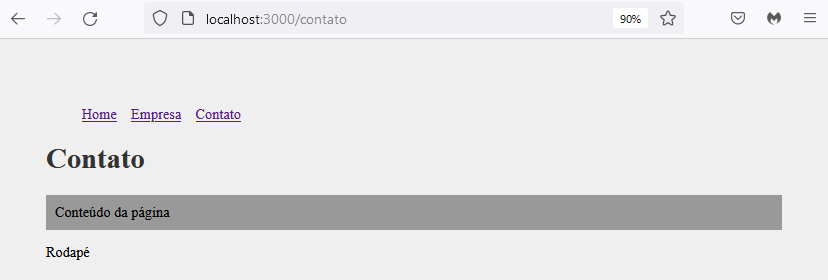
)

}

export default App;







# Aula 16 - Icons

## React Icons

* O React Icons é um pacote de ícones externo;
* Precisamos adicionar ao projeto através do npm;
* Ele nos permite adicionar ícones ao projeto com uma sintaxe parecida a de componentes;
* Além disso há uma grande quantidade de ícones disponíveis;

https://react-icons.github.io/react-icons/

npm install react-icons --save

**src\components\layout\Footer.js**

import { FaFacebook, FaInstagram, FaLinkedin } from 'react-icons/fa'

import styles from './Footer.module.css'

function Footer(){

return(

<footer>

<ul className={styles.social\_list}>

<li><FaFacebook /></li>

<li><FaInstagram /></li>

<li><FaLinkedin /></li>

</ul>

<p>Nosso rodapé</p>

</footer>

)

}

export default Footer

**src\components\layout\Footer.module.css**

.social\_list{

display: flex;

justify-content: center;

align-items: center;

list-style: none;

}

.social\_list li{

margin: 0 1em;

}

.social\_list svg{

font-size: 2em;

cursor: pointer;

color: red;

}



# Aula 17 - Criando o projeto

npx create-react-app costs

cd costs

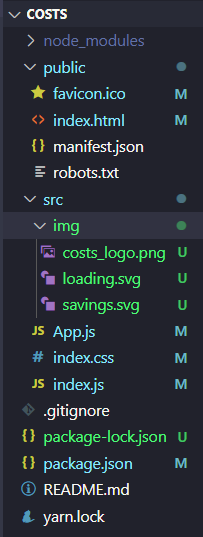
code .

npm start

## Instalando dependências do projeto

npm install json-server react-icons react-router-dom uuid

## Configurando o projeto



https://github.com/matheusbattisti/curso\_react\_yt

**src\App.js**

function App() {

return (

<div className="App">

<p>Costs</p>

</div>

);

}

export default App;

**src\index.js**

import React from 'react';

import ReactDOM from 'react-dom';

import App from './App';

import './index.css';

ReactDOM.render(

<React.StrictMode>

<App />

</React.StrictMode>,

document.getElementById('root')

);

**src\index.css**

\*{

padding: 0;

margin: 0;

box-sizing: border-box;

font-family: 'Open Sans', sans-serif;

}

html, body, #root{

background-color: #efefef;

height: 100%;

}

**public\index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<meta name="theme-color" content="#000000" />

<meta

name="description"

content="Web site created using create-react-app"

/>

<link rel="apple-touch-icon" href="%PUBLIC\_URL%/logo192.png" />

<!--

manifest.json provides metadata used when your web app is installed on a

user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/

-->

<link rel="manifest" href="%PUBLIC\_URL%/manifest.json" />

<!--

Notice the use of %PUBLIC\_URL% in the tags above.

It will be replaced with the URL of the `public` folder during the build.

Only files inside the `public` folder can be referenced from the HTML.

Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC\_URL%/favicon.ico" will

work correctly both with client-side routing and a non-root public URL.

Learn how to configure a non-root public URL by running `npm run build`.

-->

<title>Costs</title>

<!-- Google Fonts - Open Sans -->

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Open+Sans:ital,wght@0,400;0,500;0,600;0,700;0,800;1,300;1,400;1,500;1,600;1,700;1,800&display=swap" rel="stylesheet">

</head>

<body>

<noscript>You need to enable JavaScript to run this app.</noscript>

<div id="root"></div>

<!--

This HTML file is a template.

If you open it directly in the browser, you will see an empty page.

You can add webfonts, meta tags, or analytics to this file.

The build step will place the bundled scripts into the <body> tag.

To begin the development, run `npm start` or `yarn start`.

To create a production bundle, use `npm run build` or `yarn build`.

-->

</body>

</html>



# Aula 18 - Estruturando o projeto

Nesta aula estruturaremos o nosso projeto de #React

O início deve ter bases sólidas, para que não seja preciso voltar e modificar detalhes que poderiam ser configurados quando o projeto é criado, este vídeo se dedica justamente a isso: criar um bom alicerce no nosso projeto!

**src\App.js**

import { BrowserRouter as Router, Link, Route, Routes } from 'react-router-dom'

import Container from './components/layout/Container'

import Company from './components/pages/Company'

import Contact from './components/pages/Contact'

import Home from './components/pages/Home'

import NewProject from './components/pages/NewProject'

function App() {

return (

<Router>

<div>

<Link exact to='/'>Home</Link>

<Link to='/company'>Empresa</Link>

<Link to='/contact'>Contato</Link>

<Link to='/newproject'>Novo Projeto</Link>

</div>

<Container customClass="min-height">

<Routes>

<Route exact path="/" element={<Home />} />

<Route path="/company" element={<Company />} />

<Route path="/contact" element={<Contact />} />

<Route path="/newproject" element={<NewProject />} />

</Routes>

</Container>

<p>Footer</p>

</Router>

);

}

export default App;

**src\components\layout\Container.js**

import styles from './Container.module.css'

function Container(props){

return (

<div className={`${styles.container} ${styles[props.customClass]}`}>

{props.children}

</div>

)

}

export default Container

**src\components\layout\Container.module.css**

.container{

width: 1200px;

display: flex;

justify-content: space-between;

margin: 0 auto;

flex-wrap: wrap;

}

.min-height{

min-height: 75%;

}

.start{

justify-content: flex-start;

}

.column{

flex-direction: column;

justify-content: flex-start;

}

**src\components\pages\Company.js**

function Company() {

return (

<div>

<h1>Empresa</h1>

</div>

)

}

export default Company

**src\components\pages\Contact.js**

function Contact() {

return (

<div>

<h1>Contato</h1>

</div>

)

}

export default Contact

**src\components\pages\Home.js**

function Home(){

return (

<div>

<h1>Home</h1>

</div>

)

}

export default Home

**src\components\pages\NewProject.js**

function NewProject() {

return (

<div>

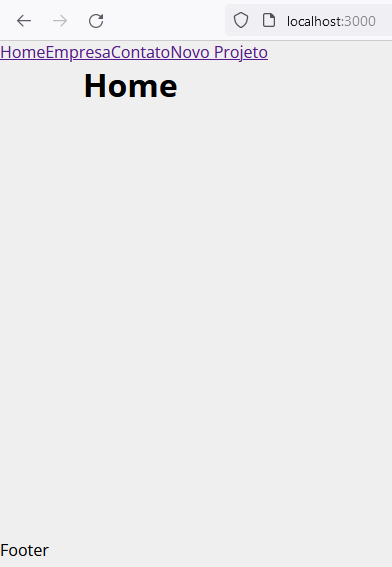
<h1>Novo Projeto</h1>

</div>

)

}

export default NewProject



# Aula 19 - Navbar e footer

**src\components\pages\Projects.js**

function Projects(){

return(

<div>

<h1>Projetos</h1>

</div>

)

}

export default Projects

**src\components\layout\Navbar.js**

import { Link } from 'react-router-dom'

import logo from '../../img/costs\_logo.png'

import Container from './Container'

import styles from './Navbar.module.css'

function Navbar() {

return (

<nav className={styles.navbar}>

<Container>

<Link to='/'>

<img src={logo} alt="Costs" />

</Link>

<ul className={styles.list}>

<li className={styles.item}><Link exact to='/'>Home</Link></li>

<li className={styles.item}><Link to='/projects'>Projetos</Link></li>

<li className={styles.item}><Link to='/company'>Empresa</Link></li>

<li className={styles.item}><Link to='/contact'>Contato</Link></li>

</ul>

</Container>

</nav>

)

}

export default Navbar

**src\components\layout\Navbar.module.css**

.navbar{

display: flex;

justify-content: space-between;

background-color: #222;

padding: 1em;

}

.list{

display: flex;

list-style: none;

align-items: center;

}

.item{

margin-right: 1em;

}

.item a{

color: #fff;

text-decoration: none;

}

.item a:hover{

color: #ffbb33;

text-decoration: none;

}

**src\components\layout\Footer.js**

import { FaFacebook, FaInstagram, FaLinkedin } from 'react-icons/fa'

import styles from './Footer.module.css'

function Footer(){

return (

<footer className={styles.footer}>

<ul className={styles.social\_list}>

<li><FaFacebook /></li>

<li><FaInstagram /></li>

<li><FaLinkedin /></li>

</ul>

<p className={styles.copy\_right}><span>Costs</span> &copy; 2021</p>

</footer>

)

}

export default Footer

**src\components\layout\Footer.module.css**

.footer{

background-color: #222;

color: #fff;

padding: 2em;

text-align: center;

}

.social\_list{

display: flex;

justify-content: center;

list-style-type: none;

}

.social\_list li{

margin: 0 1em;

}

.social\_list li:hover{

color: #ffbb33;

}

.social\_list svg{

font-size: 1.5em;

cursor: pointer;

}

.copy\_right{

margin-top: 2em;

}

.copy\_right span{

font-weight: bold;

color: #ffbb33;

}

**src\App.js**

import { BrowserRouter as Router, Route, Routes } from 'react-router-dom'

import Container from './components/layout/Container'

import Footer from './components/layout/Footer'

import Navbar from './components/layout/Navbar'

import Company from './components/pages/Company'

import Contact from './components/pages/Contact'

import Home from './components/pages/Home'

import NewProject from './components/pages/NewProject'

import Projects from './components/pages/Projects'

function App() {

return (

<Router>

<Navbar />

<Container customClass="min-height">

<Routes>

<Route path="/" element={<Home />} />

<Route path="/projects" element={<Projects />} />

<Route path="/company" element={<Company />} />

<Route path="/contact" element={<Contact />} />

<Route path="/newproject" element={<NewProject />} />

</Routes>

</Container>

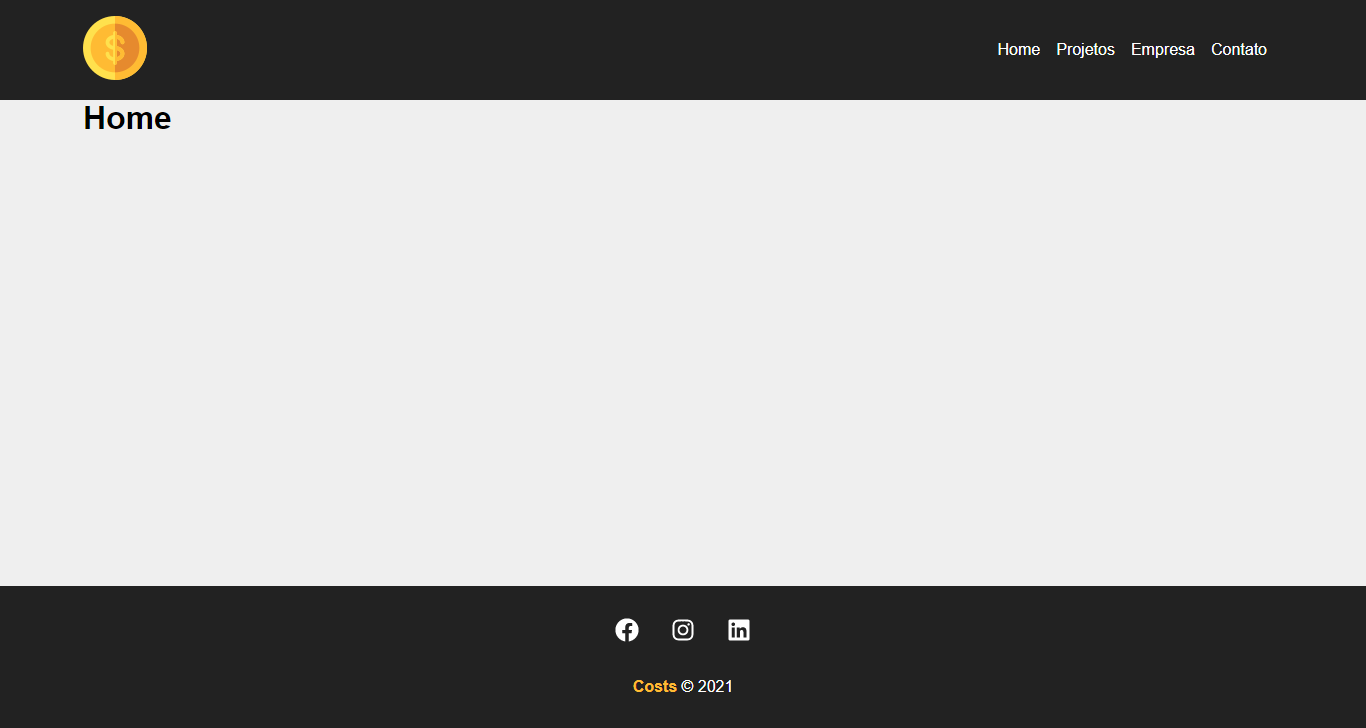
<Footer />

</Router>

);

}

export default App;



# Aula 20 - Criando a home

**src\components\pages\Home.js**

import savings from '../../img/savings.svg'

import LinkButton from '../layout/LinkButton'

import styles from './Home.module.css'

function Home(){

return (

<section className={styles.home\_container}>

<h1>Bem-vindo ao <span>Costs</span></h1>

<p>Comece a gerenciar os seus projetos agora mesmo!</p>

<LinkButton to="newproject" text="Criar Projeto" />

<img src={savings} alt="Costs" />

</section>

)

}

export default Home

**src\components\pages\Home.module.css**

.home\_container{

width: 100%;

display: flex;

flex-direction: column;

align-items: center;

justify-content: center;

padding: 4em;

}

.home\_container h1{

font-size: 2.5em;

margin-bottom: .5em;

}

.home\_container h1 span{

color: #ffbb33;

padding: 0 .2em;

background-color: #222;

}

.home\_container p{

margin-bottom: 1.5em;

color: #7b7b7b

}

.home\_container img{

width: 350px;

margin: 2em 0;

}

**src\components\layout\LinkButton.js**

import { Link } from 'react-router-dom'

import styles from './LinkButton.module.css'

function LinkButton({to,text}){

return(

<Link className={styles.btn} to={to}>

{text}

</Link>

)

}

export default LinkButton

**src\components\layout\LinkButton.module.css**

.btn{

background-color: #222;

color: #fff;

padding: .5em 1em;

text-decoration: none;

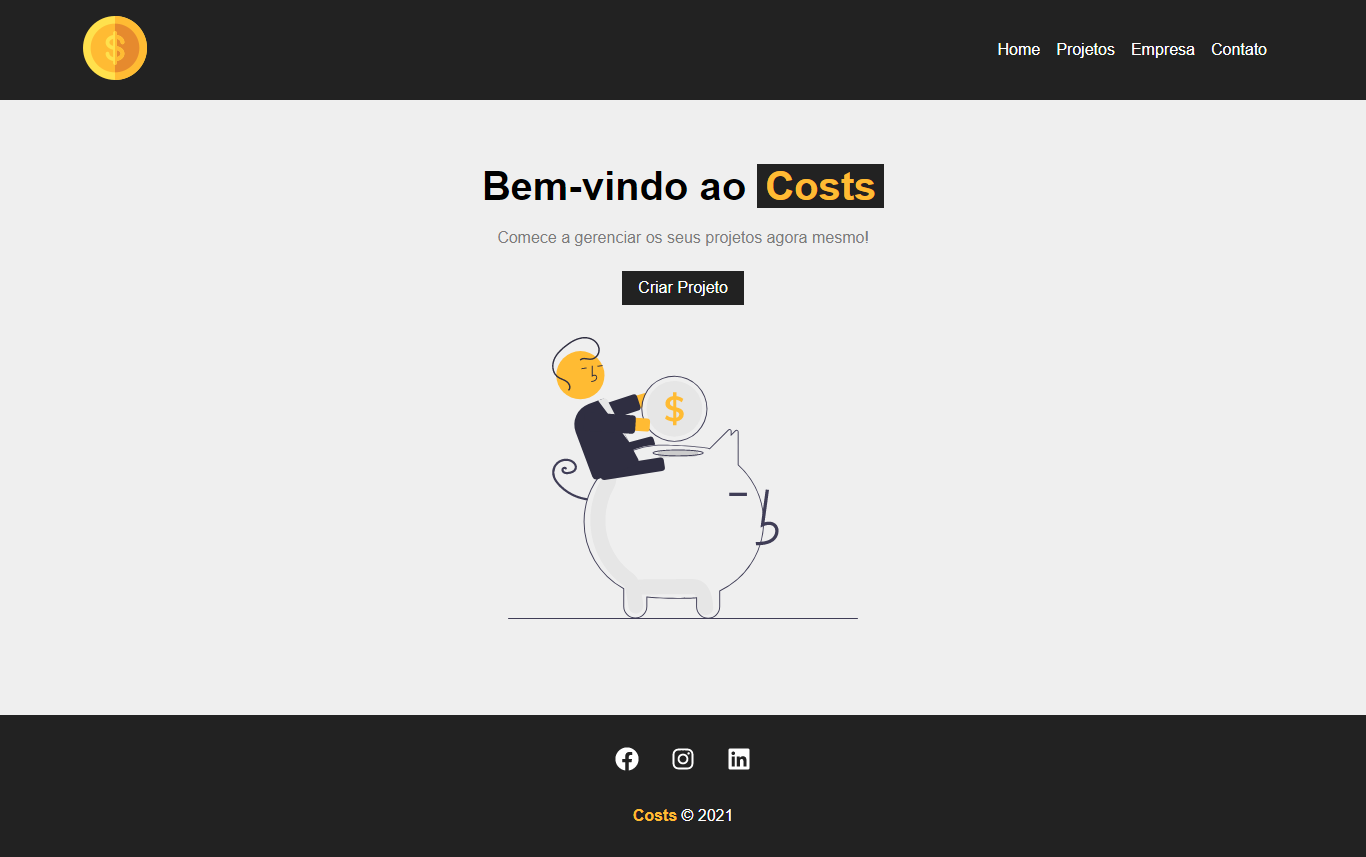
transition: 0.5s;

