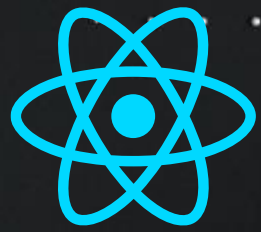
FIAP

RESPONSIVE WEB DEVELOPMENT



REACT - ROUTER

Prof. Alexandre Carlos profalexandre.jesus@fiap.com.br

Prof. Luís Carlos lsilva@fiap.com.br

Prof. Wellington Cidade profwellington.tenorio@fiap.com.br

REACT - ROTAS

FIAP

O QUE SÃO ROTAS?

Em nossas aplicações com o React, não podemos simplesmente criar links em nossas aplicações para direcionar para outras páginas internas, como fazemos em projetos com HTML simples. O React é um **Single Page Application**, ou seja, uma aplicação que **funciona em uma única página**. Isso é possível porque ele cria uma página a partir da montagem de componentes, que podem tanto fazer o papel de um container, quanto de um simples botão.

Quando trabalhamos com rotas, podemos definir quais componentes irão compor a página, dando um novo caminho (URI) para eles, sendo assim, **cada rota será uma página**, e podemos criar quantas páginas forem necessárias. O usuário nem irá perceber o que está acontecendo.

REACT - ROTAS

CRIANDO A APLICAÇÃO

Para conseguirmos criar rotas em nossa aplicação vamos precisar importar o pacote de uma biblioteca a **react-router-dom**, quando criar o projeto, siga a sequência abaixo:

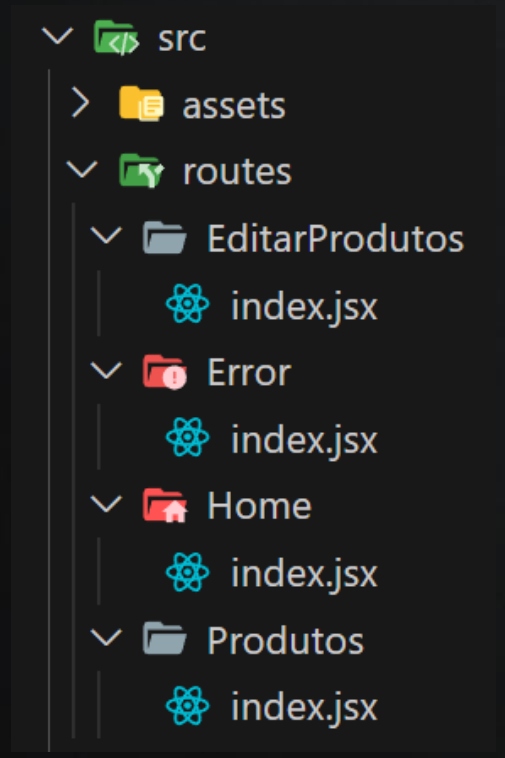
1. `npm create vite@latest aula-rotas -- --template react`
2. `cd aula-rotas`
3. `npm install`
4. `npm install react-router-dom` **Instala o pacote da biblioteca das rotas**
5. `npm run dev`

REACT - ROTAS

CRIANDO AS PÁGINAS DO NOSSO EXEMPLO

Quando criamos os componentes que representarão as nossas páginas devemos fazer algumas boas práticas para organizarmos melhor nosso projeto:

1. Vamos criar uma pasta chamada **routes**, ela irá guardas as páginas que vamos criar.
2. Para cada página vamos criar uma pasta com o nome da página e dentro um arquivo chamado index.jsx;



Teremos as páginas:

- Home
- Produtos
- EditarProdutos
- Error

REACT - ROTAS

CONTEÚDO INICIAL DOS COMPONENTES

Home

```
export default function Home(){

    return(
        <main>
            <h1>HOME</h1>
        </main>
    )
}
```

Produtos

```
export default function Produtos(){

    return(
        <main>
            <h1>Produtos</h1>
        </main>
)
}
```

EditarProduto

```
export default function EditarProduto(){

    return(
        <main>
            <h1>Editando o produto </h1>
        </main>
    )
}
```

Error

```
export default function Error(){

    return(
        <>
            <h1>Erro 404 - Página não encontrada</h1>
        </>
    )
}
```

