

Análise e Desenvolvimento de Sistemas

Frameworks Web I

Aula 06 - React JS: parte 5



DOM



VIRTUAL DOM

React Router

React Router



A versão mais atual neste momento é a 6.26.2

O React Router é uma poderosa biblioteca de roteamento construída sobre o React que ajuda a adicionar novas telas e fluxos à sua aplicação de forma incrivelmente rápida, mantendo a URL em sincronia com o que está sendo exibido na página.

- Esteja você construindo um site de várias páginas ou uma aplicação de página única, o React Router tem tudo o que você precisa.
- Vamos começar configurando o React Router em um novo projeto React. Primeiro, você precisará instalar o React Router executando o seguinte comando no diretório do seu projeto:

```
npm install react-router-dom@6.26.2
```

React Router

Acesse a documentação da biblioteca



[Clique aqui](#)

React Router

O Routes é a principal diferença na versão 6.16 do React Router em comparação com versões anteriores, como o BrowserRouter.

Após a instalação, você pode importar os componentes necessários do pacote 'react-router-dom'. Em uma aplicação React típica, você terá um único componente Routes envolvendo toda a sua aplicação. Esse componente é responsável por definir as rotas que sua aplicação usará para navegação.

```
import { Routes, Route } from 'react-router-dom';

function App() {
  return (
    <div>
      { /* Outros componentes da sua aplicação aqui */ }
      <Routes>
        { /* Suas rotas vão aqui */ }
      </Routes>
    </div>
  );
}
```

React Router

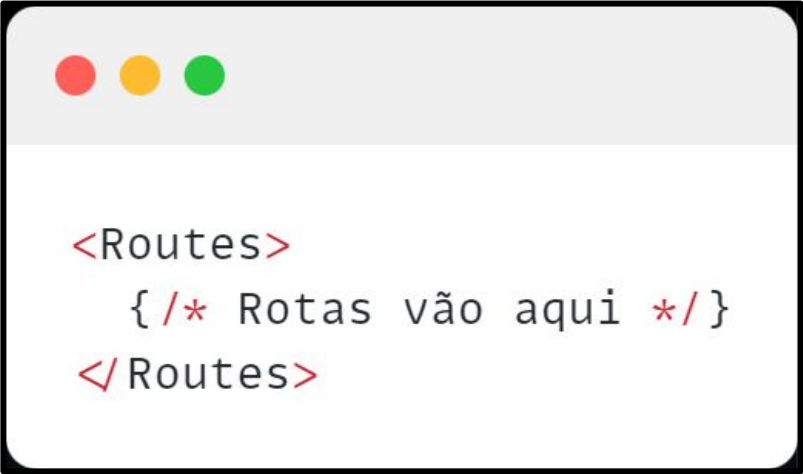
A seguir, você usará os componentes `Route` e `Routes` para definir as diferentes "páginas" da sua aplicação com o React Router versão 6.16.

```
import { Routes, Route } from 'react-router-dom';

function App() {
  return (
    <div>
      { /* Outros componentes da sua aplicação aqui */ }
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/about" element={<About />} />
        <Route path="/contact" element={<Contact />} />
      </Routes>
    </div>
  );
}
```

React Router

Aqui, estamos usando o componente `Routes` para envolver as definições das rotas da nossa aplicação. Todas as rotas serão configuradas dentro deste componente.

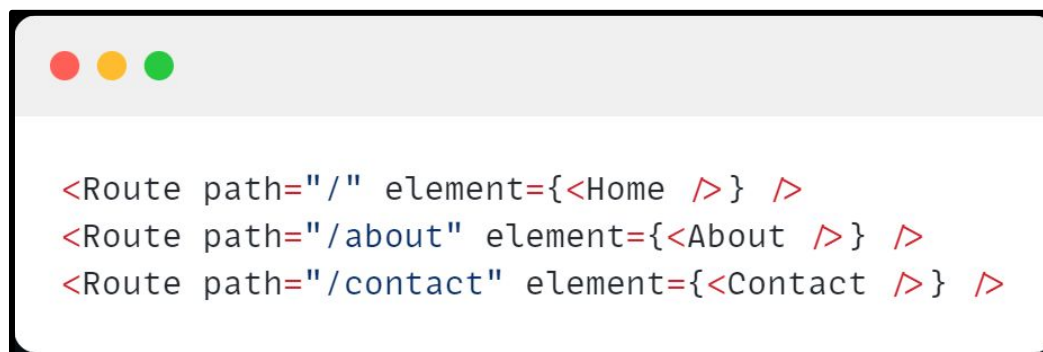


```
<Routes>
  { /* Rotas vão aqui */ }
</Routes>
```