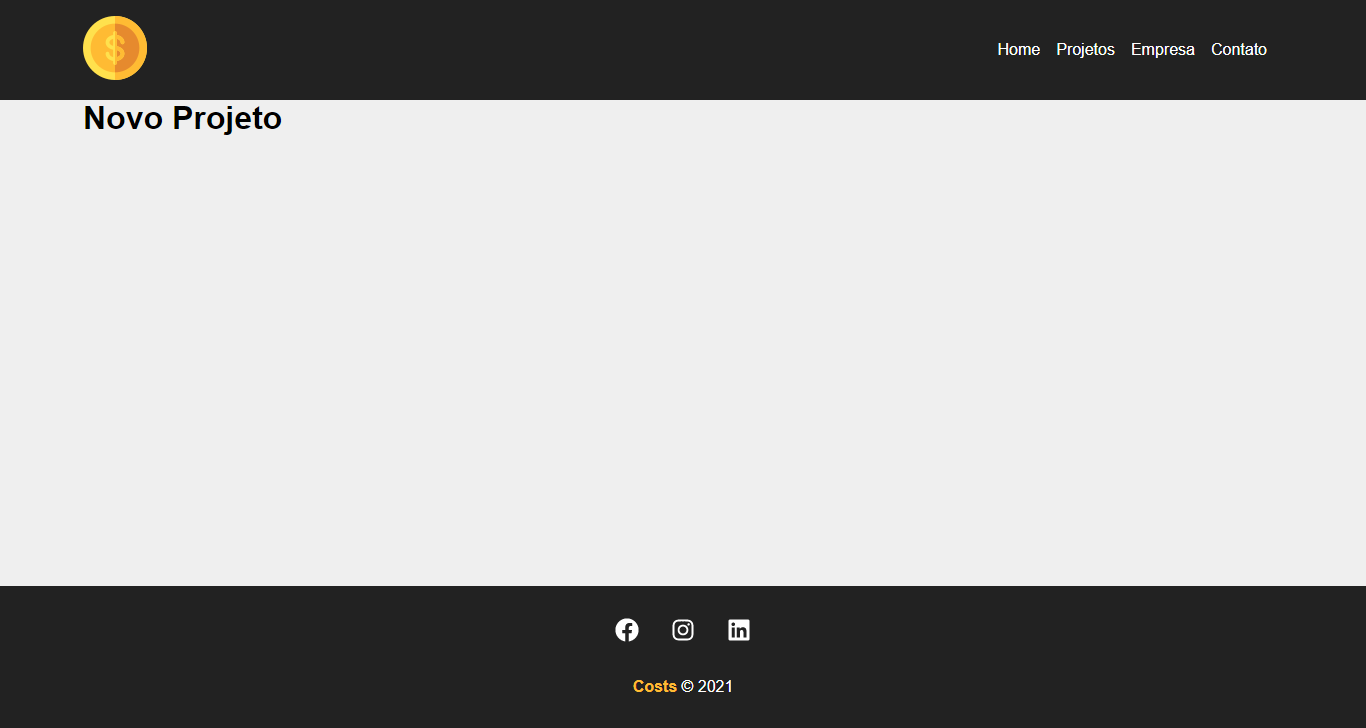
}

.btn:hover{

color: #ffbb33;

}





# Aula 21 - Página para o formulário de projeto

**src\components\pages\NewProject.js**

import ProjectForm from '../project/ProjectForm'

import styles from './NewProject.module.css'

function NewProject() {

return (

<div className={styles.newproject\_container}>

<h1>Criar Projeto</h1>

<p>Crie seu projeto para depois adicionar os serviços</p>

<p><ProjectForm /></p>

</div>

)

}

export default NewProject

**src\components\project\ProjectForm.js**

function ProjectForm(){

return (

<form>

<div>

<input type="text" placeholder="Insira o nome do projeto" />

</div>

<div>

<input type="number" placeholder="Insira o orçamento total" />

</div>

<div>

<select name="category\_id" id="">

<option disabled selected>Selecione a categoria</option>

</select>

</div>

<div>

<input type="submit" value="Criar Projeto" />

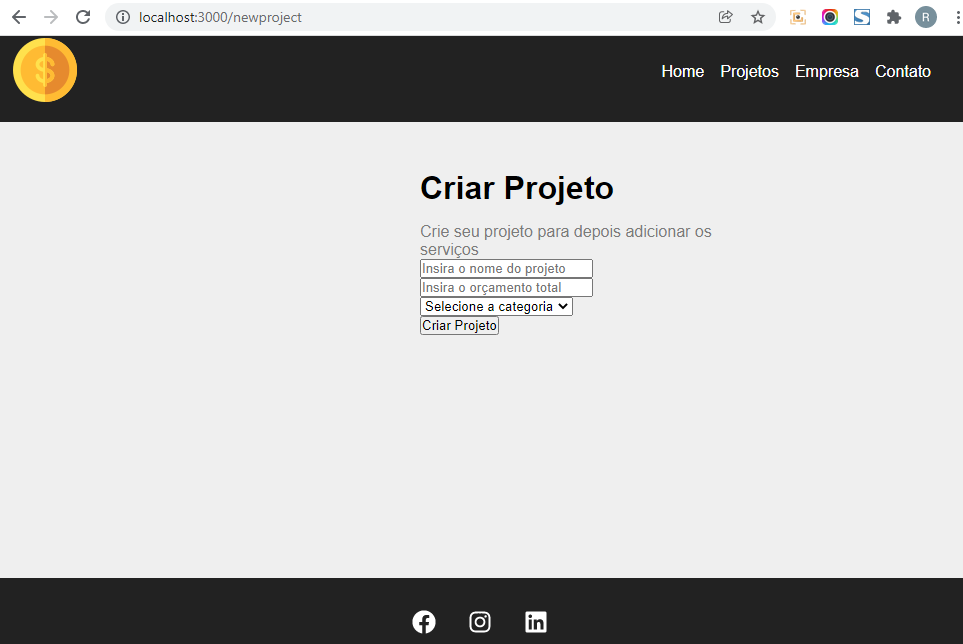
</div>

</form>

)

}

export default ProjectForm



# Aula 22 - Componentização de formulário

**src\components\form\Input.js**

import styles from './Input.module.css'

function Input({type, text, name, placeholder, handleOnChange, value}){

return(

<div className={styles.form\_control}>

<label htmlFor={name}>{text}:</label>

<input

type={type}

name={name}

id={name}

placeholder={placeholder}

onChange={handleOnChange}

value={value}

/>

</div>

)

}

export default Input

**src\components\form\Input.module.css**

.form\_control{

display: flex;

flex-direction: column;

margin-bottom: 1em;

}

.form\_control label{

margin-bottom: 0.6em;

font-weight: bold;

}

.form\_control input{

padding: 0.7em;

border-radius: 0;

border: none;

}

.form-control input::placeholder{

color: #7b7b7b;

}

**src\components\form\Select.js**

import styles from './Select.module.css'

function Select({ text, name, options, handleOnChange, value }) {

return (

<div className={styles.form\_control}>

<label htmlFor={name}>{text}:</label>

<select name={name} id={name}>

<option>Selecione uma opção</option>

</select>

</div>

)

}

export default Select

**src\components\form\Select.module.css**

.form\_control{

display: flex;

flex-direction: column;

margin-bottom: 1em;

}

.form\_control label{

margin-bottom: 0.6em;

font-weight: bold;

}

.form\_control select{

padding: 0.7em;

border-radius: 0;

border: none;

}

**src\components\form\SubmitButton.js**

import styles from './SubmitButton.module.css'

function SubmitButton({ text }) {

return (

<div>

<button className={styles.btn}>{text}</button>

</div>

)

}

export default SubmitButton

**src\components\form\SubmitButton.module.css**

.btn{

background-color: #222;

color: #fff;

padding: 0.7em 1.2em;

text-decoration: none;

transition: 0.5s;

cursor: pointer;

border: none;

}

.btn:hover{

color: #ffbb33;

}

**src\components\project\ProjectForm.js**

import Input from '../form/Input'

import Select from '../form/Select'

import SubmitButton from '../form/SubmitButton'

import styles from './ProjectForm.module.css'

function ProjectForm({btnText}){

return (

<form className={styles.form}>

<Input

type="text"

text="Nome do projeto"

name="name"

placeholder="Insira o nome do projeto"

/>

<Input

type="number"

text="Orçamento do projeto"

name="budget"

placeholder="Insira o orçamento total"

/>

<Select name="category\_id" text="Selecione a categoria" />

<SubmitButton text={btnText} />

</form>

)

}

export default ProjectForm

**src\components\project\ProjectForm.module.css**

.form{

width: 100%;

margin: 2em 0;

}

**src\components\pages\NewProject.js**

import ProjectForm from '../project/ProjectForm'

import styles from './NewProject.module.css'

function NewProject() {

return (

<div className={styles.newproject\_container}>

<h1>Criar Projeto</h1>

<p>Crie seu projeto para depois adicionar os serviços</p>

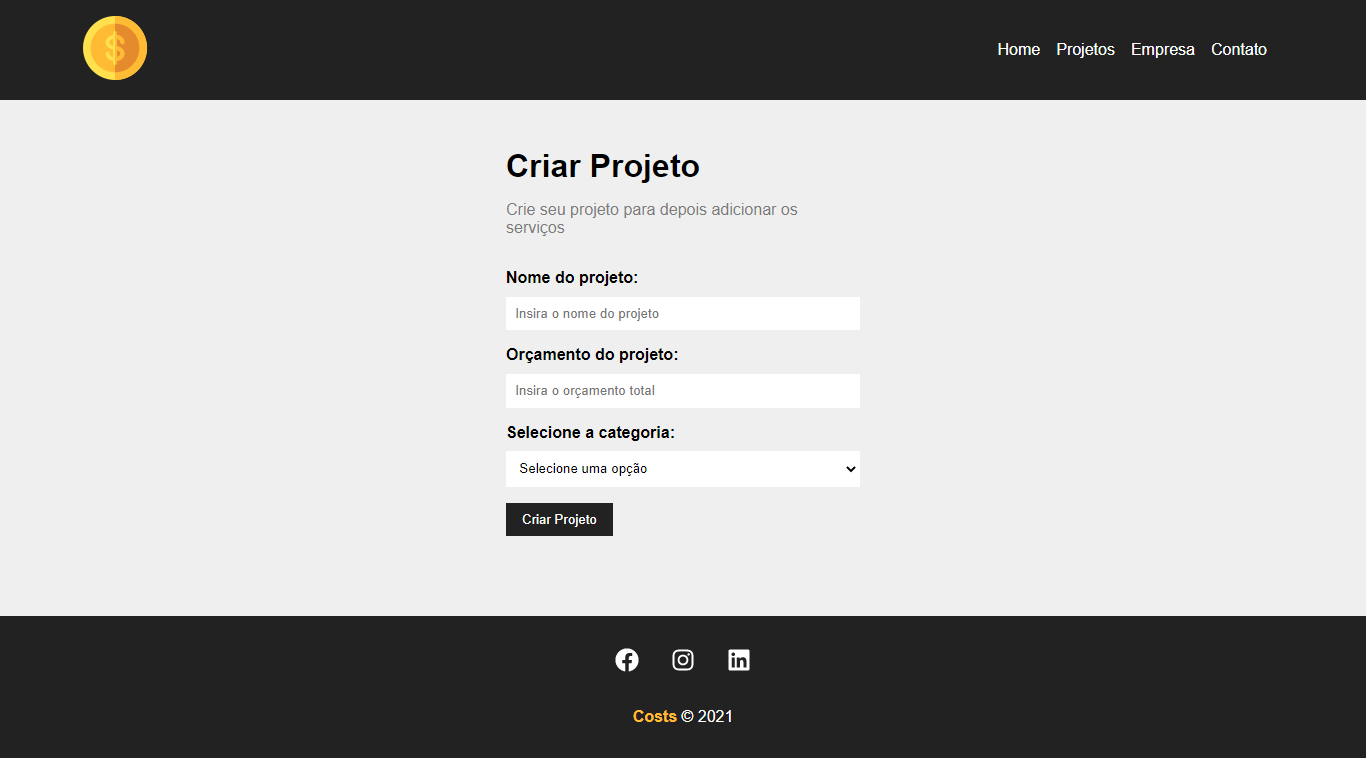
<ProjectForm btnText="Criar Projeto" />

</div>

)

}

export default NewProject



# Aula 23 - Conectando com API pelo React

**db.json**

{

"projects": []

}

**package.json**

{

"name": "costs",

"version": "0.1.0",

"private": true,

"dependencies": {

"@testing-library/jest-dom": "^5.11.4",

"@testing-library/react": "^11.1.0",

"@testing-library/user-event": "^12.1.10",

"json-server": "^0.17.0",

"react": "^17.0.2",

"react-dom": "^17.0.2",

"react-icons": "^4.3.1",

"react-router-dom": "^6.0.2",

"react-scripts": "4.0.3",

"uuid": "^8.3.2",

"web-vitals": "^1.0.1"

},

"scripts": {

"start": "react-scripts start",

"backend": "json-server --watch db.json --port 5000",

"build": "react-scripts build",

"test": "react-scripts test",

"eject": "react-scripts eject"

},

"eslintConfig": {

"extends": [

"react-app",

"react-app/jest"

]

},

"browserslist": {

"production": [

">0.2%",

"not dead",

"not op\_mini all"

],

"development": [

"last 1 chrome version",

"last 1 firefox version",

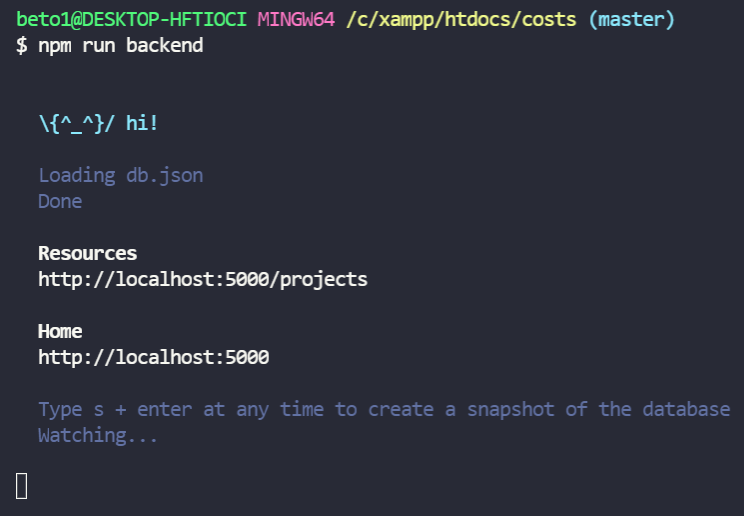
"last 1 safari version"

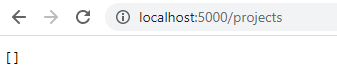
]

}

}

npm run backend





**db.json**

{

"projects": [],

"categories": [

{

"id": 1,

"name": "Infra"

},

{

"id": 2,

"name": "Desenvolvimento"

},

{

"id": 3,

"name": "Design"

},

{

"id": 4,

"name": "Planejamento"

}

]

}



**src\components\project\ProjectForm.js**

import { useState, useEffect } from 'react'

import Input from '../form/Input'

import Select from '../form/Select'

import SubmitButton from '../form/SubmitButton'

import styles from './ProjectForm.module.css'

function ProjectForm({btnText}){

const[categories, setCategories] = useState([])

useEffect(() => {

fetch("http://localhost:5000/categories",{

method: "GET",

headers: {

'Content-Type': 'application/json'

}

})

.then((resp) => resp.json())

.then((data) => {

setCategories(data)

})

.catch((err) => console.log(err))

}, [])

return (

<form className={styles.form}>

<Input

type="text"

text="Nome do projeto"

name="name"

placeholder="Insira o nome do projeto"

/>

<Input

type="number"

text="Orçamento do projeto"

name="budget"

placeholder="Insira o orçamento total"

/>

<Select

name="category\_id"

text="Selecione a categoria"

options={categories}

/>

<SubmitButton text={btnText} />

</form>

)

}

export default ProjectForm

**src\components\form\Select.js**

import styles from './Select.module.css'

function Select({ text, name, options, handleOnChange, value }) {

return (

<div className={styles.form\_control}>

<label htmlFor={name}>{text}:</label>

<select name={name} id={name}>

<option>Selecione uma opção</option>

{options.map((option) => (

<option value={option.id} key={option.id}>{option.name}</option>

))}

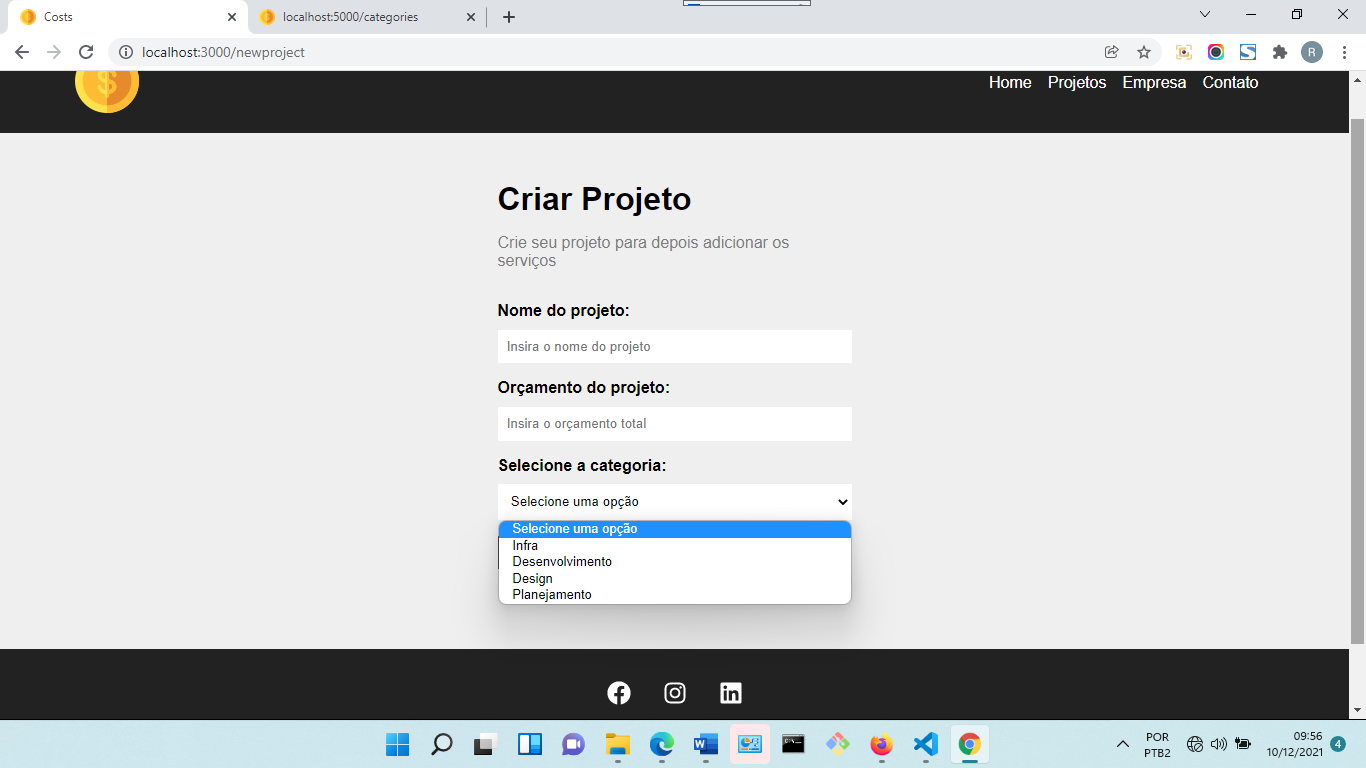
</select>

</div>

)