REACT - ROTAS

IMPORTANDO O ELEMENTOS DO ROUTER E AS PÁGINAS NO MAIN.JSX

A primeira coisa que devemos fazer após criar as páginas é importar os elementos que iremos precisar do router e as páginas no main.jsx, será a partir de lá que iremos fazer esse controle.

```
import { createBrowserRouter, RouterProvider } from 'react-router-dom'
import Home from './routes/Home/index.jsx'
import Produtos from './routes/Produtos/index.jsx'
import Error from './routes/Error/index.jsx'
import EditarProduto from './routes/EditarProduto/index.jsx'
```

REACT - ROTAS

CONFIGURANDO AS ROTAS

Para configurarmos as rotas precisamos utilizar o método createBrowserRouter, ele recebe um objeto com os dados de App e dentro do atributo children, cada objeto representa uma das rotas possíveis em nosso projeto.

errorElement – define o componente que será apresentado caso o usuário coloque uma URI inexistente.

Children – atributo de App que armazena em um array os objetos de todas as páginas que podem ser acessadas.

Path - define a **URI** da página logo após o endereço do projeto no browser.

Element - define o **Componente** que será chamado quando a URI for requisitada no browser.

```
const router = createBrowserRouter([
  {
    path: "/",
    element: <App/>,
    errorElement: <Error/>,
    children:[
      {
        path: "/",
        element: <Home/>
      },
      {
        path: "/produtos",
        element: <Produtos/>
      },
      {
        path: "/produtos/editar/:id",
        element: <EditarProduto/>
      }
    ]
  }
])
```

REACT - ROTAS

FIAP

CONFIGURANDO AS ROTAS

Após configurarmos nosso createBrowserRouter, que demos o nome de router, vamos usar o **RouterProvider** para controlar a renderização das páginas no **ReactDOM**. Para isso precisamos também passar o nosso router como atributo.

```
ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <RouterProvider router={router} />
  </React.StrictMode>,
)
```

REACT - ROTAS

FIAP

CONFIGURANDO AS ROTAS

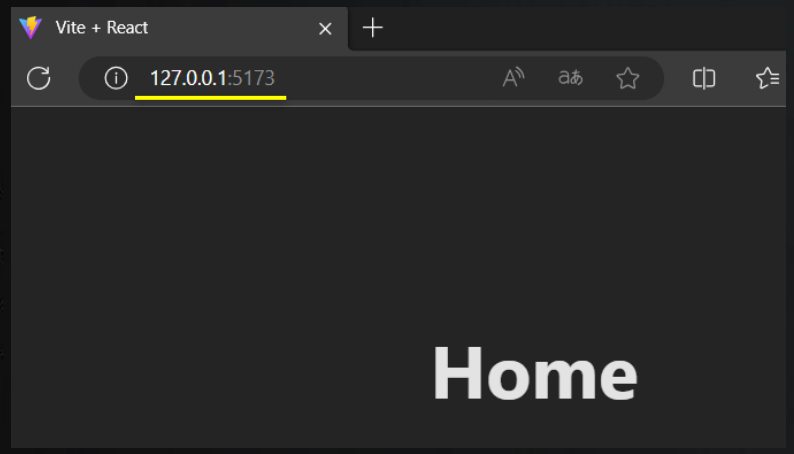
Como configuramos o **App.jsx** como componente principal agora devemos prepara-lo para trabalhar junto com o **RouterProvider**. O App vai receber o **Outlet** que vai disponibilizar a passagem das páginas por dentro dele. Faça conforme o código abaixo:

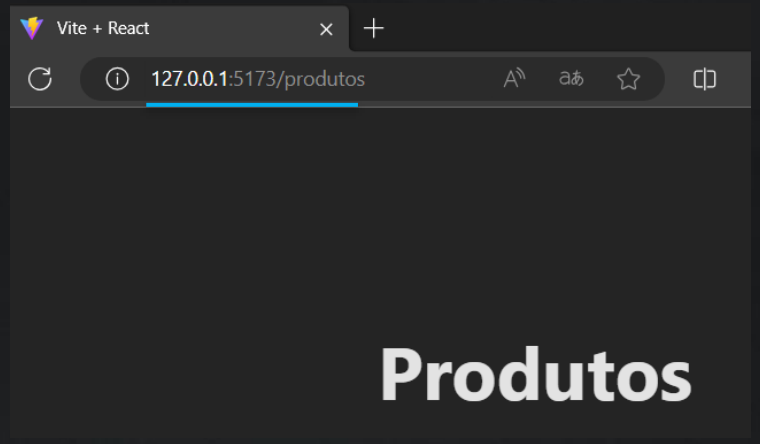
```
import './App.css'
import { Outlet } from 'react-router-dom'

function App() {

  return (
    <>
      <Outlet/>
    </>
  )
}
export default App
```