React Router

Aqui estão as definições de três rotas diferentes:

- A primeira rota corresponde ao caminho / e renderiza o componente `<Home />` quando o usuário visita a página inicial.
- A segunda rota corresponde ao caminho /about e renderiza o componente `<About />` quando o usuário visita a página "Sobre".
- A terceira rota corresponde ao caminho /contact e renderiza o componente `<Contact />` quando o usuário visita a página de "Contato".



```
<Route path="/" element={<Home />} />  
<Route path="/about" element={<About />} />  
<Route path="/contact" element={<Contact />} />
```

React Router

Agora, digamos que você queira criar links para essas diferentes páginas a partir de uma barra de navegação.

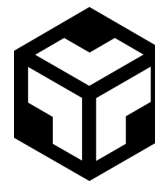
É aqui que usamos o componente Link.

Este código cria um componente de barra de navegação que contém links que permitem que os usuários naveguem para diferentes partes da sua aplicação React. Quando um usuário clica em um link, a aplicação irá direcioná-lo para a página correspondente definida no atributo `to` do componente `Link`.

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

function Navigation() {
  return (
    <nav>
      <ul>
        <li>
          <Link to='/'>Home</Link>
        </li>
      </ul>
      <ul>
        <li>
          <Link to='/about'>Sobre</Link>
        </li>
      </ul>
      <ul>
        <li>
          <Link to='/contact'>Contato</Link>
        </li>
      </ul>
    </nav>
  );
}
```

React Router



[Acesse o projeto](#)

Projeto de exemplo



Route Params

Route Params

Os parâmetros de rota são segmentos dinâmicos da URL que o React Router pode extrair e disponibilizar para seus componentes.

- Eles são essenciais quando você deseja criar rotas dinâmicas, onde a própria rota pode mudar com base na interação do usuário ou em outros fatores, mas ainda corresponde a um tipo específico de visualização.
- Eles também são úteis quando você está lidando com aplicativos orientados por dados, onde um pedaço único de dados (como um ID) pode ditar qual informação exibir.

Route Params



Este código define uma rota usando o componente Route:

- **<Route>:** Este é o componente principal do React Router usado para definir rotas em sua aplicação.
- **path="/post/:id":** Esta é a propriedade path que especifica o caminho da URL que esta rota corresponderá. Neste caso, a rota corresponderá a URLs que tenham o formato "/post/algum-id", onde ":id" é um espaço reservado para um valor dinâmico, como um identificador de postagem.
- **element={<Post />}:** Esta é a propriedade element que define o componente a ser renderizado quando a rota corresponder ao URL especificado. Aqui, ele está configurado para renderizar o componente Post.

Route Params

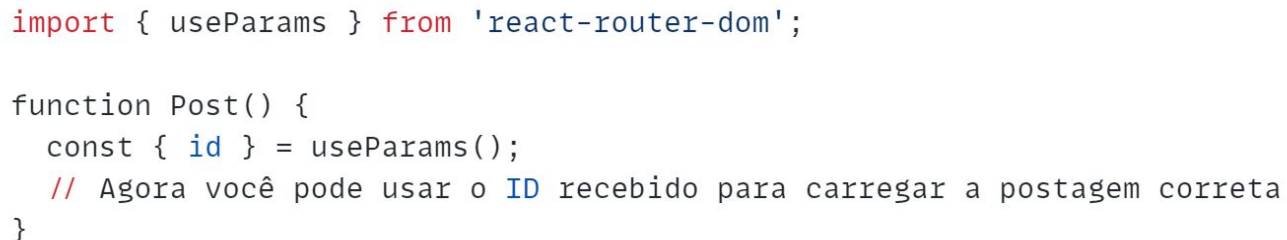


Portanto, no exemplo, quando um URL corresponde ao padrão `/post/algum-id`, a rota direcionará o navegador para renderizar o componente `Post`.

- O valor dinâmico `:id` na URL será extraído e estará disponível como um parâmetro para o componente `Post`, permitindo que você acesse e utilize esse valor em sua página. Isso é útil, por exemplo, quando você deseja exibir detalhes específicos de uma postagem com base no ID fornecido na URL.

Route Params

Mas como você acessa esse parâmetro em seu componente? O React Router fornece um gancho útil chamado `useParams` exatamente para esse propósito.



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

function Post() {
  const { id } = useParams();
  // Agora você pode usar o ID recebido para carregar a postagem correta
}
```


Route Params

Ao chamar `useParams()`, você obtém um objeto que contém todos os parâmetros de rota para a rota atual.