}

export default Select



# Aula 24 - Inserindo dados no banco via API

**src\components\pages\NewProject.js**

import { useNavigate } from 'react-router-dom'

import ProjectForm from '../project/ProjectForm'

import styles from './NewProject.module.css'

function NewProject() {

const navigate = useNavigate()

function createPost(project){

// Initialize costs and services

project.cost = 0

project.services = []

fetch("http://localhost:5000/projects", {

method: "POST",

headers: {

"Content-Type": "application/json"

},

body: JSON.stringify(project),

}).then((resp) => resp.json())

.then((data) => {

console.log(data)

// redirect

navigate('/projects', {message: 'Projeto criado com sucesso!'})

})

.catch((err) => console.log(err))

}

return (

<div className={styles.newproject\_container}>

<h1>Criar Projeto</h1>

<p>Crie seu projeto para depois adicionar os serviços</p>

<ProjectForm handleSubmit={createPost} btnText="Criar Projeto" />

</div>

)

}

export default NewProject

**src\components\project\ProjectForm.js**

import { useEffect, useState } from 'react'

import Input from '../form/Input'

import Select from '../form/Select'

import SubmitButton from '../form/SubmitButton'

import styles from './ProjectForm.module.css'

function ProjectForm({ handleSubmit, btnText, projectData}){

const[categories, setCategories] = useState([])

const [project, setProject] = useState(projectData || {})

useEffect(() => {

fetch("http://localhost:5000/categories",{

method: "GET",

headers: {

'Content-Type': 'application/json'

}

})

.then((resp) => resp.json())

.then((data) => {

setCategories(data)

})

.catch((err) => console.log(err))

}, [])

const submit = (e) => {

e.preventDefault()

// console.log(project)

handleSubmit(project)

}

function handleChange(e){

setProject({...project, [e.target.name]: e.target.value })

// console.log(project)

}

function handleCategory(e) {

setProject({

...project,

category: {

id: e.target.value,

name: e.target.options[e.target.selectedIndex].text,

},

})

}

return (

<form onSubmit={submit} className={styles.form}>

<Input

type="text"

text="Nome do projeto"

name="name"

placeholder="Insira o nome do projeto"

handleOnChange={handleChange}

value={project.name ? project.name : ''}

/>

<Input

type="number"

text="Orçamento do projeto"

name="budget"

placeholder="Insira o orçamento total"

handleOnChange={handleChange}

value={project.budget ? project.budget : ''}

/>

<Select

name="category\_id"

text="Selecione a categoria"

options={categories}

handleOnChange={handleCategory}

value={project.category ? project.category.id : ''}

/>

<SubmitButton text={btnText} />

</form>

)

}

export default ProjectForm

**src\components\form\Select.js**

import styles from './Select.module.css'

function Select({ text, name, options, handleOnChange, value }) {

return (

<div className={styles.form\_control}>

<label htmlFor={name}>{text}:</label>

<select

name={name}

id={name}

onChange={handleOnChange}

value={value || ''}

>

<option>Selecione uma opção</option>

{options.map((option) => (

<option value={option.id} key={option.id}>{option.name}</option>

))}

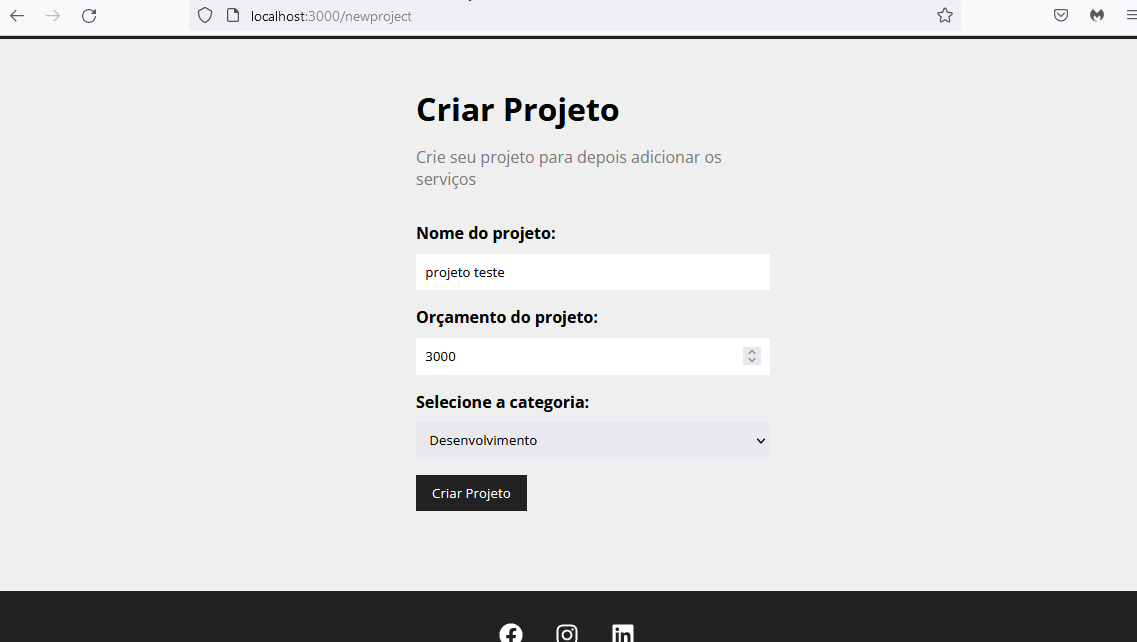
</select>

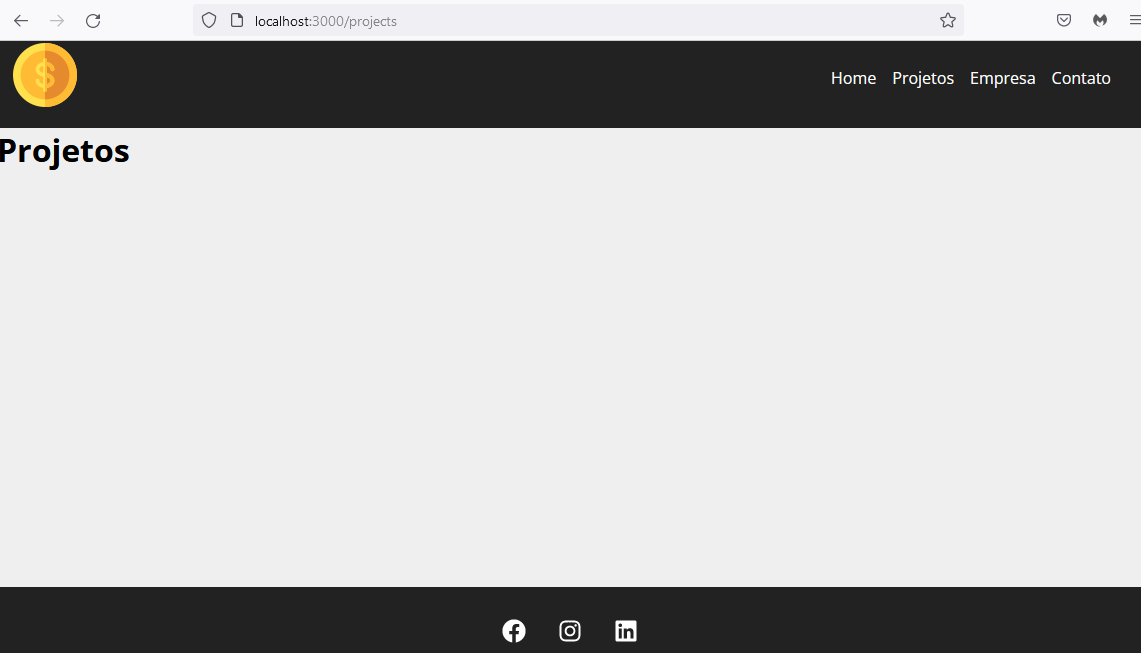
</div>

)

}

export default Select





**db.json**

{

"projects": [

{

"name": "projeto teste",

"budget": "3000",

"category": {

"id": "2",

"name": "Desenvolvimento"

},

"cost": 0,

"services": [],

"id": 1

}

],

"categories": [

{

"id": 1,

"name": "Infra"

},

{

"id": 2,

"name": "Desenvolvimento"

},

{

"id": 3,

"name": "Design"

},

{

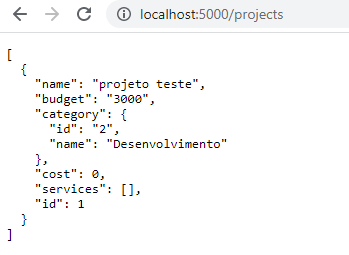
"id": 4,

"name": "Planejamento"

}

]

}



# Aula 25 - Mensagens do sistema

**src\components\pages\Projects.js**

import { useLocation } from 'react-router-dom'

import Message from '../layout/Message'

function Projects(){

const location = useLocation()

let message = ''

if(location.state){

message = location.state.message

}

return(

<div>

<h1>Meus Projetos</h1>

{message && <Message type="success" msg={message} />}

</div>

)

}

export default Projects

**src\components\layout\Message.js**

import { useEffect, useState } from 'react';

import styles from './Message.module.css';

function Message({type, msg}){

const[visible, setVisible] = useState(false)

useEffect(() => {

if(!msg){

setVisible(false)

return

}

setVisible(true)

const timer = setTimeout(() => {

setVisible(false)

}, 3000)

return () => clearTimeout(timer)

}, [msg]);

return(

<>

{visible && (

<div className={`${styles.message} ${styles[type]}`}>{msg}</div>

)}

</>

)

}

export default Message

**src\components\layout\Message.module.css**

.message{

width: 100%;

padding: 1em;

border: 1px solid #000;

margin: 0 auto;

text-align: center;

margin-bottom: 2em;

border-radius: 5px;

}

.success{

color: #155724;

background-color: #d4edda;

border-color: #c3e6cb;

}

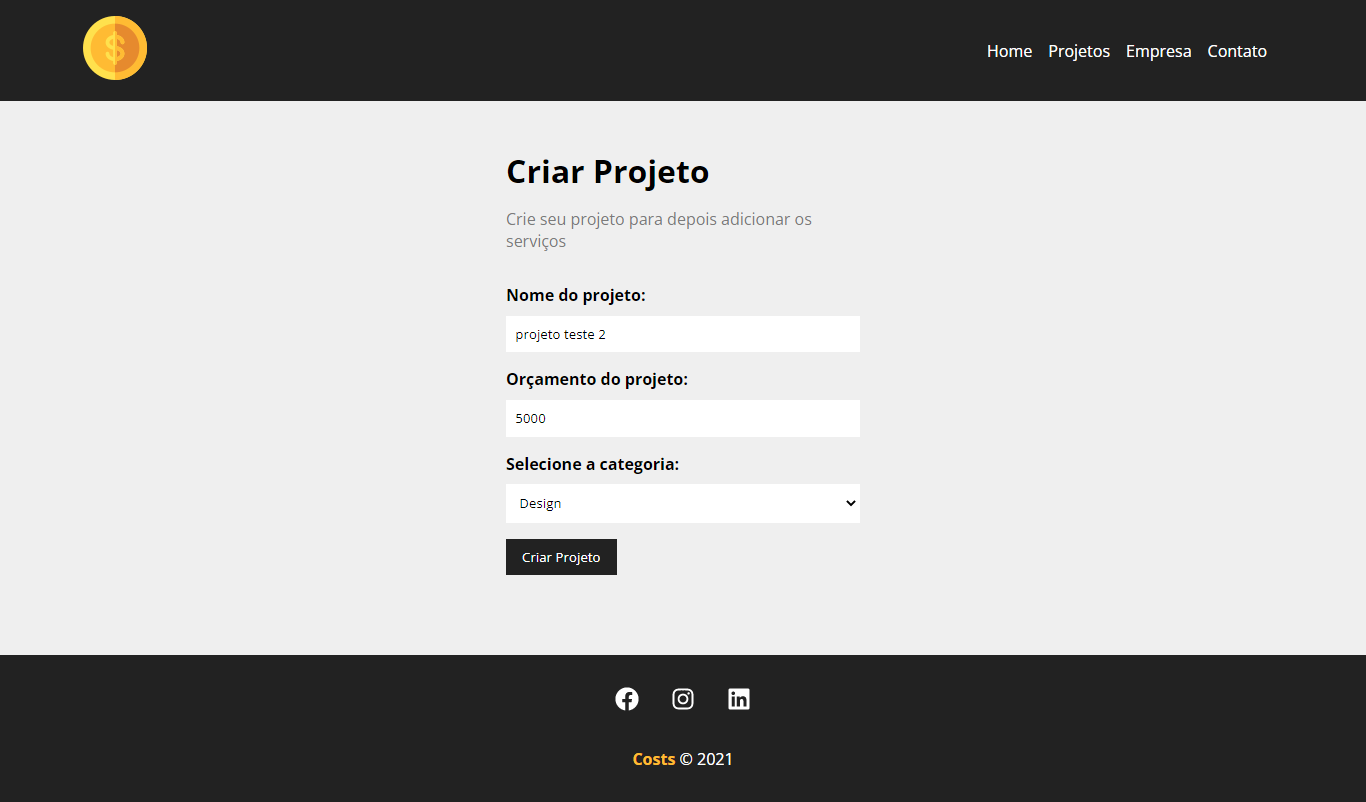
.error{

color: #721c24;

background-color: #f8d7da;

border-color: #f5c6cb;

}



# Aula 26 - Criando Dashboard

**src\components\pages\Projects.js**

import { useLocation } from 'react-router-dom'

import Container from '../layout/Container'

import LinkButton from '../layout/LinkButton'

import Message from '../layout/Message'

import styles from './Projects.module.css'

function Projects(){

const location = useLocation()

let message = ''

if(location.state){

message = location.state.message

}

return(

<div className={styles.project\_container}>

<div className={styles.title\_container}>

<h1>Meus Projetos</h1>

<LinkButton to="/newproject" text="Criar Projeto" />

</div>

{message && <Message type="success" msg={message} />}

<Container customClass="start">

<p>Projetos...</p>

</Container>

</div>

)

}

export default Projects

**src\components\pages\Projects.module.css**

.project\_container{

padding: 2em;

}

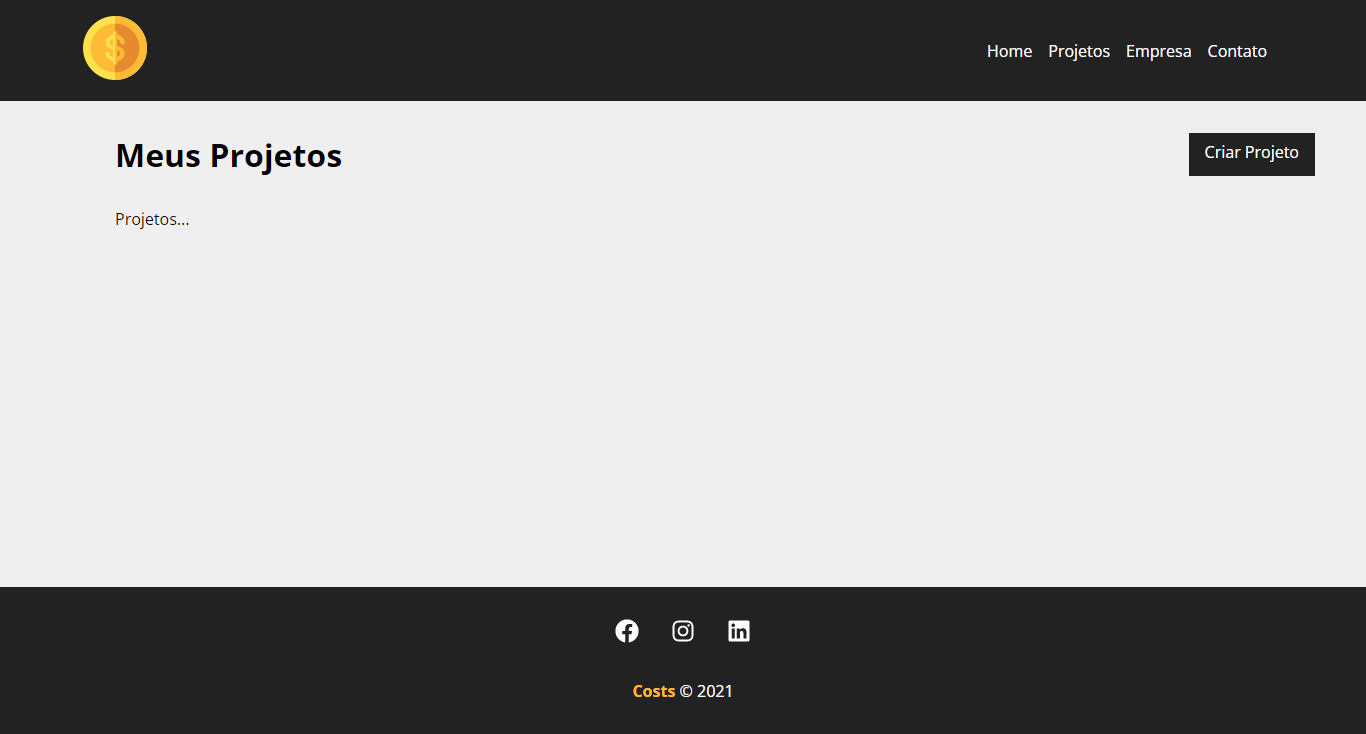
.title\_container{

display: flex;

justify-content: space-between;

margin-bottom: 2em;