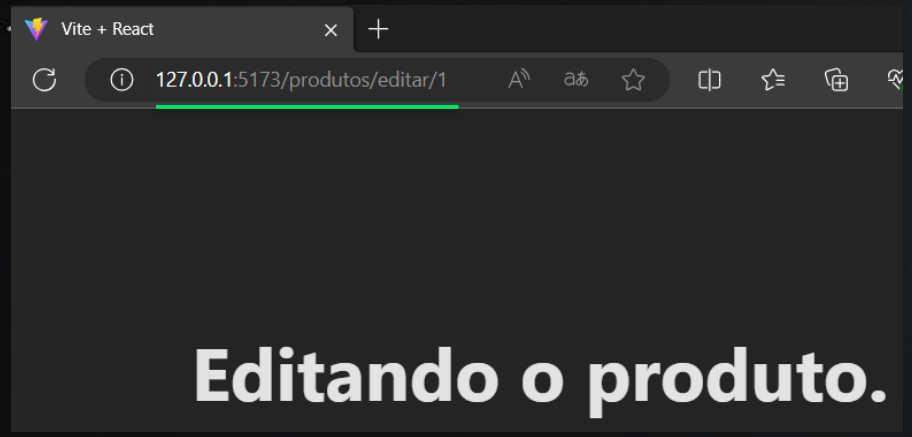
REACT - ROTAS

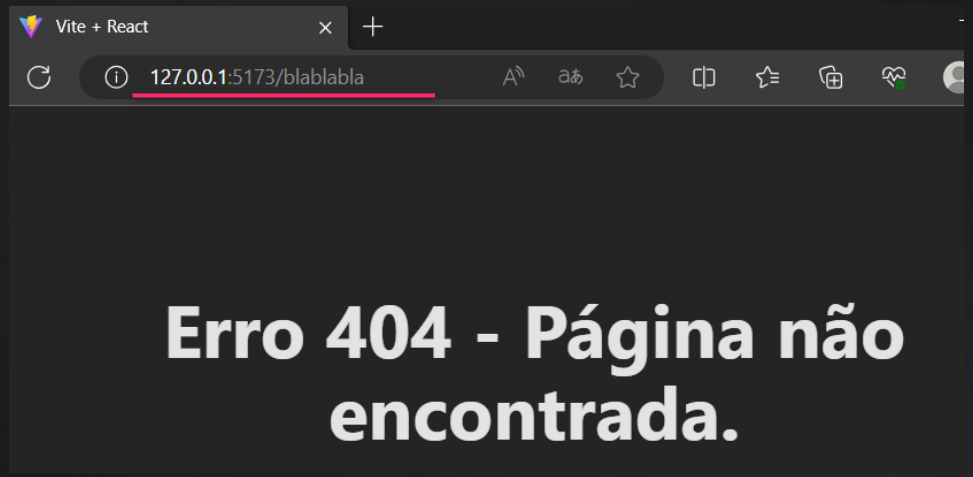
NAVEGANDO USANDO AS URI'S

Com essa primeira parte implementada já podemos testar nossas rotas.









REACT - ROTAS

CRIANDO LINKS PARA NAVEGAR

Não podemos deixar uma aplicação desta forma, os usuários não sabem os endereços, então precisamos preparar Links para ele poderem navegar.
Vamos Criar um menu e como estamos em uma SPA, ele vai ficar disponível em todas as páginas.
Crie uma pasta chamada **components** dentro de **src** e crie o componente Menu.jsx.

Importando o componente Link do router dom.

```jsx
import { Link } from "react-router-dom"

export default function Menu(){

    return(
        <nav className="menu">
            <Link to='/'>Home</Link>
            <span> | </span>
            <Link to='/produtos'>Produtos</Link>
        </nav>
    )
}
```

Componente Link passando a rota.

Obs. Não se esqueça de chamar o componente **Menu** dentro de **App**, coloque o componente logo acima do **Outlet**.

