Do Zero ao Deploy com React e Next.js

WebTech PUC Minas

Mariana Almeida
Nilson Deon
Alice Salim

Organização

Como será trabalhado o curso?

02 **React.js: principais conceitos**

03 **Next.js: principais conceitos**

O que é o React.js?

O React.js é uma **biblioteca para JavaScript**, ou seja, ela simplesmente transforma o existente na linguagem padrão, possibilitando **melhorias** para o **desenvolvedor** e na **experiência de usuário.**

MELHORIAS

Componentização

Divide a interface em componentes reutilizáveis e independentes, facilitando a manutenção e a escalabilidade do código.

Virtual DOM:

Utiliza uma representação em memória do DOM para otimizar atualizações, melhorando o desempenho em aplicações interativas.

Roteamento sem Recarga:

Permite a navegação entre páginas sem recarregar a página, proporcionando uma experiência de usuário mais fluida.

Labs

Todo projeto desenvolvido pelos membros da WebTech gera um ou mais labs, que são repositórios no GitHub que contam com todo o detalhamento técnico das tecnologias utilizadas e dos conhecimentos desenvolvidos.

lab-react

Lab de introdução ao React. Ele visa ajudar desenvolvedores iniciantes a entender os conceitos básicos do React e a criar uma aplicação básica.

</> JavaScript ★ 1

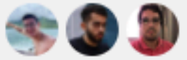
Saiba mais

lab-spring-boot-mvc

</> HTML ★ 2

Saiba mais

lab-git-gitflow

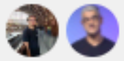


Conceitos fundamentais da ferramenta de controle de versão Git e do modelo de fluxo de trabalho GitFlow.

★ 3

Saiba mais

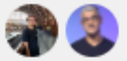
lab-devops-github-actions



Passo a passo para a elaboração de um workflow de deploy no GitHub.

★ 1

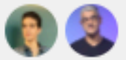
lab-azure-web-server



Passo a passo para montagem de Web Server no ambiente do Microsoft Azure.

★ 2

lab-puppeteer



Exemplos de uso da biblioteca Puppeteer para raspagem de dados (Web scrape) e automações na Web.

</> JavaScript ★ 3

00

Setup Padrão

Iniciando um novo projeto em React!

```
node -v
npm -v
```

```
npm install -g create-react-app
```

```
npm install -g npm@10.8.3
```

```
npx create-react-app web_tech_page
```

01

Nosso projeto

Por questões de produtividade já vamos clonar um projeto iniciado.

```
git clone https://github.com/WebTech-PUC-Minas/workshop-react.git
```

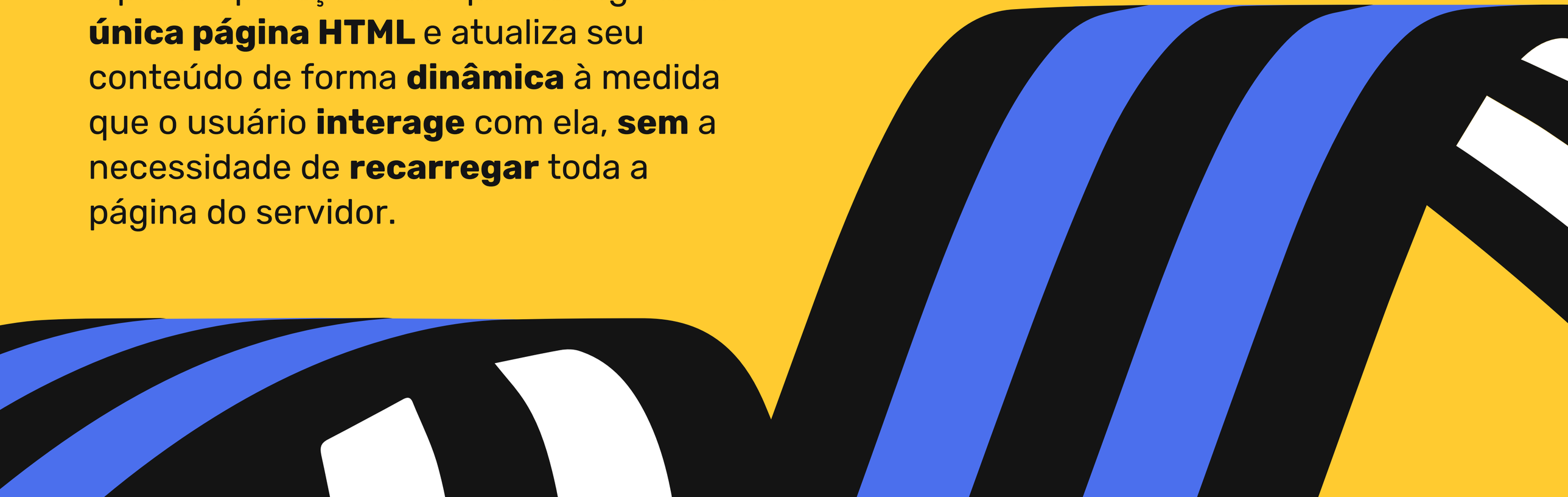
```
cd workshop-react
```

```
npm start
```