}



# Aula 27 - Resgatando projetos do banco de dados

**src\components\project\ProjectCard.js**

import { BsFillTrashFill, BsPencil } from 'react-icons/bs'

import { Link } from 'react-router-dom'

import styles from './ProjectCard.module.css'

function ProjectCard({id, name, budget, category, handleRemove}){

return(

<div className={styles.project\_card}>

<h4>{name}</h4>

<p><span>Orçamento:</span> R${budget}</p>

<p className={styles.category\_text}>

<span className={`${styles[category.toLowerCase()]}`}></span> {category}

</p>

<div className={styles.project\_card\_actions}>

<Link to="/">

<BsPencil /> Editar

</Link>

<button>

<BsFillTrashFill /> Excluir

</button>

</div>

</div>

)

}

export default ProjectCard

**src\components\project\ProjectCard.module.css**

.project\_card{

padding: 1em;

border: 1px solid #7a7a7a;

border-radius: 5px;

width: 24%;

margin: 0.5%;

}

.project\_card h4{

background-color: #222;

color: #ffbb33;

padding: 0.4em;

margin-bottom: 1.3em;

font-size: 1.3em;

}

.project\_card p{

color: #7a7a7a;

margin-bottom: 1em;

}

.project\_card p span{

font-weight: bold;

}

.category\_text{

display: flex;

align-items: center;

}

.category\_text span{

display: block;

width: 12px;

height: 12px;

border-radius: 50%;

background-color: #ccc;

margin-right: 5px;

}

.category\_text .infra{

background-color: #ffaebc;

}

.category\_text .desenvolvimento{

background-color: #a0e7e5;

}

.category\_text .design{

background-color: #b4f8c8;

}

.category\_text .planejamento{

background-color: #fbe7c6;

}

.project\_card actions{

margin-top: 1.2em;

display: flex;

align-items: center;

}

.project\_card\_actions a,

.project\_card\_actions button{

text-decoration: none;

border: none;

background-color: #fff;

color: #222;

font-size: 0.9em;

padding: 0.6em 1em;

margin-right: 1em;

cursor: pointer;

border: 1px solid #222;

display: flex;

align-items: center;

justify-content: center;

transition: 0.5s;

}

.project\_card\_actions a,

.project\_card\_actions button{

display: inline-flex

}

.project\_card\_actions svg{

margin-right: 0.5em;

}

.project\_card\_actions a:hover,

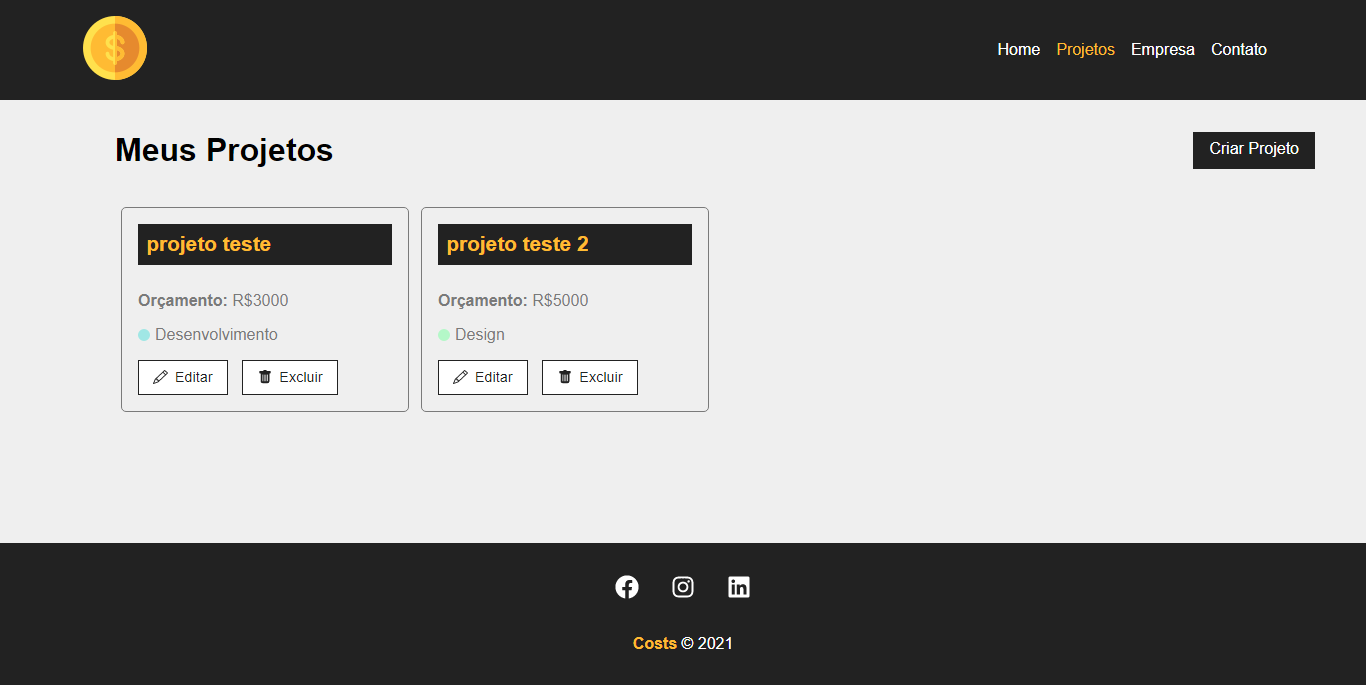
.project\_card\_actions button:hover

{

background-color: #222;

color: #ffbb33;

}



# Aula 28 - Criando componente de loader

npm start

npm run backend

**src\components\layout\Loading.js**

import loading from '../../img/loading.svg'

import styles from './Loading.module.css'

function Loading(){

return(

<div className={styles.loader\_container}>

<img className={styles.loader} src={loading} alt="Loading" />

</div>

)

}

export default Loading

**src\components\layout\Loading.module.css**

.loader\_container{

width: 100%;

height: 100%;

display: flex;

justify-content: center;

align-items: center;

}

.loader{

width: 50px;

}

**src\components\pages\Projects.js**

import { useEffect, useState } from 'react'

import { useLocation } from 'react-router-dom'

import Container from '../layout/Container'

import LinkButton from '../layout/LinkButton'

import Loading from '../layout/Loading'

import Message from '../layout/Message'

import ProjectCard from '../project/ProjectCard'

import styles from './Projects.module.css'

function Projects(){

const [projects, setProjects] = useState([])

const [removeLoading, setRemoveLoading] = useState(false)

const location = useLocation()

let message = ''

if(location.state){

message = location.state.message

}

useEffect(() => {

setTimeout(() => {

fetch("http://localhost:5000/projects", {

method: 'GET',

headers: {

'Content-Type': 'application/json'

}

}).then(resp => resp.json())

.then(data => {

console.log(data)

setProjects(data)

setRemoveLoading(true)

})

.catch((err) => console.log(err))

}, 300);

}, []);

return(

<div className={styles.project\_container}>

<div className={styles.title\_container}>

<h1>Meus Projetos</h1>

<LinkButton to="/newproject" text="Criar Projeto" />

</div>

{message && <Message type="success" msg={message} />}

<Container customClass="start">

{projects.length > 0 &&

projects.map((project) => (

<ProjectCard

id={project.id}

name={project.name}

budget={project.budget}

category={project.category.name}

key={project.id}

/>

)) }

{!removeLoading && <Loading />}

{removeLoading && projects.length === 0 && (

<p>Não há projetos cadastrados!</p>

)}

</Container>

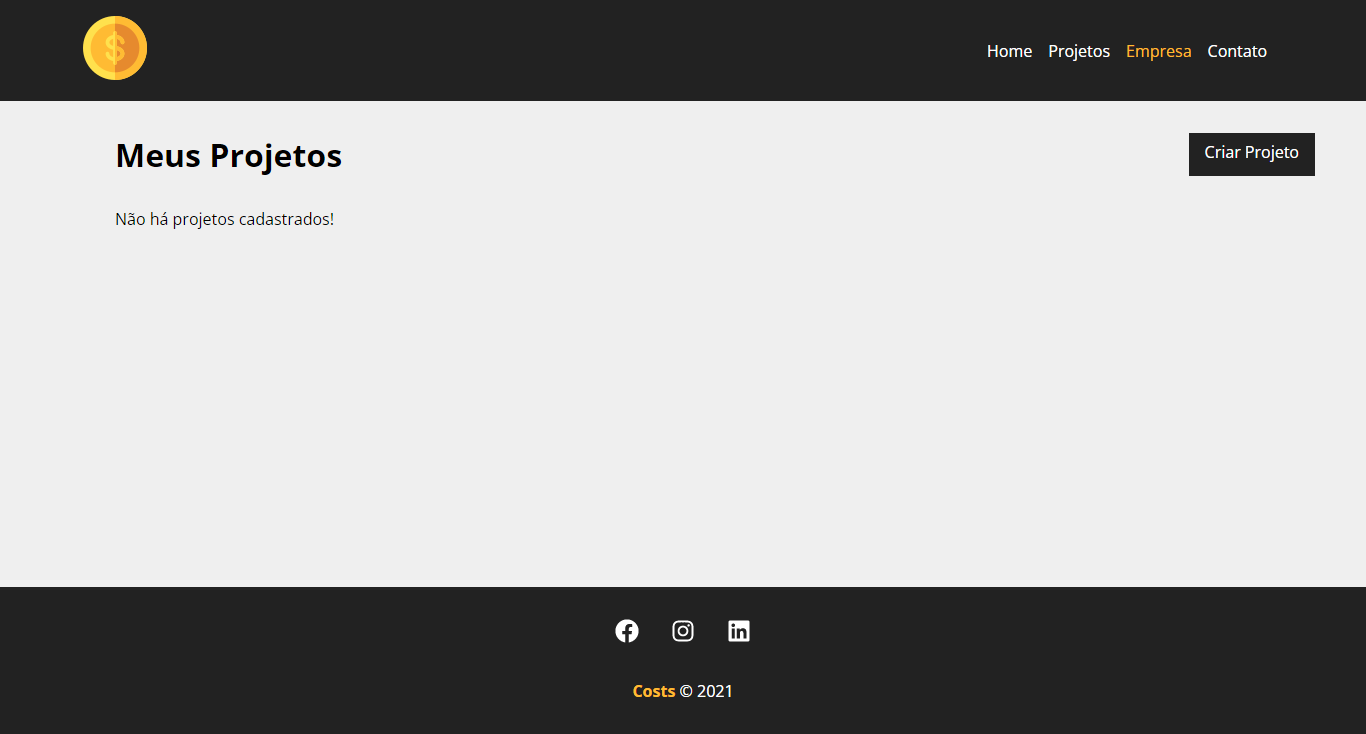
</div>

)

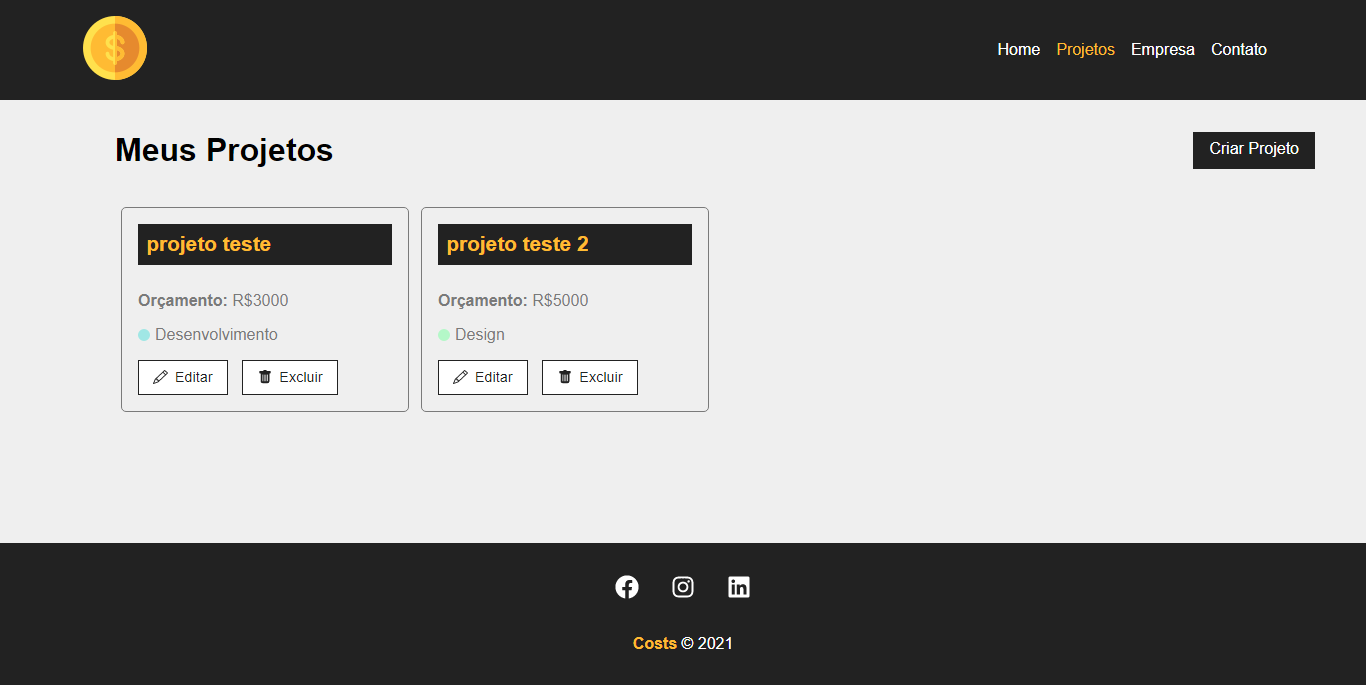
}

export default Projects

- Em db.json apague os dois projetos cadastrados.



- Cadastre-os novamente.



# Aula 29 - Removendo projetos

**src\components\project\ProjectCard.js**

import { BsFillTrashFill, BsPencil } from 'react-icons/bs'

import { Link } from 'react-router-dom'

import styles from './ProjectCard.module.css'

function ProjectCard({id, name, budget, category, handleRemove}){

const remove = (e) => {

e.preventDefault()

handleRemove(id)

}

return(

<div className={styles.project\_card}>

<h4>{name}</h4>

<p><span>Orçamento:</span> R${budget}</p>

<p className={styles.category\_text}>

<span className={`${styles[category.toLowerCase()]}`}></span> {category}

</p>

<div className={styles.project\_card\_actions}>

<Link to="/">

<BsPencil /> Editar

</Link>

<button onClick={remove}>

<BsFillTrashFill /> Excluir

</button>

</div>

</div>

)

}

export default ProjectCard

**src\components\pages\Projects.js**

import { useEffect, useState } from 'react'

import { useLocation } from 'react-router-dom'

import Container from '../layout/Container'

import LinkButton from '../layout/LinkButton'

import Loading from '../layout/Loading'

import Message from '../layout/Message'

import ProjectCard from '../project/ProjectCard'

import styles from './Projects.module.css'

function Projects(){

const [projects, setProjects] = useState([])

const [removeLoading, setRemoveLoading] = useState(false)

const [projectMessage, setProjectMessage] = useState('')

const location = useLocation()

let message = ''

if(location.state){

message = location.state.message

}

useEffect(() => {

setTimeout(() => {

fetch("http://localhost:5000/projects", {

method: 'GET',

headers: {

'Content-Type': 'application/json'

}

}).then(resp => resp.json())

.then(data => {

console.log(data)

setProjects(data)

setRemoveLoading(true)

})

.catch((err) => console.log(err))

}, 300);

}, []);

function removeProject(id){

fetch(`http://localhost:5000/projects/${id}`, {

method: "DELETE",

headers: {

'Content-Type': 'application/json'

},

}).then((resp) => resp.json())

.then(() => {

setProjects(projects.filter((project) => project.id !== id))

setProjectMessage('Projeto removido com sucesso!')

})

.catch(err => console.log(err))

}

return(

<div className={styles.project\_container}>

<div className={styles.title\_container}>

<h1>Meus Projetos</h1>

<LinkButton to="/newproject" text="Criar Projeto" />

</div>

{message && <Message type="success" msg={message} />}

{projectMessage && <Message type="success" msg={projectMessage} />}

<Container customClass="start">

{projects.length > 0 &&

projects.map((project) => (

<ProjectCard

id={project.id}

name={project.name}

budget={project.budget}

category={project.category.name}

key={project.id}

handleRemove={removeProject}

/>

)) }

{!removeLoading && <Loading />}

{removeLoading && projects.length === 0 && (

<p>Não há projetos cadastrados!</p>

)}

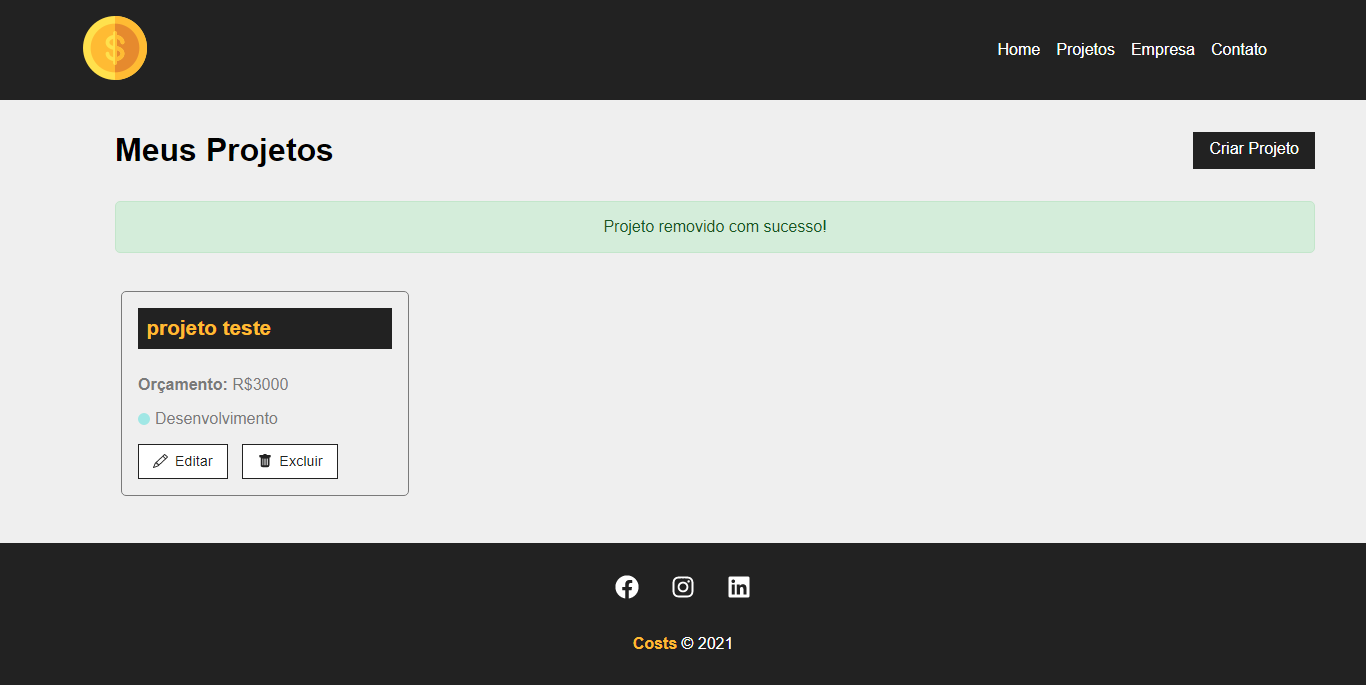
</Container>

</div>

)