- Você pode, então, usar esses parâmetros para determinar quais dados carregar, qual API chamar, etc.

Route Params

Por exemplo, em uma aplicação de blog, você pode usar o ID para buscar a postagem correspondente em seu servidor.

Neste exemplo, sempre que o ID muda, o gancho `useEffect` dispara uma nova solicitação `fetch`, e o componente é renderizado novamente com os novos dados da postagem.

```
import { useParams } from 'react-router-dom';
import { useEffect, useState } from 'react';

function Post() {
  const { id } = useParams();
  const [post, setPost] = useState(null);

  useEffect(() => {
    fetch(`/api/posts/${id}`)
      .then(response => response.json())
      .then(data => setPost(data));
  }, [id]);

  if (!post) {
    return 'Carregando...';
  }

  return (
    <div>
      <h1>{post.title}</h1>
      <p>{post.body}</p>
    </div>
  );
}
```

Route Params

Parâmetros de rota também podem ser combinados e aninhados para formar estruturas complexas de roteamento. Por exemplo, se você estivesse construindo um navegador de arquivos, poderia usar rotas aninhadas para representar o caminho do arquivo:



```
<Route path="/folder/:folderId/file/:fileId" element={<File />} />
```

- Neste caso, tanto `folderId` quanto `fileId` são parâmetros de rota, e você pode acessá-los em seu componente `File` usando o hook `useParams`.

Rotas aninhadas

Rotas aninhadas

Rotas aninhadas, como o nome sugere, são rotas que estão aninhadas dentro de outras rotas.

- Elas são usadas para encapsular parte da rota de sua aplicação em uma parte menor e autocontida de sua aplicação.
- Isso é especialmente útil para aplicativos maiores nos quais o roteamento pode se tornar bastante complexo.
- Com rotas aninhadas, você pode modelar a estrutura de roteamento para corresponder à estrutura de sua interface de usuário, tornando-a mais intuitiva para entender e manter.

Rotas aninhadas

No React Router, o roteamento aninhado é alcançado definindo um componente `Route` dentro de outro componente `Route`.

- Vamos pegar um exemplo de uma página de perfil de usuário com duas abas: 'Visão Geral' e 'Configurações'. Queremos navegar por essas abas enquanto permanecemos na rota 'Usuário'. Primeiro, configuraríamos nossas rotas de nível superior em nosso componente principal, o componente `App`:

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

function App() {
  return (
    <Router>
      <Routes>
        <Route path="/users/:userId" element={<User />} />
        { /* outras rotas */ }
      </Routes>
    </Router>
  );
}
```

Rotas aninhadas

Em seguida, dentro do nosso componente User, podemos definir rotas aninhadas para as abas 'Visão Geral' e 'Configurações':

```
import { Link, Route, Routes } from 'react-router-dom';