SPA

Single Page Application

Tipo de aplicação web que carrega uma **única página HTML** e atualiza seu conteúdo de forma **dinâmica** à medida que o usuário **interage** com ela, **sem** a necessidade de **recarregar** toda a página do servidor.

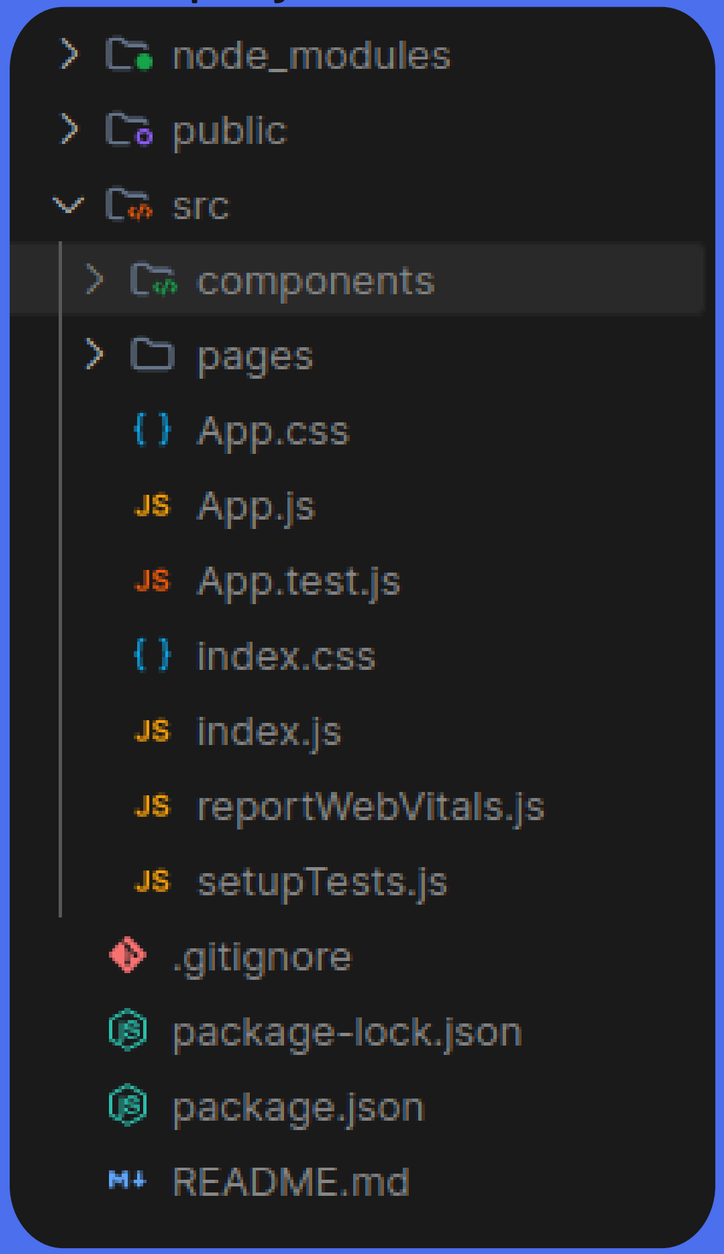


Organização das pastas no React:

Padrão:

```
web_tech_page/
├── public/
├── src/
│   ├── App.js
│   ├── index.js
├── package.json
├── README.md
└── .gitignore
```

Nosso projeto no Git:



02

NAVBAR:

Nosso primeiro componente.