}

export default Projects



# Aula 30 - Página de edição de dados

**src\components\pages\Project.js**

import styles from './Project.module.css'

import { useEffect, useState } from 'react'

import { useParams } from 'react-router-dom'

function Project(){

const {id} = useParams()

console.log(id)

const [project, setProject] = useState([])

useEffect(() => {

fetch(`http://localhost:5000/projects/${id}`, {

method: "GET",

headers: {

'Content-Type': 'application/json'

}

}).then(resp => resp.json())

.then((data) => {

setProject(data)

})

.catch((err) => console.log(err))

}, [id]);

return (

<div>

<p>{project.name}</p>

</div>

)

}

export default Project

**src\components\project\ProjectCard.js**

import { BsFillTrashFill, BsPencil } from 'react-icons/bs'

import { Link } from 'react-router-dom'

import styles from './ProjectCard.module.css'

function ProjectCard({id, name, budget, category, handleRemove}){

const remove = (e) => {

e.preventDefault()

handleRemove(id)

}

return(

<div className={styles.project\_card}>

<h4>{name}</h4>

<p><span>Orçamento:</span> R${budget}</p>

<p className={styles.category\_text}>

<span className={`${styles[category.toLowerCase()]}`}></span> {category}

</p>

<div className={styles.project\_card\_actions}>

<Link to={`/project/${id}`}>

<BsPencil /> Editar

</Link>

<button onClick={remove}>

<BsFillTrashFill /> Excluir

</button>

</div>

</div>

)

}

export default ProjectCard

**src\App.js**

import { BrowserRouter as Router, Route, Routes } from 'react-router-dom'

import Container from './components/layout/Container'

import Footer from './components/layout/Footer'

import Navbar from './components/layout/Navbar'

import Company from './components/pages/Company'

import Contact from './components/pages/Contact'

import Home from './components/pages/Home'

import NewProject from './components/pages/NewProject'

import Project from './components/pages/Project'

import Projects from './components/pages/Projects'

function App() {

return (

<Router>

<Navbar />

<Container customClass="min-height">

<Routes>

<Route path="/" element={<Home />} />

<Route path="/projects" element={<Projects />} />

<Route path="/company" element={<Company />} />

<Route path="/contact" element={<Contact />} />

<Route path="/newproject" element={<NewProject />} />

<Route path="/project/:id" element={<Project />} />

</Routes>

</Container>

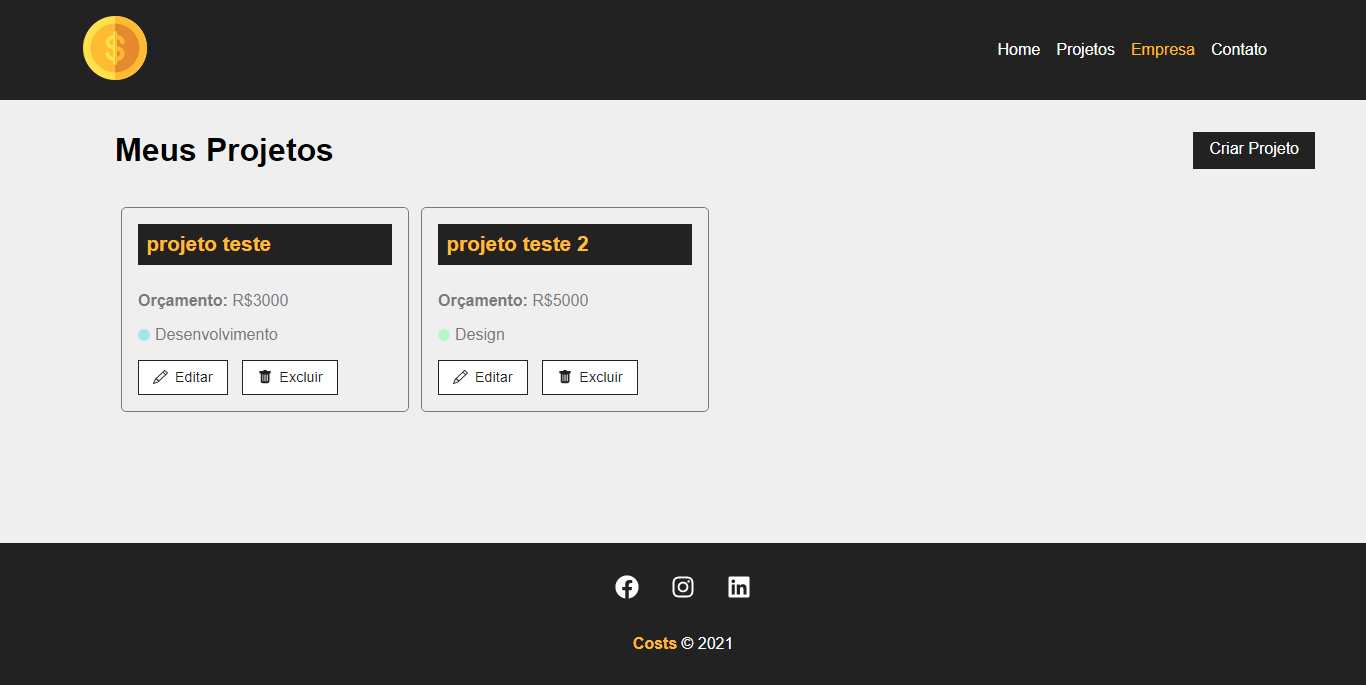
<Footer />

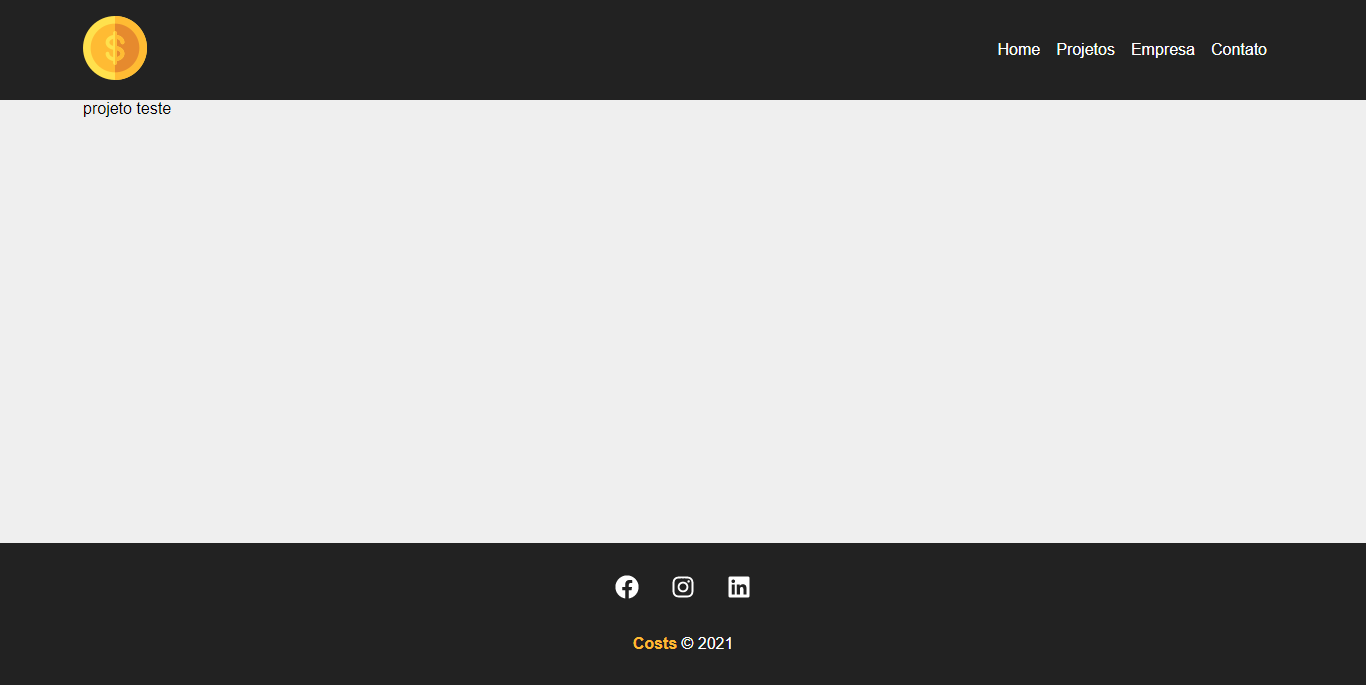
</Router>

);

}

export default App;





# Aula 31 - Exibindo detalhes do projeto

**src\components\pages\Project.js**

import { useEffect, useState } from 'react'

import { useParams } from 'react-router-dom'

import Container from '../layout/Container'

import Loading from '../layout/Loading'

import styles from './Project.module.css'

function Project(){

const {id} = useParams()

console.log(id)

const [project, setProject] = useState([])

const [showProjectForm, setShowProjectForm] = useState(false)

useEffect(() => {

setTimeout(() => {

fetch(`http://localhost:5000/projects/${id}`, {

method: "GET",

headers: {

'Content-Type': 'application/json'

}

}).then(resp => resp.json())

.then((data) => {

setProject(data)

})

.catch((err) => console.log(err))

}, 300);

}, [id]);

function toggleProjectForm(){

setShowProjectForm(!showProjectForm)

}

return (

<>

{project.name ? (

<div className={styles.project\_details}>

<Container customClass="column">

<div className={styles.details\_container}>

<h1>Projeto: {project.name}</h1>

<button className={styles.btn} onClick={toggleProjectForm}>

{!showProjectForm ? 'Editar projeto' : 'Fechar'}

</button>

{!showProjectForm ? (

<div className={styles.project\_info}>

<p>

<span>Categoria:</span> {project.category.name}

</p>

<p>

<span>Total de orçamento:</span> R${project.budget}

</p>

<p>

<span>Total utilizado:</span> R${project.cost}

</p>

</div>

) : (

<div className={styles.project\_info}>

<p>detalhes do projeto</p>

</div>

)}

</div>

</Container>

</div>

) : (

<Loading />

)}

</>

)

}

export default Project

**src\components\pages\Project.module.css**

.project\_details{

padding: 2em;

}

.project\_details h1,

.project\_details h2,

.project\_details p{

margin-bottom: 0.5em;

}

.project\_details h1{

background-color: #222;

color: #ffbb33;

padding: 0.4em;

}

.project\_details span{

font-weight: bold;

}

.details\_container,

.service\_form\_container{

border-bottom: 1px solid #7a7a7a;

margin-bottom: 1.2em;

padding-bottom: 1.2em;

display: flex;

justify-content: space-between;

flex-wrap: wrap;

}

.btn{

background-color: #222;

color: #fff;

padding: .5em 1em;

text-decoration: none;

transition: 0.5s;

cursor: pointer;

max-height: 40px;

border: none;

}

.btn:hover{

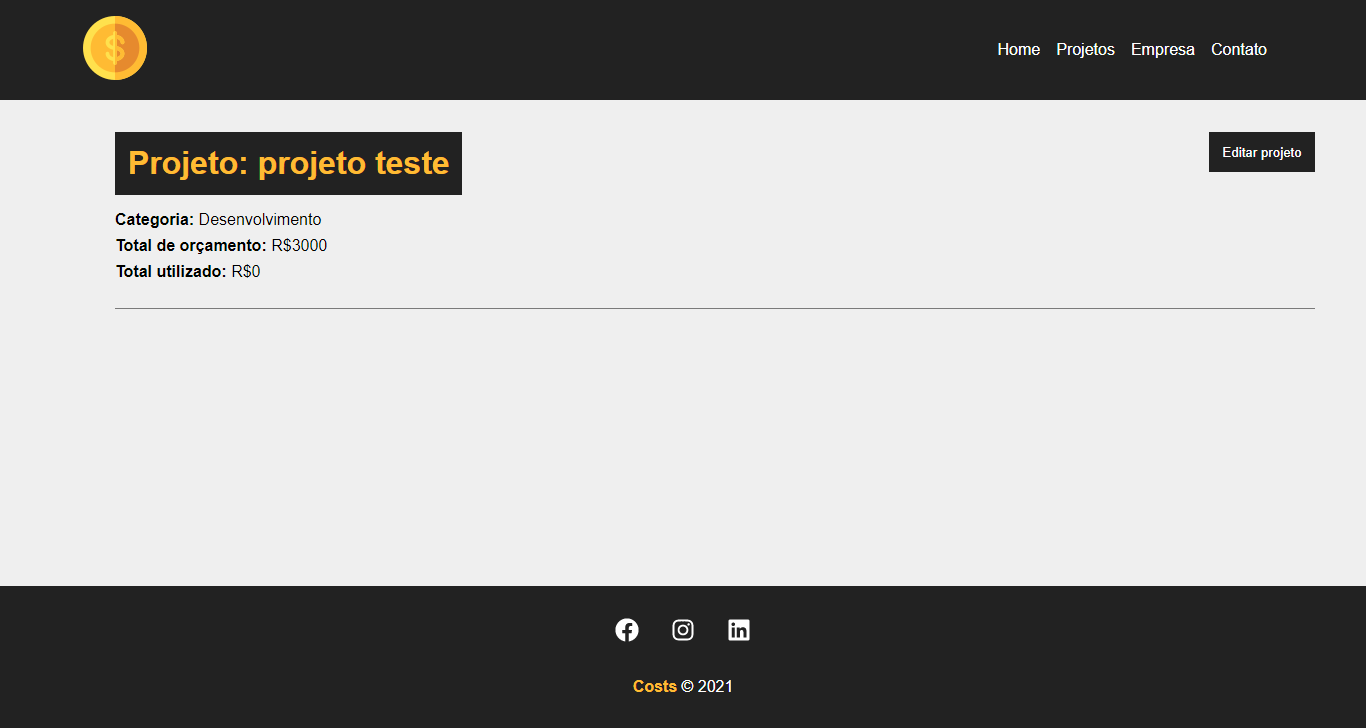
color: #ffbb33;

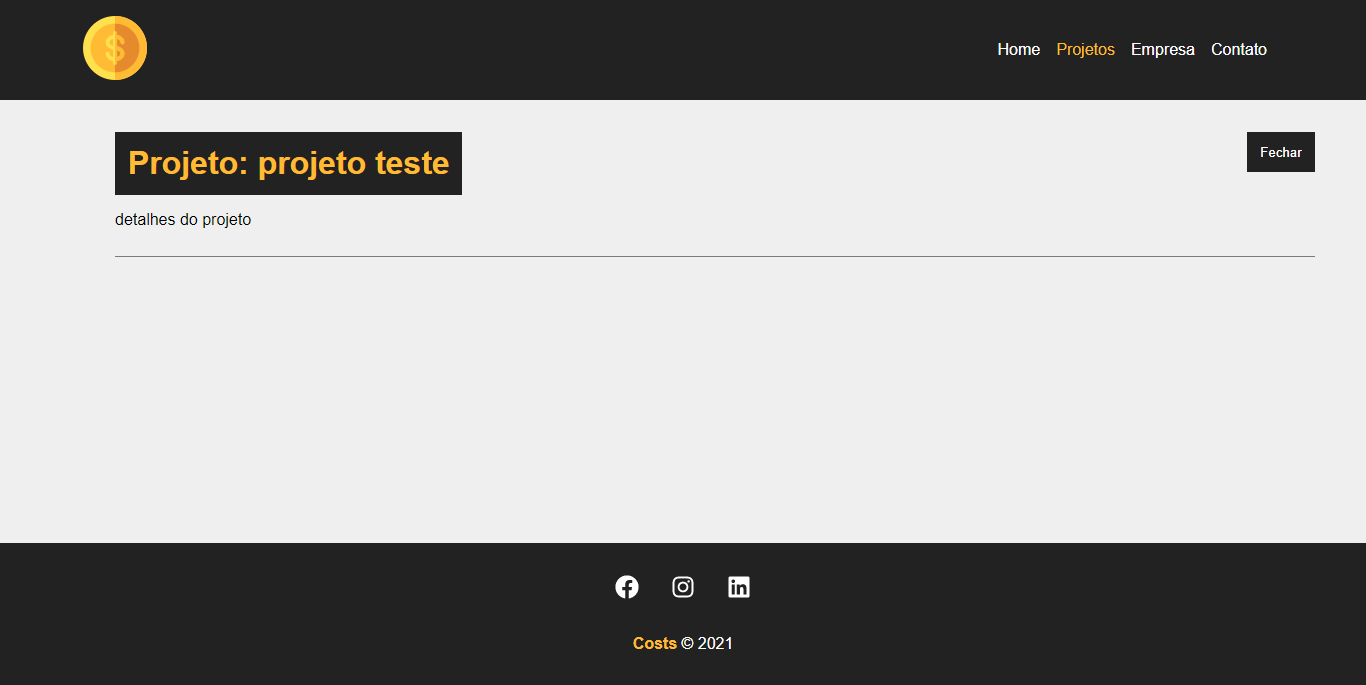
}

.project\_info{

width: 100%;

}





# Aula 32 - Atualização do projeto

**src\components\pages\Project.js**

import { useEffect, useState } from 'react'

import { useParams } from 'react-router-dom'

import Container from '../layout/Container'

import Loading from '../layout/Loading'

import Message from '../layout/Message'

import ProjectForm from '../project/ProjectForm'

import styles from './Project.module.css'

function Project(){

const {id} = useParams()

console.log(id)

const [project, setProject] = useState([])

const [showProjectForm, setShowProjectForm] = useState(false)

const [message, setMessage] = useState()

const [type, setType] = useState()

useEffect(() => {

setTimeout(() => {

fetch(`http://localhost:5000/projects/${id}`, {

method: "GET",

headers: {

'Content-Type': 'application/json'

}

}).then(resp => resp.json())

.then((data) => {

setProject(data)

})

.catch((err) => console.log(err))

}, 300);

}, [id]);

function editPost(project){

setMessage('')

// budget validation

if(project.budget < project.cost){

setMessage("O orçamento não pode ser menor que o custo do projeto!")