function User() {
  return (
    <div>
      { /* Conteúdo do perfil do usuário */ }
      <ul>
        <li>
          <Link to="overview">Visão Geral</Link>
        </li>
        <li>
          <Link to="settings">Configurações</Link>
        </li>
      </ul>
      { /* Rotas aninhadas */ }
      <Routes>
        <Route path="overview" element={ <Overview /> } />
        <Route path="settings" element={ <Settings /> } />
      </Routes>
    </div>
  );
}
```

Rotas aninhadas

O código apresentado define um componente React chamado User, que representa a página de perfil de um usuário.

- Este componente inclui uma lista de links, "Visão Geral" e "Configurações", que permitem ao usuário navegar entre as abas da página.
- As rotas para as abas são aninhadas usando o componente Routes, com rotas individuais definidas usando o componente Route.
- Quando um link é clicado, a URL muda para corresponder à aba selecionada, e o React Router renderiza o conteúdo apropriado para essa aba.
- Isso cria uma experiência de navegação aninhada, onde as abas do perfil do usuário são acessadas através da URL e os componentes são renderizados dinamicamente com base na escolha do usuário.

Rotas aninhadas

Como resultado, quando o usuário navega para `/users/1/overview`, o componente `User` é renderizado pela rota de nível superior e, em seguida, o componente `Overview` é renderizado pela rota aninhada.

Rotas aninhadas



[Acesse o projeto](#)

Projeto de exemplo



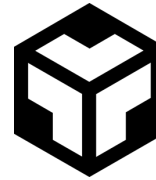
Prática

Prática

Este projeto se trata de uma aplicação web que simula uma plataforma de lojas e produtos.

- Os usuários podem escolher entre várias lojas listadas na página inicial, cada uma com sua própria lista de produtos exclusivos.
- Ao clicar em uma loja, eles são direcionados para a página da loja, onde podem ver os produtos disponíveis com detalhes, como nome e descrição.
- Além disso, há uma opção para voltar à página inicial a partir de qualquer página da loja.
- A aplicação utiliza React e React Router para criar diferentes rotas para a página inicial, as páginas de loja e os detalhes do produto, tornando-a uma experiência de navegação interativa e amigável para o usuário.

Prática



[Acesse o projeto](#)

Minha Loja de Produtos

Escolha uma loja:

Loja A

Uma loja incrível com produtos incríveis.

[Ver produtos](#)

Loja B

Tudo o que você precisa em um só lugar.

[Ver produtos](#)

Prática

Passo 1: Configuração do Ambiente

- Crie um diretório (pasta) em seu computador para o projeto, se ainda não tiver feito isso.

Passo 2: Configuração do Projeto

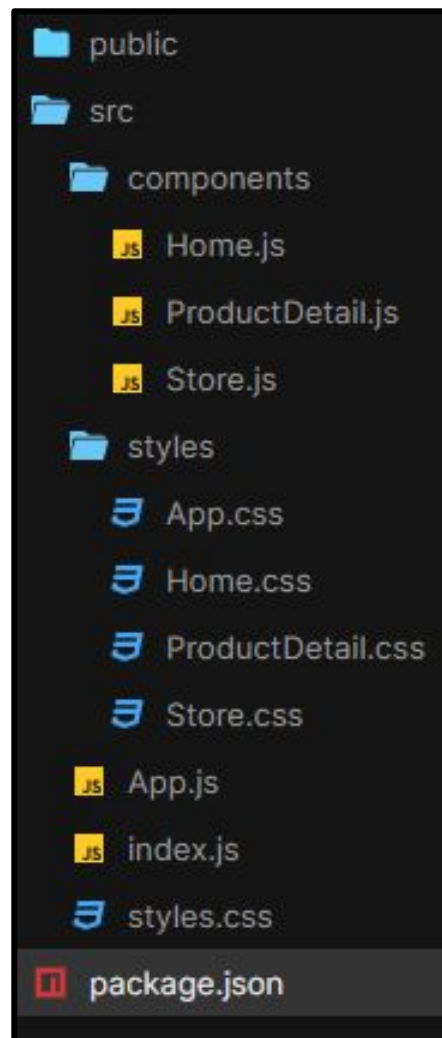
- Abra o terminal e navegue até o diretório do projeto.
- Inicie um novo projeto React utilizando o seguinte comando:

```
npm create vite@latest lojas-e-produtos -- --template react
cd lojas-e-produtos
npm install
npm run dev
```

Prática

Passo 3: Estruturação do Projeto

- Apague os arquivos de código padrão gerados pelo Create React App na pasta "src".
- O projeto terá a seguinte estrutura de pastas e arquivos

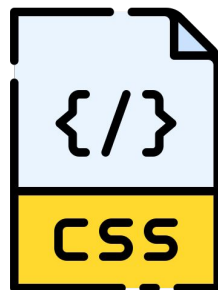


Prática

Passo 4: arquivos .css

- Baixe os arquivos .css e os adicione na pasta style do projeto

[Clique aqui para baixar](#)



Prática

Passo 5: arquivo App.js

Este código define um componente React chamado App. Ele configura o roteamento da aplicação usando o pacote react-router-dom. A página tem um cabeçalho com o título "Minha Loja de Produtos" e três rotas:

- A rota padrão "/" leva ao componente Home.
- A rota "/store/:storeId" leva ao componente Store, permitindo passar um parâmetro chamado storeId.
- A rota "/product/:productId" leva ao componente ProductDetail, permitindo passar um parâmetro chamado productId. Este código forma a estrutura básica de uma aplicação React com roteamento.

Prática



[Clique aqui para ampliar a imagem](#)