HTML

```html
<!DOCTYPE html>
<html lang="pt-BR">

<body>
    <header>
        <h1>WebTech PUC Minas</h1>
        <nav>
            <ul>
                <li>Sobre</li>
                <li>Labs</li>
                <li>Eventos</li>
                <li>Equipe</li>
                <li>Contato</li>
            </ul>
        </nav>
    </header>
</body>
</html>
```

JSX

```jsx
function Navbar() {
  return (
    <header className={style.navbar}>
      <h1><a>WebTech PUC Minas</a></h1>
      <nav>
        <ul>
          <li><a>React.js</a></li>
          <li><a>Next.js</a></li>
        </ul>
      </nav>
    </header>
  );
}
```

Porque devemos componentizar?

Reutilização

Manutenção **Facilitada**

Desempenho Aprimorado

03

O QUE SÃO OS COMPONENTES

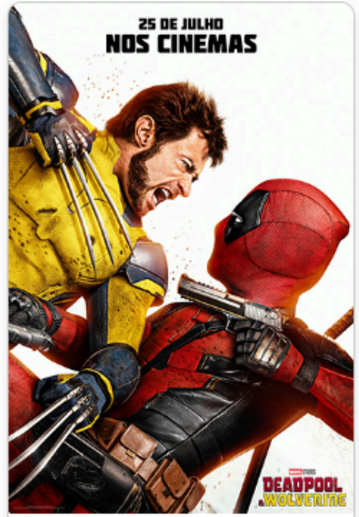
Componentes são como **funções** que **retornam** uma **interface** (User interface - UI)

Eduardo e Mônica

23 de set de 2024

Guardiões da Galáxia

24 de set de 2024



Deadpool e Wolverine

25 de set de 2024

Batman

26 de set de 2024

CARD

```jsx
function Card() {
  return (
    <div className={style.card}>
      <figure>
        <img src="/img/movie_01.jpeg" />
      </figure>
      <article>
        <h1>Eduardo e Mônica</h1>
        <div>
          <time>24 de set. 2024</time>
          <img className={style.icon} src="icons/heart-regular.svg" />
        </div>
      </article>
    </div>
  );
```

04 Rotas

- Usamos a biblioteca ***react-router-dom*** para gerenciamento de rotas
- Permite **navegação** entre **diferentes** páginas

Sobre react-router-dom:

BrowserRouter

Componente principal que envolve toda a aplicação

Controla a **sincronização** do URL com a interface do usuário.

Route

Define uma rota

Componente ser **renderizado** quando o **URL corresponder** ao **caminho** especificado

Switch

Renderiza somente a primeira rota filha

A **primeira rota** filha corresponde ao caminho atual

Evita que múltiplas rotas sejam renderizadas ao mesmo tempo.

Link

Criar links de navegação

Cria links de navegação que modificam o URL sem recarregar a página

Substitui o uso de <a> para navegação interna.

APP

```
import React from "./pages/React"
import Next from "./pages/Next"

import Navbar from './components/Navbar';

function App() {
  return (
    <Router>
      <Navbar />
        <Routes>
          <Route exact path="/" Component={React} />
          <Route exact path="/next" Component={Next} />
        </Routes>
    </Router>
  );
}
```