setType("error")

return false

}

fetch(`http://localhost:5000/projects/${project.id}`, {

method: 'PATCH',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(project),

}).then(resp => resp.json())

.then((data) => {

setProject(data)

setShowProjectForm(false)

setMessage("Projeto atualizado!")

setType("success")

})

.catch((err) => console.log(err))

}

function toggleProjectForm(){

setShowProjectForm(!showProjectForm)

}

return (

<>

{project.name ? (

<div className={styles.project\_details}>

<Container customClass="column">

{message && <Message type={type} msg={message} />}

<div className={styles.details\_container}>

<h1>Projeto: {project.name}</h1>

<button className={styles.btn} onClick={toggleProjectForm}>

{!showProjectForm ? 'Editar projeto' : 'Fechar'}

</button>

{!showProjectForm ? (

<div className={styles.project\_info}>

<p>

<span>Categoria:</span> {project.category.name}

</p>

<p>

<span>Total de orçamento:</span> R${project.budget}

</p>

<p>

<span>Total utilizado:</span> R${project.cost}

</p>

</div>

) : (

<div className={styles.project\_info}>

<ProjectForm

handleSubmit={editPost}

btnText="Concluir edição"

projectData={project}

/>

</div>

)}

</div>

</Container>

</div>

) : (

<Loading />

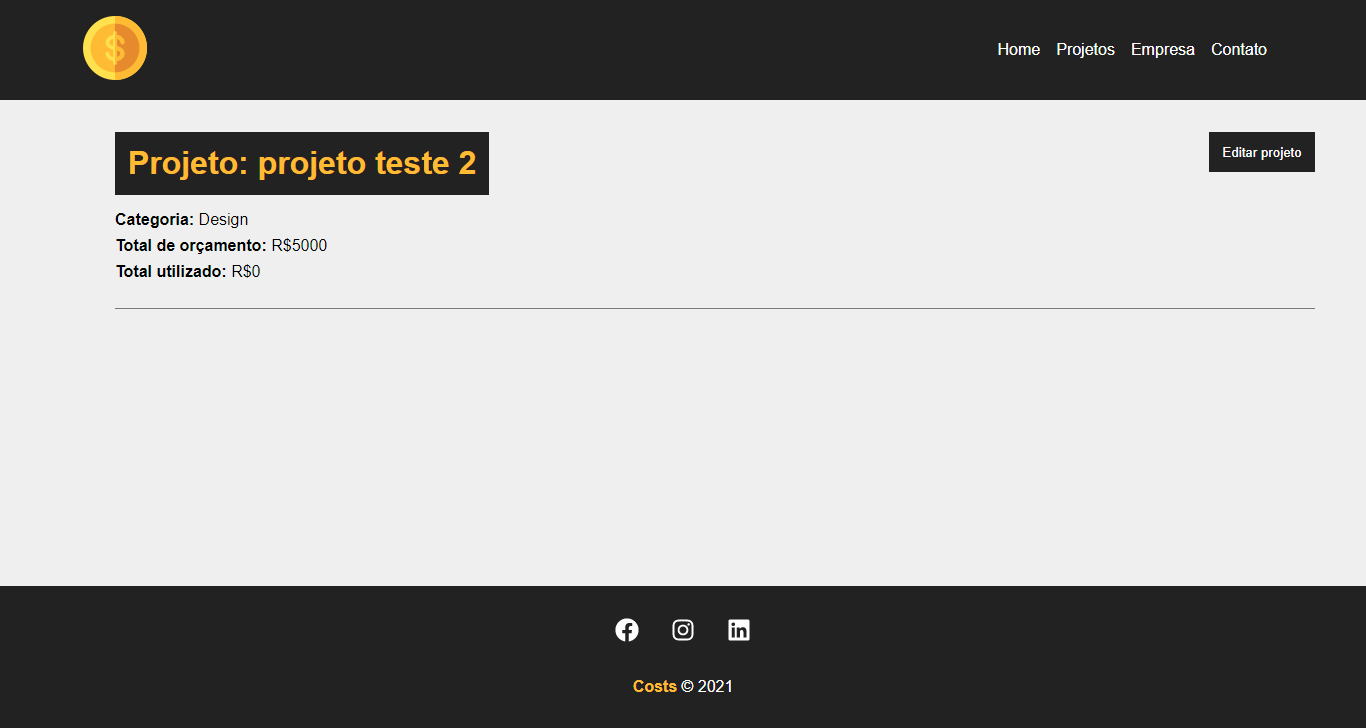
)}

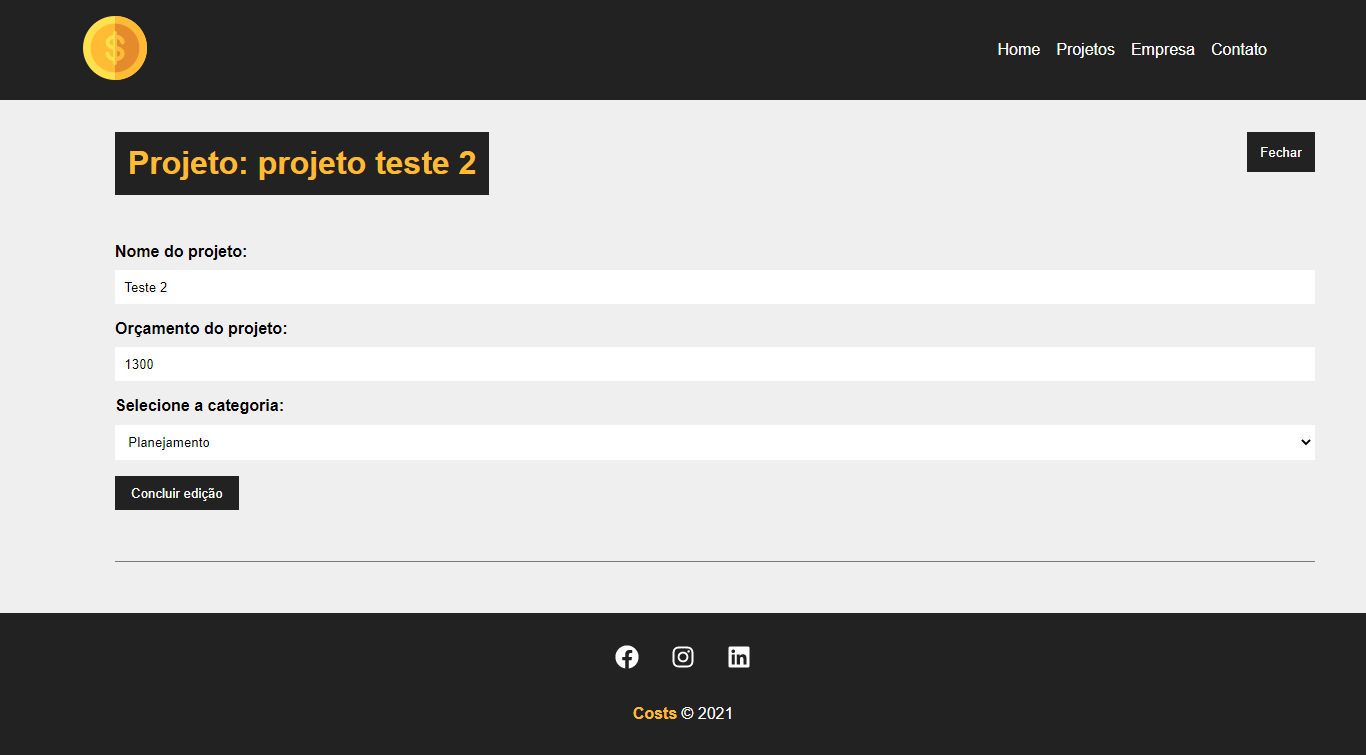
</>

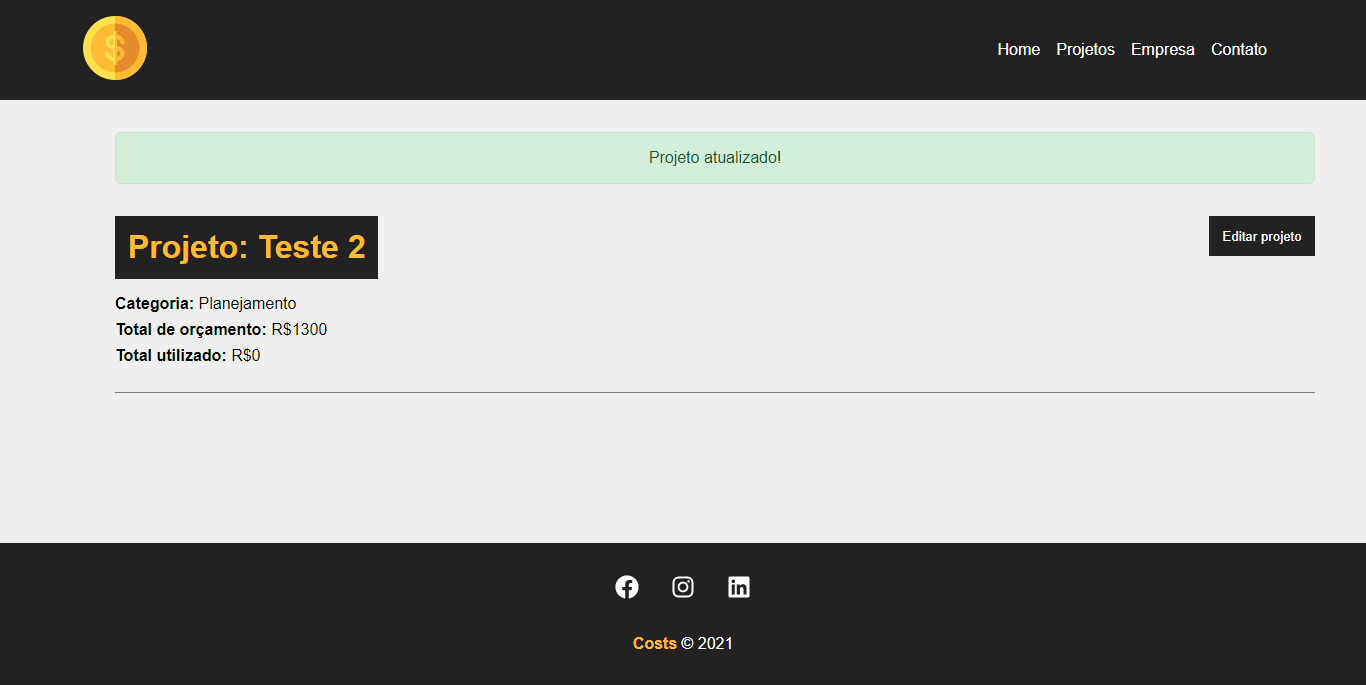
)

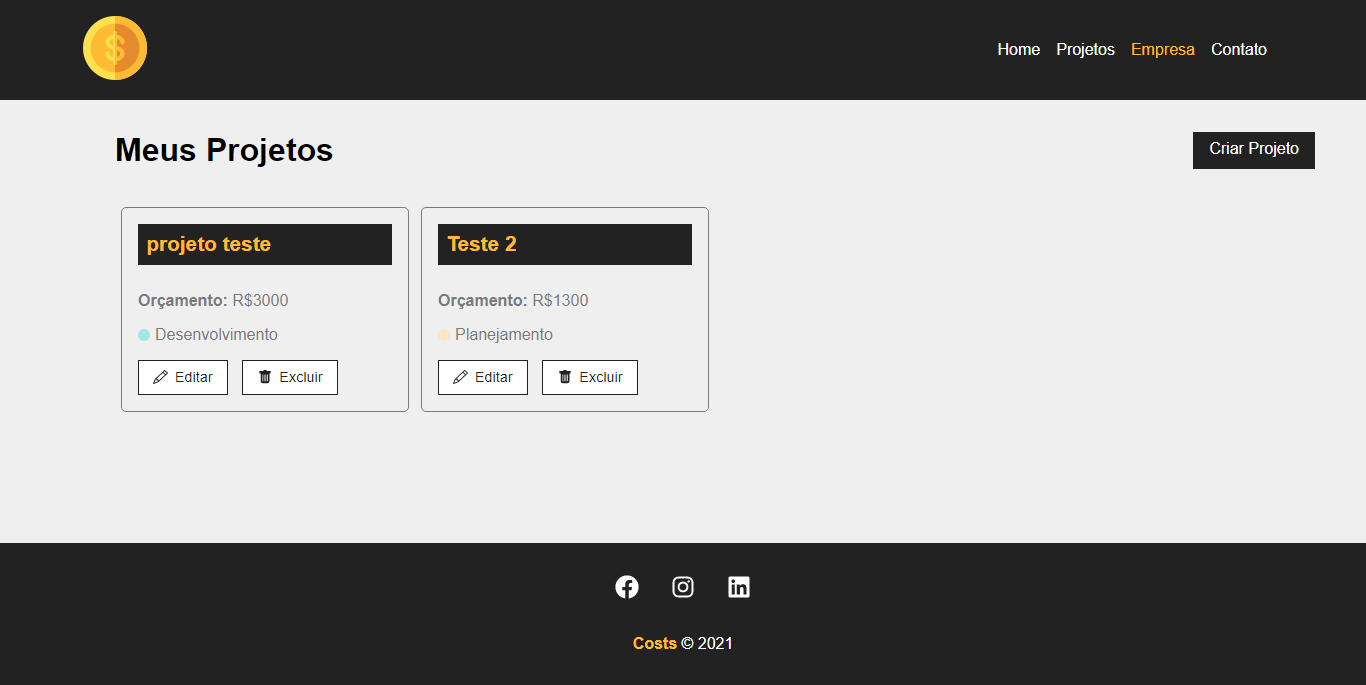
}

export default Project









# Aula 33 - Criando área de serviços

**src\components\pages\Project.js**

import { useEffect, useState } from 'react'

import { useParams } from 'react-router-dom'

import Container from '../layout/Container'

import Loading from '../layout/Loading'

import Message from '../layout/Message'

import ProjectForm from '../project/ProjectForm'

import styles from './Project.module.css'

function Project(){

const {id} = useParams()

console.log(id)

const [project, setProject] = useState([])

const [showProjectForm, setShowProjectForm] = useState(false)

const [showServiceForm, setShowServiceForm] = useState(false)

const [message, setMessage] = useState()

const [type, setType] = useState()

useEffect(() => {

setTimeout(() => {

fetch(`http://localhost:5000/projects/${id}`, {

method: "GET",

headers: {

'Content-Type': 'application/json'

}

}).then(resp => resp.json())

.then((data) => {

setProject(data)

})

.catch((err) => console.log(err))

}, 300);

}, [id]);

function editPost(project){

setMessage('')

// budget validation

if(project.budget < project.cost){

setMessage("O orçamento não pode ser menor que o custo do projeto!")

setType("error")

return false

}

fetch(`http://localhost:5000/projects/${project.id}`, {

method: 'PATCH',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(project),

}).then(resp => resp.json())

.then((data) => {

setProject(data)

setShowProjectForm(false)

setMessage("Projeto atualizado!")

setType("success")

})

.catch((err) => console.log(err))

}

function toggleProjectForm(){

setShowProjectForm(!showProjectForm)

}

function toggleServiceForm() {

setShowServiceForm(!showServiceForm)

}

return (

<>

{project.name ? (

<div className={styles.project\_details}>

<Container customClass="column">

{message && <Message type={type} msg={message} />}

<div className={styles.details\_container}>

<h1>Projeto: {project.name}</h1>

<button className={styles.btn} onClick={toggleProjectForm}>

{!showProjectForm ? 'Editar projeto' : 'Fechar'}

</button>

{!showProjectForm ? (

<div className={styles.project\_info}>

<p>

<span>Categoria:</span> {project.category.name}

</p>

<p>

<span>Total de orçamento:</span> R${project.budget}

</p>

<p>

<span>Total utilizado:</span> R${project.cost}

</p>

</div>

) : (

<div className={styles.project\_info}>

<ProjectForm

handleSubmit={editPost}

btnText="Concluir edição"