```
import React from "react";
import { BrowserRouter as Router, Route, Routes } from "react-router-dom";
import "./styles/App.css";

import Home from "../components/Home";
import Store from "../components/Store";
import ProductDetail from "../components/ProductDetail";

function App() {
  return (
    <Router>
      <div className="App">
        <header className="header">
          <h1>Minha Loja de Produtos</h1>
        </header>
        <div className="container">
          <Routes>
            <Route path="/" element={<Home />} />
            <Route path="/store/:storeId" element={<Store />} />
            <Route path="/product/:productId" element={<ProductDetail />} />
          </Routes>
        </div>
      </div>
    </Router>
  );
}

export default App;
```

Prática

Passo 6: arquivo Home.js

Este código é um arquivo JavaScript que define o componente React chamado Home, que representa a página inicial de uma aplicação web.

- Define uma função chamada Home, que é um componente React.
- Declara uma lista fictícia de lojas com dados como id, name, e description.
- Renderiza um título "Escolha uma loja:" e uma lista de lojas com base nos dados fictícios.
- Usa um loop map para criar elementos de lista () para cada loja.
- Utiliza o componente Link para criar links para a página de detalhes de cada loja, com base no id da loja.

Prática



[Clique aqui para ampliar a imagem](#)

```
import React from "react";
import { Link } from "react-router-dom"; // Importa o componente Link do React Router
import "../styles/Home.css"; // Importa o estilo CSS para este componente

function Home() {
  // Dados fictícios da lista de lojas
  const stores = [
    {
      id: 1,
      name: "Loja A",
      description: "Uma loja incrível com produtos incríveis."
    },
    {
      id: 2,
      name: "Loja B",
      description: "Tudo o que você precisa em um só lugar."
    },
    {
      id: 3,
      name: "Loja C",
      description: "Qualidade e variedade em nossos produtos."
    }
  ];

  return (
    <div>
      <h2>Escolha uma loja:</h2>
      <ul className="store-list">
        {stores.map((store) => (
          <li className="store-item" key={store.id}>
            <h3 className="store-name">{store.name}</h3>
            <p className="store-description">{store.description}</p>
            <Link to={`/store/${store.id}`} className="store-link">
              Ver produtos
            </Link>
          </li>
        ))}
      </ul>
    </div>
  );
}

export default Home;
```

Prática

Passo 7: arquivo ProductDetail.js

Este código é um arquivo JavaScript que define o componente React chamado ProductDetail, responsável por exibir detalhes de um produto em uma aplicação web.

- Define uma função chamada ProductDetail, que é um componente React. Usa o hook useParams para obter o parâmetro productId da URL, que representa o ID do produto.
- Usa o hook useNavigate para obter uma função de navegação para redirecionar o usuário para outras páginas.
- Cria um objeto product com base no productId e atribui valores fictícios a id, name e description.
- Renderiza o nome e a descrição do produto. Implementa um botão "Voltar para a lista de produtos" que, ao ser clicado, utiliza a função navigate para redirecionar o usuário de volta para a página da loja correspondente ao produto.

Prática



[Clique aqui para ampliar a imagem](#)

```
import React from "react";
import { useParams, useNavigate } from "react-router-dom"; // Importe o hook useNavigate
import "../styles/ProductDetail.css";

function ProductDetail() {
  const { productId } = useParams(); // Obtém o parâmetro productId da URL
  const navigate = useNavigate(); // Use o hook useNavigate para navegação entre rotas

  const product = {
    id: productId,
    name: "Produto " + productId,
    description: "Descrição do Produto " + productId + "."
  };

  const handleBack = () => {
    // Use navigate para voltar para a página da loja com o ID correto
    navigate(`/store/${product.id}`);
  };

  return (
    <div className="product-detail-container">
      <h2 className="product-name">{product.name}</h2>
      <p className="product-description">{product.description}</p>
      { /* Adicione o ID da loja à rota de volta para a loja */ }
      <button onClick={handleBack} className="back-button">
        Voltar para a lista de produtos
      </button>
    </div>
  );
}

export default ProductDetail;
```

Prática

Passo 8: arquivo Store.js

Este código é um arquivo JavaScript que define o componente React chamado Store. Ele representa a página que exibe os produtos de uma loja específica em uma aplicação web.

- Define uma função chamada Store, que é um componente React. Usa o hook `useParams` para obter o parâmetro `storeId` da URL, que representa o ID da loja atual. Usa o hook `useNavigate` para obter uma função de navegação para redirecionar o usuário para outras páginas.
- Declara uma lista fictícia de produtos com dados como `id`, `name` e `description`. Filtra os produtos para exibir apenas os produtos da loja específica com base no `storeId`.
- Renderiza o nome e a descrição de cada produto, juntamente com um link "Ver detalhes" que leva à página de detalhes desse produto. Implementa um botão "Voltar para a página inicial" que, ao ser clicado, utiliza a função `navigate` para redirecionar o usuário de volta à página inicial.

Prática



[Clique aqui para ampliar a imagem](#)