REACT - ROTAS

AJUSTANDO COMPONENTES REAPROVEITÁVEIS

Vamos reforçar a ideia de que podemos ter componentes que ficam fixos, sendo reutilizados em todas as estruturas de página, assim vamos criar em **components** um componente **Rodape.jsx** e estilizar ele e o nosso menu.

```jsx
export default function Rodape(){

    return(
        <footer>
            <p>Rodapé</p>
        </footer>
    )
}
```

No arquivo App.css inclua as seguintes estilizações.

```css
nav{
    display: flex;  width: 100vw;
    justify-content: center; align-items: center;
    height: 8vh; background-color: royalblue;

    & a{color: white; font-size: 1.5em; padding: 15px; }
}

main{ display: flex; width: 100vw;
  flex-direction: column; justify-content: center;
  align-items: center; height: 70vh;
}

footer{
  display: flex; width: 100vw; justify-content: center;
  align-items: center; height: 5vh;
  background-color: darkblue;
}
```

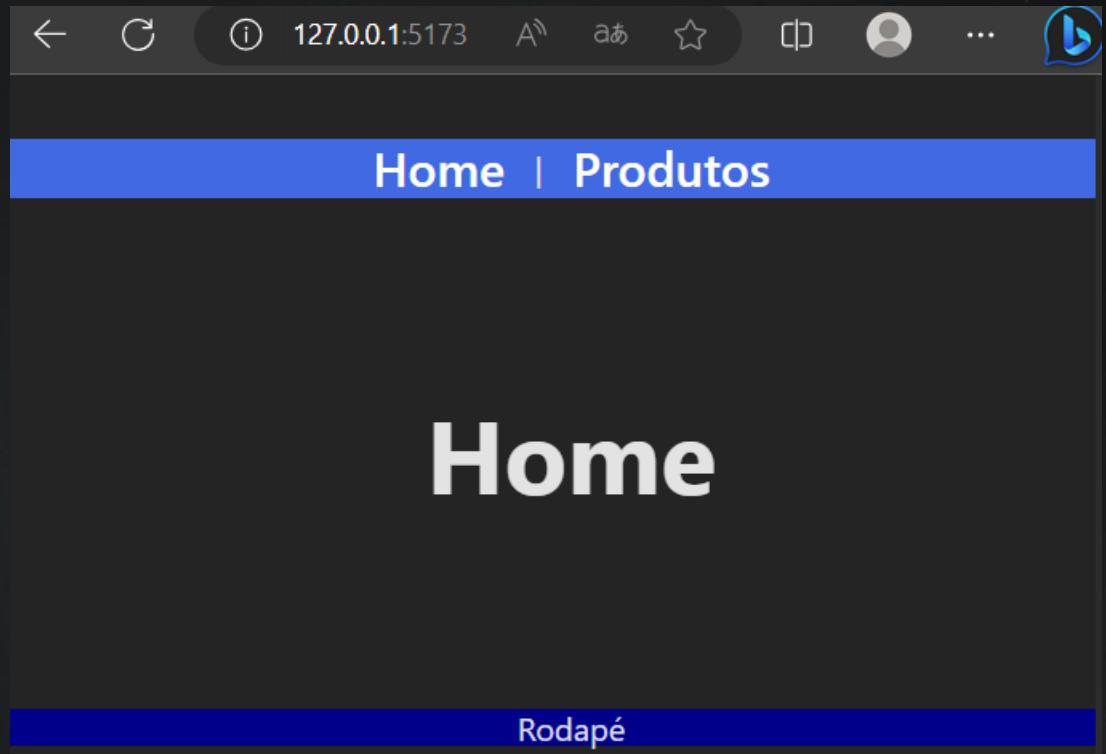
REACT - ROTAS

AJUSTANDO COMPONENTES REAPROVEITÁVEIS

Não esqueça de chamar os componentes **Menu** e **Rodape** no **App**.

```
import './App.css'
import { Outlet } from 'react-router-dom'
import Menu from './components/Menu'
import Rodape from './components/Rodape'

function App() {

  return (
    <>
      <Menu/>
      <Outlet/>
      <Rodape/>
    </>
  )
}
export default App
```

Nossa página deve ficar assim. Lembra que já estamos aproveitando um pouco o CSS do template.