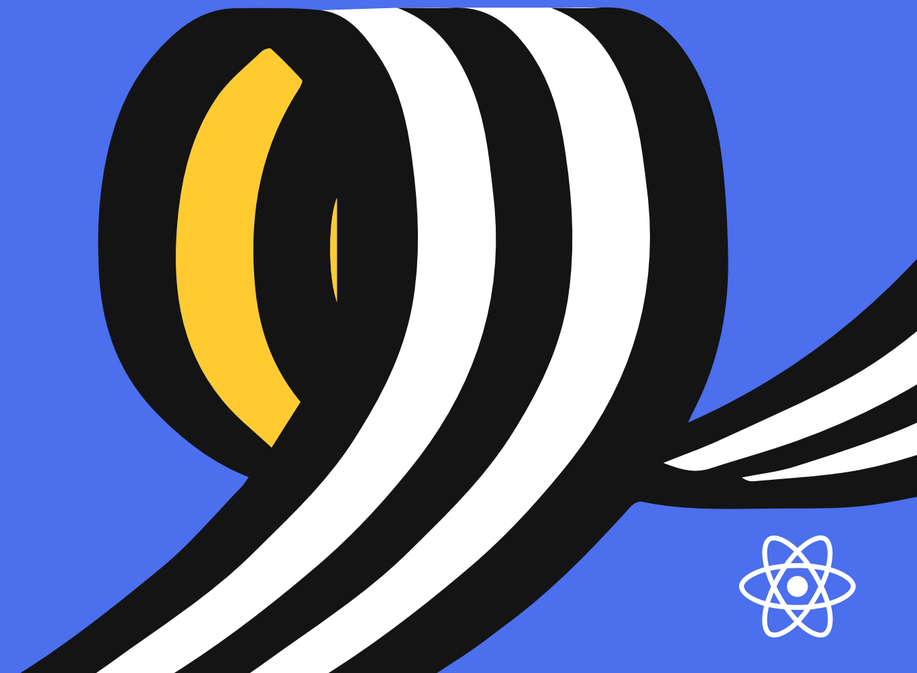
Passo a passo: criando rotas

01 Criar páginas **React** e **Next**

02 **Adicionar texto** dentro das páginas

03 Criar **“rotas”** no APP.js

04 Trocar **<a>** por **<Link/>**



Recapitulando

01 O que é React?

- Biblioteca Java Script
- Componentização

02 Componentes

- Simplificação de partes da tela
- Reutilizáveis

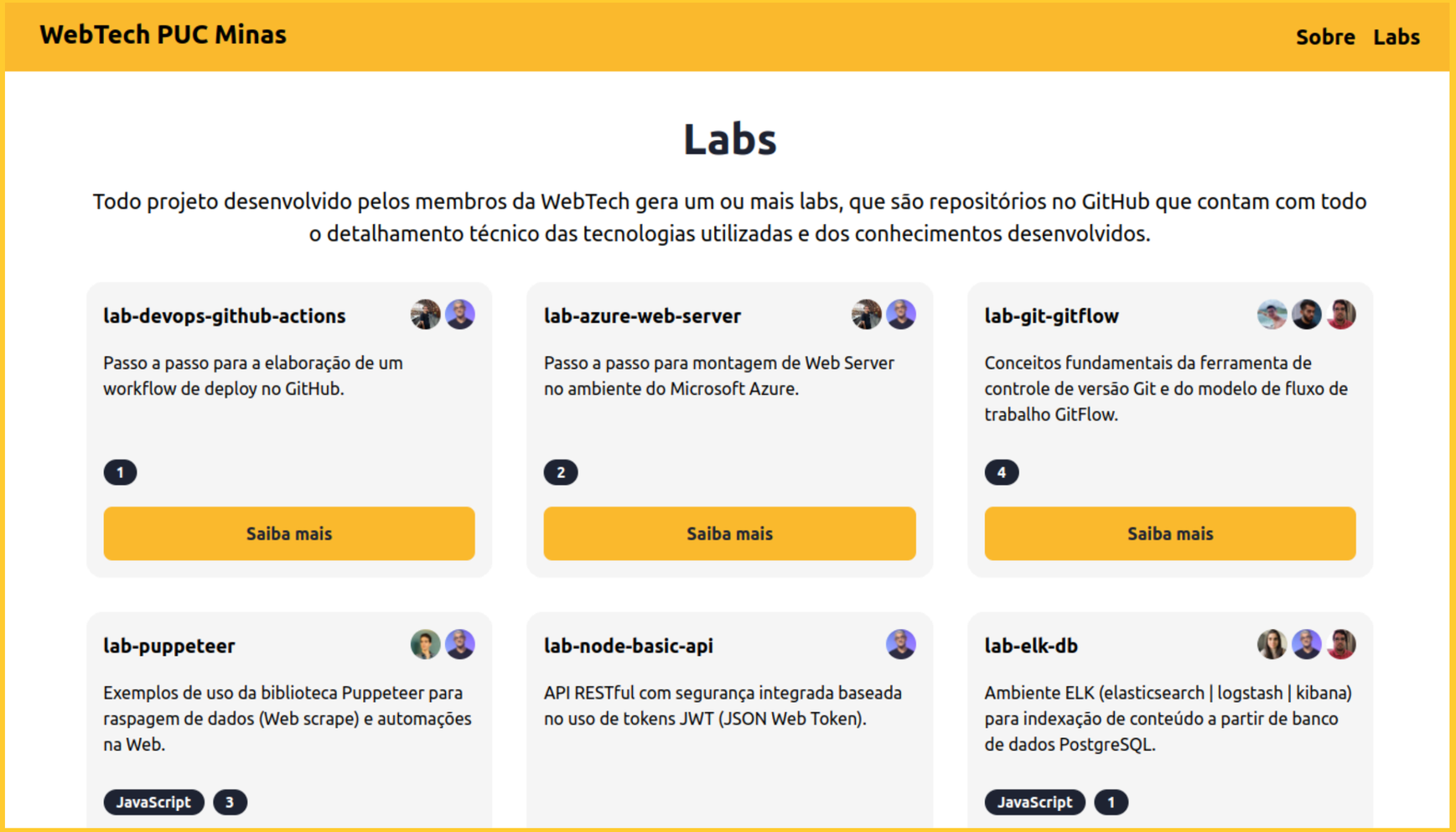
03 Rotas

- Facilita o controle de URL's

05

O QUE SÃO AS PROPS

Props são argumentos passados para componentes React que permitem personalizar e reutilizar componentes



CARD

```jsx
function Card({banner, title, date}) {
  return (
    <div className={style.card}>
      <figure>
        <img src={banner} />
      </figure>
      <article>
        <h1>{title}</h1>
        <div>
          <time>{date}</time>
          <img className={style.icon} src="icons/heart-regular.svg" />
        </div>
      </article>
    </div>
  );
}
```

06

Estados

Estados são objetos que **armazenam dados dinâmicos** e podem mudar ao longo do **tempo**, permitindo que os componentes React sejam **interativos** e **reativos** a mudanças

— **Gerenciamento de Estado:** Use hooks como useState e useReducer para gerenciar o estado local dos componentes.

CARD

```jsx
function Card({ banner, title, date }) {