```
import React from "react";
import { Link, useParams, useNavigate } from "react-router-dom"; // Importe o hook useNavigate
import "../styles/Store.css";

function Store() {
  const { storeId } = useParams(); // Obtém o parâmetro storeId da URL
  const navigate = useNavigate(); // Use o hook useNavigate para navegação entre rotas

  // Dados fictícios da lista de produtos da loja
  const products = [
    {
      id: 1,
      name: "Produto 1",
      description: "Descrição do Produto 1."
    },
    {
      id: 2,
      name: "Produto 2",
      description: "Descrição do Produto 2."
    },
    {
      id: 3,
      name: "Produto 3",
      description: "Descrição do Produto 3."
    }
  ];

  // Filtra os produtos para exibir apenas os da loja específica com base no storeId
  const filteredProducts = products.filter(
    (product) => product.id === parseInt(storeId, 10) // Converte o storeId para número
  );

  const handleBackToHome = () => {
    // Use navigate para voltar para a página inicial (URL '/')
    navigate("/");
  };

  return (
    <div className="store-container">
      <h2>Produtos da Loja {storeId}</h2>
      <ul className="product-list">
        {filteredProducts.map((product) => (
          <li className="product-item" key={product.id}>
            <h3 className="product-name">{product.name}</h3>
            <p className="product-description">{product.description}</p>
            {/* Cria um link para o detalhe do produto com base no ID */}
            <Link to={`/product/${product.id}`} className="store-link">
              Ver detalhes
            </Link>
          </li>
        ))}
      </ul>
      {/* Adicione um botão para voltar para a página inicial */}
      <button onClick={handleBackToHome} className="back-button">
        Voltar para a página inicial
      </button>
    </div>
  );
}

export default Store;
```


Prática

Passo 9: Dependências

Certifique-se de que as dependências necessárias do React Router estejam instaladas no projeto. Você pode instalá-las com o seguinte comando:

```
npm install react-router-dom@6.26.2
```

Prática

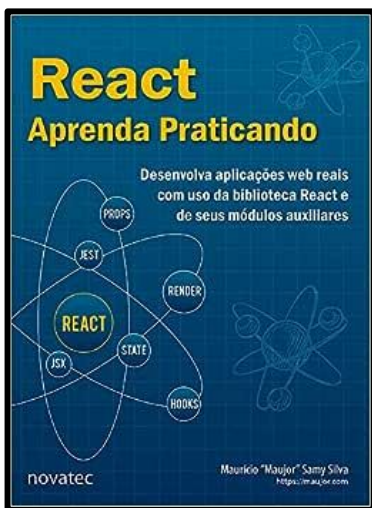
Passo 10: Execute e teste o projeto

Abra um navegador da web e acesse `http://localhost:5173` para visualizar a aplicação.

- Você verá a página inicial, onde poderá escolher entre as lojas listadas.
- Clique em uma loja para ver os produtos disponíveis e detalhes sobre eles.
- Use os botões "Voltar para a página inicial" ou "Voltar para a lista de produtos" para navegar entre as páginas.

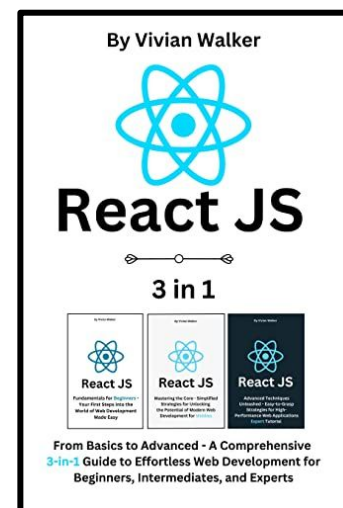
Referências

Referências



SILVA, Maurício Samy. **React - Aprenda praticando**. Novatec. 2021.

Walker, Vivian. **React JS: From Basics to Advanced - A Comprehensive 3-in-1 Guide to Effortless Web Development for Beginners, Intermediates, and Experts**.



Análise e Desenvolvimento de Sistemas

Frameworks Web I

Aula 06 - React JS: parte 5