projectData={project}

/>

</div>

)}

</div>

<div className={styles.service\_form\_container}>

<h2>Adicione um serviço</h2>

<button className={styles.btn} onClick={toggleServiceForm}>

{!showServiceForm ? 'Adicionar serviço' : 'Fechar'}

</button>

<div className={styles.project\_info}>

{showServiceForm && (

<div>

formulário do serviço

</div>

)}

</div>

</div>

<h2>Serviços</h2>

<Container customClass="start">

<p>Itens de serviços</p>

</Container>

</Container>

</div>

) : (

<Loading />

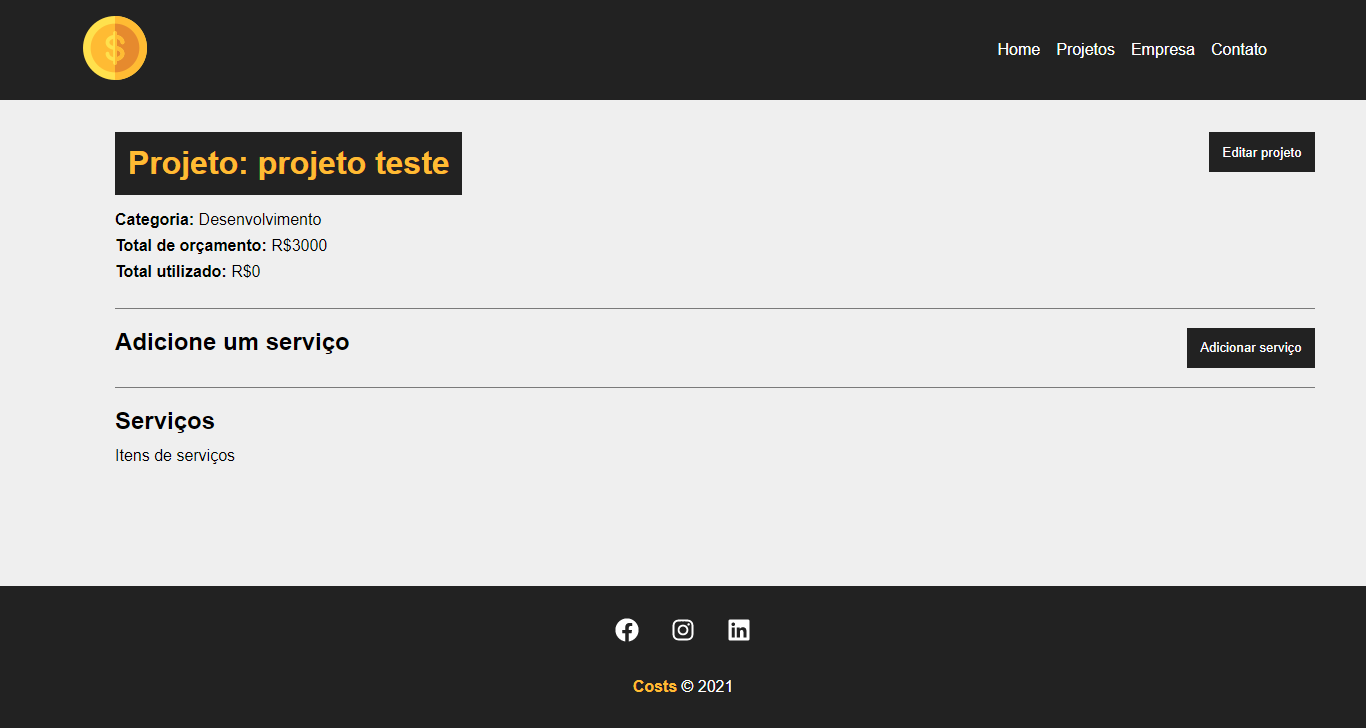
)}

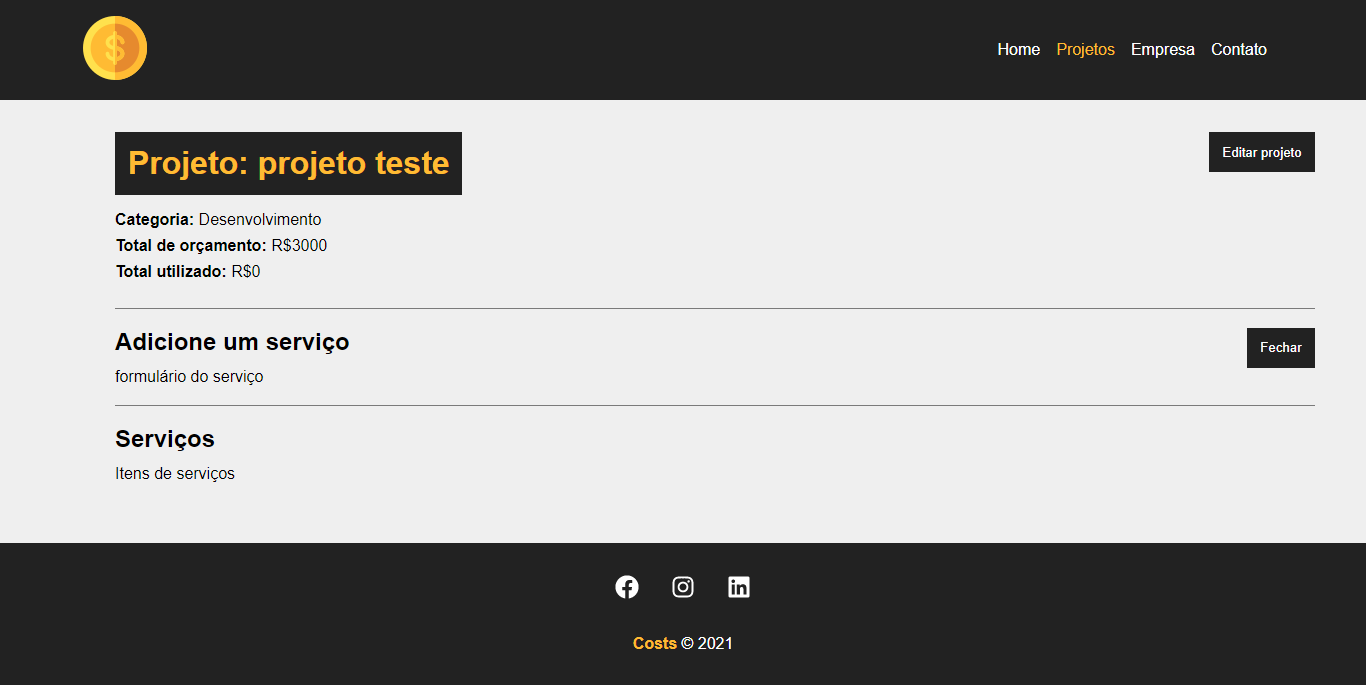
</>

)

}

export default Project





# Aula 34 - Adicionando novos serviços

**src\components\pages\Project.js**

import { useEffect, useState } from 'react'

import { useParams } from 'react-router-dom'

import { v4 as uuidv4 } from 'uuid'

import Container from '../layout/Container'

import Loading from '../layout/Loading'

import Message from '../layout/Message'

import ProjectForm from '../project/ProjectForm'

import ServiceForm from '../service/ServiceForm'

import styles from './Project.module.css'

function Project(){

const {id} = useParams()

console.log(id)

const [project, setProject] = useState([])

const [showProjectForm, setShowProjectForm] = useState(false)

const [showServiceForm, setShowServiceForm] = useState(false)

const [message, setMessage] = useState()

const [type, setType] = useState()

useEffect(() => {

setTimeout(() => {

fetch(`http://localhost:5000/projects/${id}`, {

method: "GET",

headers: {

'Content-Type': 'application/json'

}

}).then(resp => resp.json())

.then((data) => {

setProject(data)

})

.catch((err) => console.log(err))

}, 300);

}, [id]);

function editPost(project){

setMessage('')

// budget validation

if(project.budget < project.cost){

setMessage("O orçamento não pode ser menor que o custo do projeto!")

setType("error")

return false

}

fetch(`http://localhost:5000/projects/${project.id}`, {

method: 'PATCH',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(project),

}).then(resp => resp.json())

.then((data) => {

setProject(data)

setShowProjectForm(false)

setMessage("Projeto atualizado!")

setType("success")

})

.catch((err) => console.log(err))

}

function createService(project){

setMessage('')

// last service

const lastService = project.services[project.services.length - 1]

lastService.id = uuidv4()

const lastServiceCost = lastService.cost

const newCost = parseFloat(project.cost) + parseFloat(lastServiceCost)

// maximum value validation

if(newCost > parseFloat(project.budget)){

setMessage("Orçamento ultrapassado! Verifique o valor do serviço")

setType('error')

project.services.pop()

return false

}

// add service cost to project total cost

project.cost = newCost

// update project

fetch(`http://localhost:5000/projects/${project.id}`, {

method: 'PATCH',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(project)

}).then((resp) => resp.json())

.then((data) => {

// exibir os serviços

console.log(data)

})

.catch((err) => console.log(err))

}

function toggleProjectForm(){

setShowProjectForm(!showProjectForm)

}

function toggleServiceForm() {

setShowServiceForm(!showServiceForm)

}

return (

<>

{project.name ? (

<div className={styles.project\_details}>

<Container customClass="column">

{message && <Message type={type} msg={message} />}

<div className={styles.details\_container}>

<h1>Projeto: {project.name}</h1>

<button className={styles.btn} onClick={toggleProjectForm}>

{!showProjectForm ? 'Editar projeto' : 'Fechar'}

</button>

{!showProjectForm ? (

<div className={styles.project\_info}>

<p>

<span>Categoria:</span> {project.category.name}

</p>

<p>

<span>Total de orçamento:</span> R${project.budget}

</p>

<p>

<span>Total utilizado:</span> R${project.cost}

</p>

</div>

) : (

<div className={styles.project\_info}>

<ProjectForm

handleSubmit={editPost}

btnText="Concluir edição"

projectData={project}

/>

</div>

)}

</div>

<div className={styles.service\_form\_container}>

<h2>Adicione um serviço</h2>

<button className={styles.btn} onClick={toggleServiceForm}>

{!showServiceForm ? 'Adicionar serviço' : 'Fechar'}

</button>

<div className={styles.project\_info}>

{showServiceForm && <ServiceForm

handleSubmit={createService}

btnText="Adicionar Serviço"

projectData={project}

/>}

</div>

</div>

<h2>Serviços</h2>

<Container customClass="start">

<p>Itens de serviços</p>

</Container>

</Container>

</div>

) : (

<Loading />

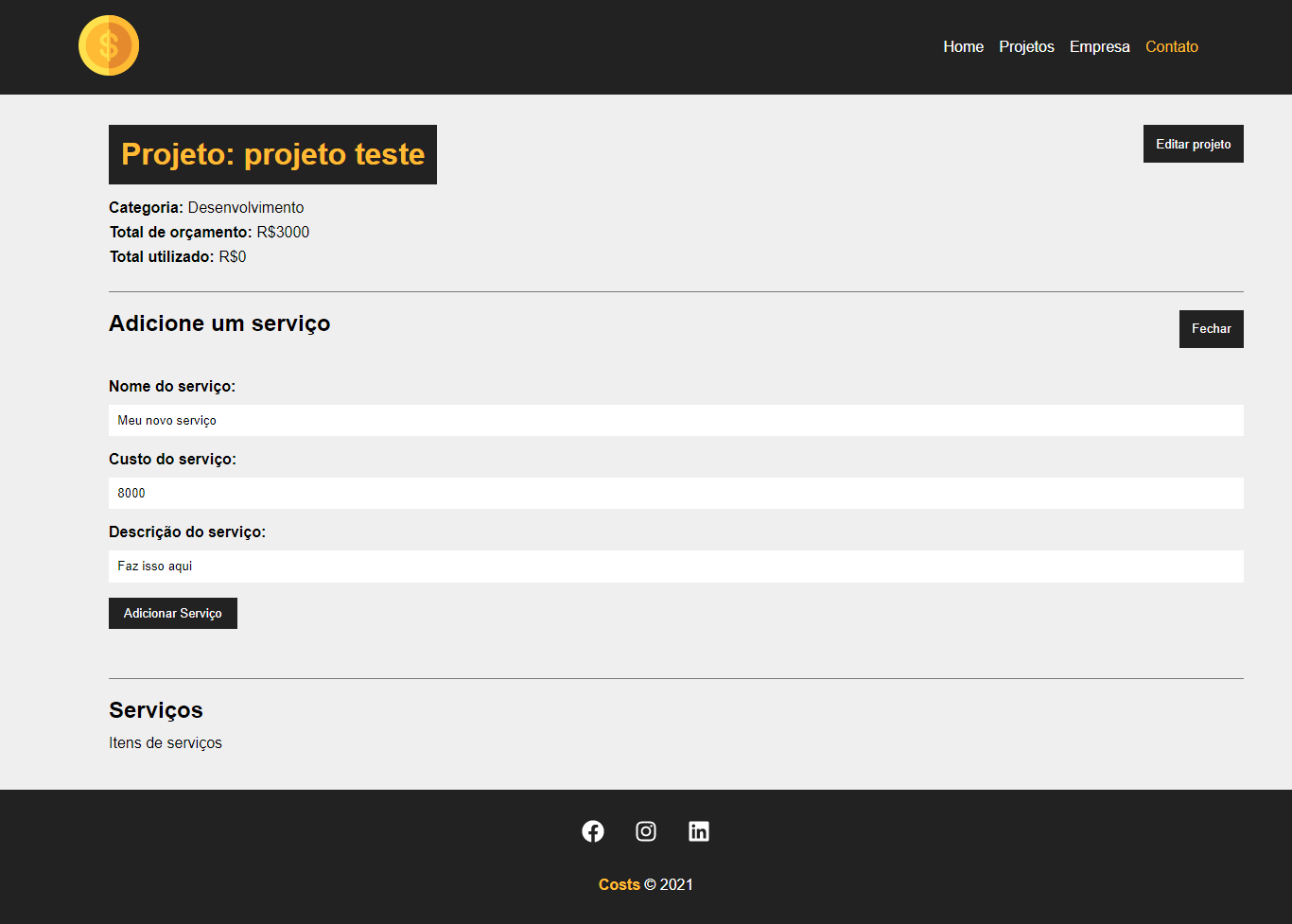
)}

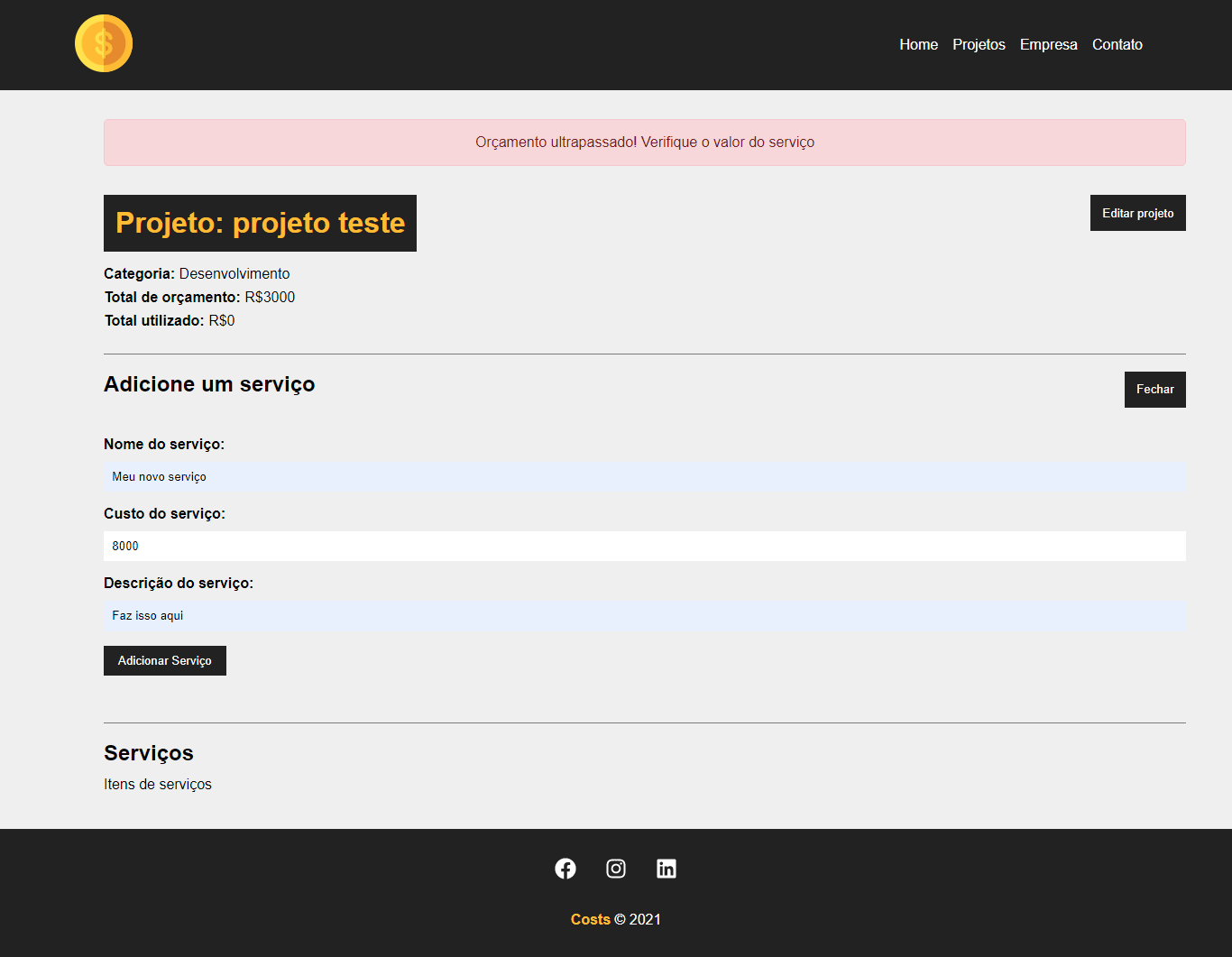
</>

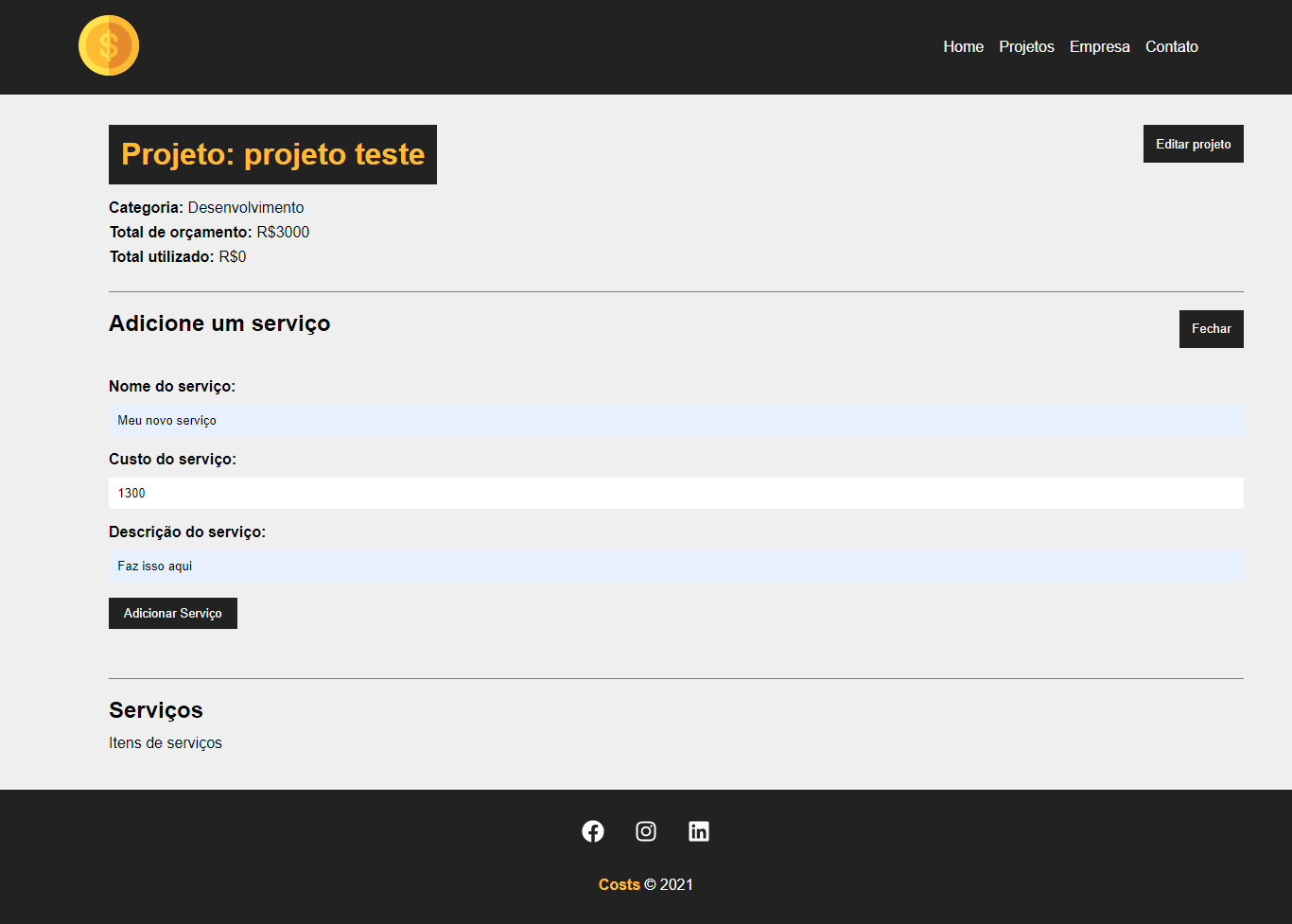
)