  const [show, setShow] = useState(false);
  const handleClick = () => {
    setShow(!show);
  };

  return (
    <div className={style.card}>
      <figure>
        <img src={banner} />
      </figure>
      <article>
        <h1>{title}</h1>
        <div>
          <time>{date}</time>
          {show ? (
            <img className={style.icon} src="icons/heart-solid.svg" onClick={handleClick} />
          ) : (
            <img className={style.icon} src="icons/heart-regular.svg" onClick={handleClick} />
          )}
        </div>
      </article>
    </div>
  );
}
```

TIPOS DE HOOKS

USE STATE:

Adiciona e gerencia estados em componentes funcionais

USE EFFECT:

Lida com efeitos colaterais, como API’s, timers ou atualizações manuais

USE CONTEXT:

Compartilhamento de dados sem usar props

07

Boas Práticas

Organização de Componentes

- Componentes Pequenos e Reutilizáveis
- Pastas por Funcionalidade

Estilização

- CSS Modules

Estado e Props

- Gerenciamento de Estado
- Lift State Up
- Prop-Types

Hooks

- useEffect
- Custom Hooks

07

Novas tendências

01 **React Server Components**

02 **Web Vitals e Performance**

03 **TypeScript e Tipagem Estática**

04 **Next.js e Frameworks de Renderização**

OBRIGADO(A)!

Prontos para o workshop de Next?