REACT - ROTAS

PASSANDO VALORES PELA URI

Quando precisamos passar valores de uma página para outra, como por exemplo o ID de um produto, podemos usar um Hook chamado UseParams. Vamos criar um arquivo JS com um array de produtos, simulando dados vindos do backend. Crie o arquivo **listaProdutos.js**.

```
export const listaProdutos = [
    {
        id: 1,
        nome: 'Teclado',
        preco: 150
    },
    {
        id: 2,
        nome: 'Mouse',
        preco: 120
    },
    {
        id: 3,
        nome: 'Monitor',
        preco: 950
    },
]
```

REACT - ROTAS

PASSANDO VALORES PELA URI

Vamos carregar os dados de nossa lista na página **Produtos.jsx**. Você reparou que nosso array tinha um export? Usaremos ele para acessar a lista e carregar, usando o método map para criar os links.

```jsx
import { Link } from "react-router-dom"
import { listaProdutos } from "../../components/listaProdutos"

export default function Produtos(){

    return(
        <main>
            <h1>Produtos</h1>
            {listaProdutos.map(prod => (
                <div key={prod.id}>
                    <Link to={`/produtos/editar/${prod.id}`}>
                        Editar o produto: {prod.nome}
                    </Link>
                </div>
            ))}
        </main>
)
}
```

Vamos passar o ID do produto pela URI.

A passagem do id do produto está configurado lá no Main.jsx, no objeto que presenta a página EditarProduto, temos **:id** representando que passaremos dados através dele .

```jsx
{
    path: "/produtos/editar/:id",
    element: <EditarProduto/>
},
```

O próximo passo será configurar o recebimento dele.

REACT - ROTAS

PASSANDO VALORES PELA URI

Na página EditarProduto.jsx vamos receber o ID do produto que devemos selecionar acessando os dados da lista.
Para isso precisaremos usar:

useParams – método do router para pegar os dados passados na URI;

useNavigate – nos permite redirecionar como um link, só que de dentro do código.

Obs. Como os dados que estamos acessando são estáticos, não estamos realmente editando eles, somente acessando.

```jsx
import { useParams, useNavigate } from "react-router-dom"
import { listaProdutos } from "../../components/listaProdutos"

export default function EditarProduto(){

    const lista = listaProdutos
    const navegacao = useNavigate()
    const {id} = useParams()

    const proc = lista.filter(prod => prod.id == id)
    const produto = proc[0]

    const salvar = ()=>{
        alert(`Produto: ${produto.nome} editado com sucesso!`)
        return navegacao('/produtos')
    }

    return(
        <main>
            <h1>Editando o produto </h1>
            <p>Editando os dados do produto: {produto.nome}</p>
            <button onClick={salvar}>Salvar</button>
        </main>
    )
}
```

REACT - ROTAS

REDIRECIONANDO UM ENDEREÇO INUTILIZADO

Um caso que pode acontecer é precisarmos desativar uma página temporariamente ou definitivamente e não termos o tempo hábil de remover todas as chamadas dela na aplicação, para isso podemos usar o Navigate diretamente no **Main.jsx** para trazer o usuário de volta a página principal.

Importando o Navigate do pacote Router-DOM.

```jsx
import { createBrowserRouter, RouterProvider, Navigate } from 'react-router-dom'
import Home from './routes/Home/index.jsx'
import Produtos from './routes/Produtos/index.jsx'
```

```jsx
    path: "/produtos/editar/:id",
    element: <EditarProduto/>
  },
  {
    path: "/antiga",
    element: <Navigate to="/"/>
  }
]
```

Novo objeto em nosso array de rotas. Assim se alguém tentar ir para o endereço antigo, será direcionado para a página home.

EXERCÍCIO

FIAP

Crie uma aplicação para um site de vendas de Smartphones e Tablets.
Ele deve conter as seguintes páginas:

- Home – Na página inicial deve ter duas propagandas de promoções;
- Aparelhos – Nesta página deverá ter uma lista de aparelhos, os dados deverão vir de uma lista criada em um arquivo JS a parte, no mínimo 5 aparelhos;
- VisualizarAparelho – Após o cliente escolher um dos aparelhos, ele deverá ser direcionado para esta página, onde poderá ver a foto e todos os detalhes do aparelho.

- O Cabeçalho e rodapé deverão ficar fixos, sendo aproveitados por todas as páginas.
- Deverá ter uma página para orientar o usuário caso ocorra o Erro 404, e ele deverá ter a opção de voltar para página Home.
- Todas as páginas devem estar estilizadas.

OBRIGADO

FIAP

FIAP