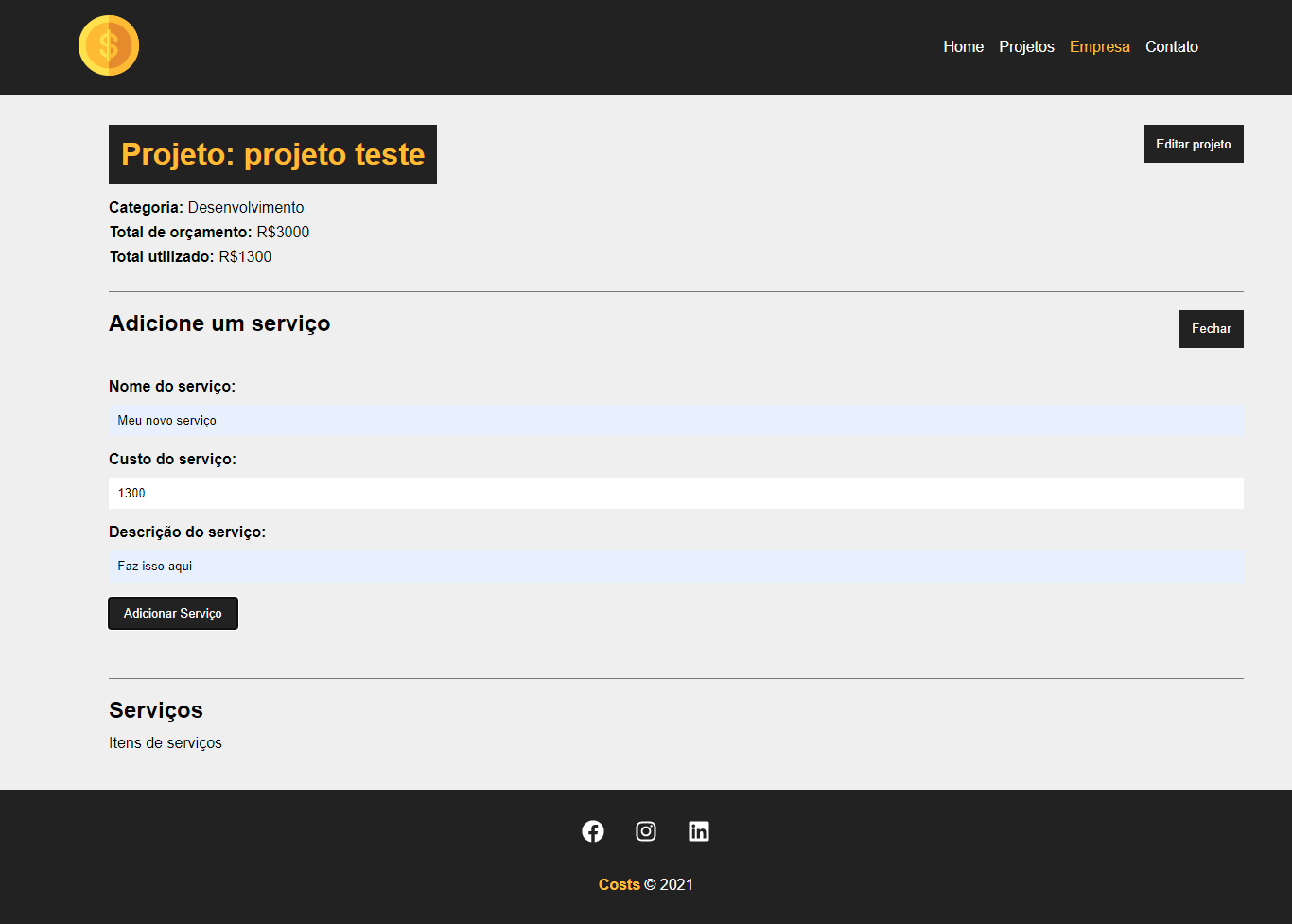
}

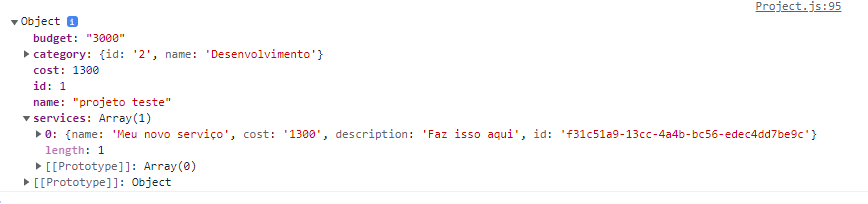
export default Project











# Aula 35 - Exibindo os serviços

**src\components\service\ServiceCard.js**

import { BsFillTrashFill } from 'react-icons/bs'

import styles from '../project/ProjectCard.module.css'

function ServiceCard({id, name, cost, description, handleRemove}){

const remove = (e) => {

}

return (

<div className={styles.project\_card}>

<h4>{name}</h4>

<p>

<span>Custo total:</span> R${cost}

</p>

<p>{description}</p>

<div className={styles.project\_card\_actions}>

<button onClick={remove}>

<BsFillTrashFill /> Excluir

</button>

</div>

</div>

)

}

export default ServiceCard

**src\components\pages\Project.js**

import { useEffect, useState } from 'react'

import { useParams } from 'react-router-dom'

import { v4 as uuidv4 } from 'uuid'

import Container from '../layout/Container'

import Loading from '../layout/Loading'

import Message from '../layout/Message'

import ProjectForm from '../project/ProjectForm'

import ServiceCard from '../service/ServiceCard'

import ServiceForm from '../service/ServiceForm'

import styles from './Project.module.css'

function Project(){

const {id} = useParams()

console.log(id)

const [project, setProject] = useState([])

const [services, setServices] = useState([])

const [showProjectForm, setShowProjectForm] = useState(false)

const [showServiceForm, setShowServiceForm] = useState(false)

const [message, setMessage] = useState()

const [type, setType] = useState()

useEffect(() => {

setTimeout(() => {

fetch(`http://localhost:5000/projects/${id}`, {

method: "GET",

headers: {

'Content-Type': 'application/json'

}

}).then(resp => resp.json())

.then((data) => {

setProject(data)

setServices(data.services)

})

.catch((err) => console.log(err))

}, 300);

}, [id]);

function editPost(project){

setMessage('')

// budget validation

if(project.budget < project.cost){

setMessage("O orçamento não pode ser menor que o custo do projeto!")

setType("error")

return false

}

fetch(`http://localhost:5000/projects/${project.id}`, {

method: 'PATCH',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(project),

}).then(resp => resp.json())

.then((data) => {

setProject(data)

setShowProjectForm(false)

setMessage("Projeto atualizado!")

setType("success")

})

.catch((err) => console.log(err))

}

function createService(project){

setMessage('')

// last service

const lastService = project.services[project.services.length - 1]

lastService.id = uuidv4()

const lastServiceCost = lastService.cost

const newCost = parseFloat(project.cost) + parseFloat(lastServiceCost)

// maximum value validation

if(newCost > parseFloat(project.budget)){

setMessage("Orçamento ultrapassado! Verifique o valor do serviço")

setType('error')

project.services.pop()

return false

}

// add service cost to project total cost

project.cost = newCost

// update project

fetch(`http://localhost:5000/projects/${project.id}`, {

method: 'PATCH',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(project)

}).then((resp) => resp.json())

.then((data) => {

// exibir os serviços

setShowServiceForm(false)

})

.catch((err) => console.log(err))

}

function removeService(){

}

function toggleProjectForm(){

setShowProjectForm(!showProjectForm)

}

function toggleServiceForm() {

setShowServiceForm(!showServiceForm)

}

return (

<>

{project.name ? (

<div className={styles.project\_details}>

<Container customClass="column">

{message && <Message type={type} msg={message} />}

<div className={styles.details\_container}>

<h1>Projeto: {project.name}</h1>

<button className={styles.btn} onClick={toggleProjectForm}>

{!showProjectForm ? 'Editar projeto' : 'Fechar'}

</button>

{!showProjectForm ? (

<div className={styles.project\_info}>

<p>

<span>Categoria:</span> {project.category.name}

</p>

<p>

<span>Total de orçamento:</span> R${project.budget}

</p>

<p>

<span>Total utilizado:</span> R${project.cost}

</p>

</div>

) : (

<div className={styles.project\_info}>

<ProjectForm

handleSubmit={editPost}

btnText="Concluir edição"

projectData={project}

/>

</div>

)}

</div>

<div className={styles.service\_form\_container}>

<h2>Adicione um serviço</h2>

<button className={styles.btn} onClick={toggleServiceForm}>

{!showServiceForm ? 'Adicionar serviço' : 'Fechar'}

</button>

<div className={styles.project\_info}>

{showServiceForm && <ServiceForm

handleSubmit={createService}

btnText="Adicionar Serviço"

projectData={project}

/>}

</div>

</div>

<h2>Serviços</h2>

<Container customClass="start">

{services.length > 0 &&

services.map((service) => (

<ServiceCard

id={service.id}

name={service.name}

cost={service.cost}

description={service.description}

key={service.id}

handleRemove={removeService}

/>

))

}

{services.length === 0 && <p>Não há serviços cadastrados!</p>}

</Container>

</Container>

</div>

) : (

<Loading />

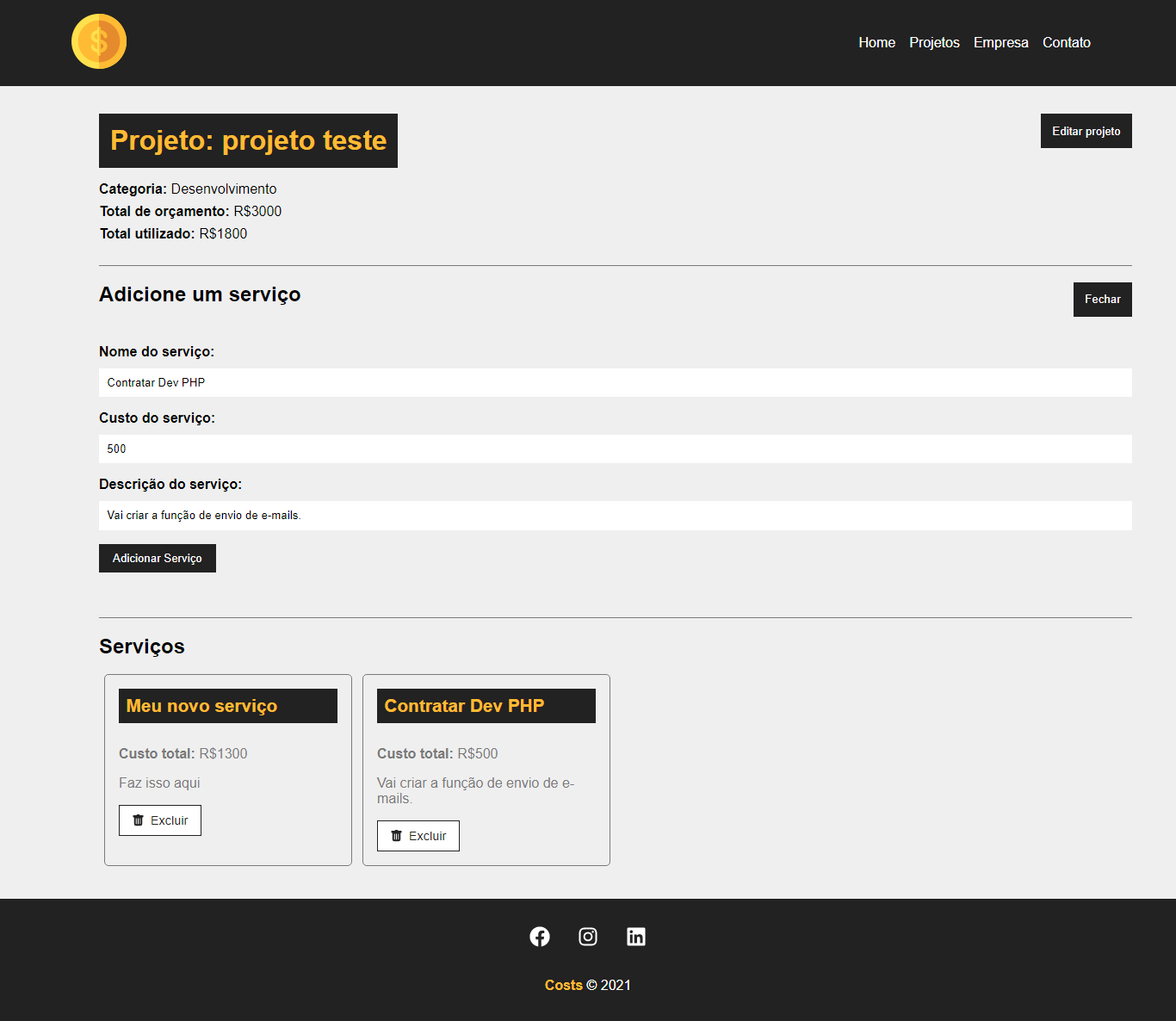
)}

</>

)

}

export default Project



# Aula 36 - Removendo serviços

**src\components\service\ServiceCard.js**

import { BsFillTrashFill } from 'react-icons/bs'

import styles from '../project/ProjectCard.module.css'

function ServiceCard({id, name, cost, description, handleRemove}){

const remove = (e) => {

e.preventDefault()

handleRemove(id, cost)

}

return (

<div className={styles.project\_card}>

<h4>{name}</h4>

<p>

<span>Custo total:</span> R${cost}

</p>

<p>{description}</p>

<div className={styles.project\_card\_actions}>

<button onClick={remove}>

<BsFillTrashFill /> Excluir

</button>

</div>

</div>

)

}

export default ServiceCard

**src\components\pages\Project.js**

import { useEffect, useState } from 'react'

import { useParams } from 'react-router-dom'

import { v4 as uuidv4 } from 'uuid'

import Container from '../layout/Container'

import Loading from '../layout/Loading'

import Message from '../layout/Message'

import ProjectForm from '../project/ProjectForm'

import ServiceCard from '../service/ServiceCard'

import ServiceForm from '../service/ServiceForm'

import styles from './Project.module.css'

function Project(){

const {id} = useParams()

console.log(id)

const [project, setProject] = useState([])

const [services, setServices] = useState([])

const [showProjectForm, setShowProjectForm] = useState(false)

const [showServiceForm, setShowServiceForm] = useState(false)

const [message, setMessage] = useState()

const [type, setType] = useState()

useEffect(() => {

setTimeout(() => {

fetch(`http://localhost:5000/projects/${id}`, {

method: "GET",

headers: {

'Content-Type': 'application/json'

}

}).then(resp => resp.json())

.then((data) => {

setProject(data)

setServices(data.services)

})

.catch((err) => console.log(err))

}, 300);

}, [id]);

function editPost(project){

setMessage('')

// budget validation

if(project.budget < project.cost){

setMessage("O orçamento não pode ser menor que o custo do projeto!")

setType("error")

return false

}

fetch(`http://localhost:5000/projects/${project.id}`, {

method: 'PATCH',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(project),

}).then(resp => resp.json())

.then((data) => {

setProject(data)

setShowProjectForm(false)

setMessage("Projeto atualizado!")

setType("success")

})

.catch((err) => console.log(err))

}

function createService(project){

setMessage('')

// last service

const lastService = project.services[project.services.length - 1]

lastService.id = uuidv4()

const lastServiceCost = lastService.cost

const newCost = parseFloat(project.cost) + parseFloat(lastServiceCost)

// maximum value validation

if(newCost > parseFloat(project.budget)){

setMessage("Orçamento ultrapassado! Verifique o valor do serviço")

setType('error')

project.services.pop()

return false

}

// add service cost to project total cost

project.cost = newCost

// update project

fetch(`http://localhost:5000/projects/${project.id}`, {

method: 'PATCH',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(project)

}).then((resp) => resp.json())

.then((data) => {

// exibir os serviços

setShowServiceForm(false)

})

.catch((err) => console.log(err))

}

function removeService(id, cost){

const servicesUpdated = project.services.filter(

(service) => service.id !== id

)

const projectUpdated = project

projectUpdated.services = servicesUpdated

projectUpdated.cost = parseFloat(projectUpdated.cost) - parseFloat(cost)

fetch(`http://localhost:5000/projects/${projectUpdated.id}`, {

method: "PATCH",

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(projectUpdated)

}).then((resp) => resp.json())

.then((data) => {

setProject(projectUpdated)

setServices(servicesUpdated)

setMessage("Serviço removido com sucesso!")

setType("success")

})

.catch((err) => console.log(err))

}

function toggleProjectForm(){

setShowProjectForm(!showProjectForm)

}

function toggleServiceForm() {

setShowServiceForm(!showServiceForm)

}

return (

<>

{project.name ? (

<div className={styles.project\_details}>

<Container customClass="column">

{message && <Message type={type} msg={message} />}

<div className={styles.details\_container}>

<h1>Projeto: {project.name}</h1>

<button className={styles.btn} onClick={toggleProjectForm}>

{!showProjectForm ? 'Editar projeto' : 'Fechar'}

</button>

{!showProjectForm ? (

<div className={styles.project\_info}>

<p>

<span>Categoria:</span> {project.category.name}

</p>

<p>

<span>Total de orçamento:</span> R${project.budget}

</p>

<p>

<span>Total utilizado:</span> R${project.cost}

</p>

</div>

) : (

<div className={styles.project\_info}>

<ProjectForm

handleSubmit={editPost}

btnText="Concluir edição"

projectData={project}

/>

</div>

)}

</div>

<div className={styles.service\_form\_container}>

<h2>Adicione um serviço</h2>

<button className={styles.btn} onClick={toggleServiceForm}>

{!showServiceForm ? 'Adicionar serviço' : 'Fechar'}

</button>

<div className={styles.project\_info}>

{showServiceForm && <ServiceForm

handleSubmit={createService}

btnText="Adicionar Serviço"

projectData={project}

/>}

</div>

</div>

<h2>Serviços</h2>

<Container customClass="start">

{services.length > 0 &&

services.map((service) => (

<ServiceCard

id={service.id}

name={service.name}

cost={service.cost}

description={service.description}

key={service.id}

handleRemove={removeService}

/>

))

}

{services.length === 0 && <p>Não há serviços cadastrados!</p>}

</Container>

</Container>

</div>

) : (

<Loading />

)}

</>

)

}

export default Project

