Do Zero ao Deploy com React e Next.js

WebTech PUC Minas

Mariana Almeida
Nilson Deon
Alice Salim

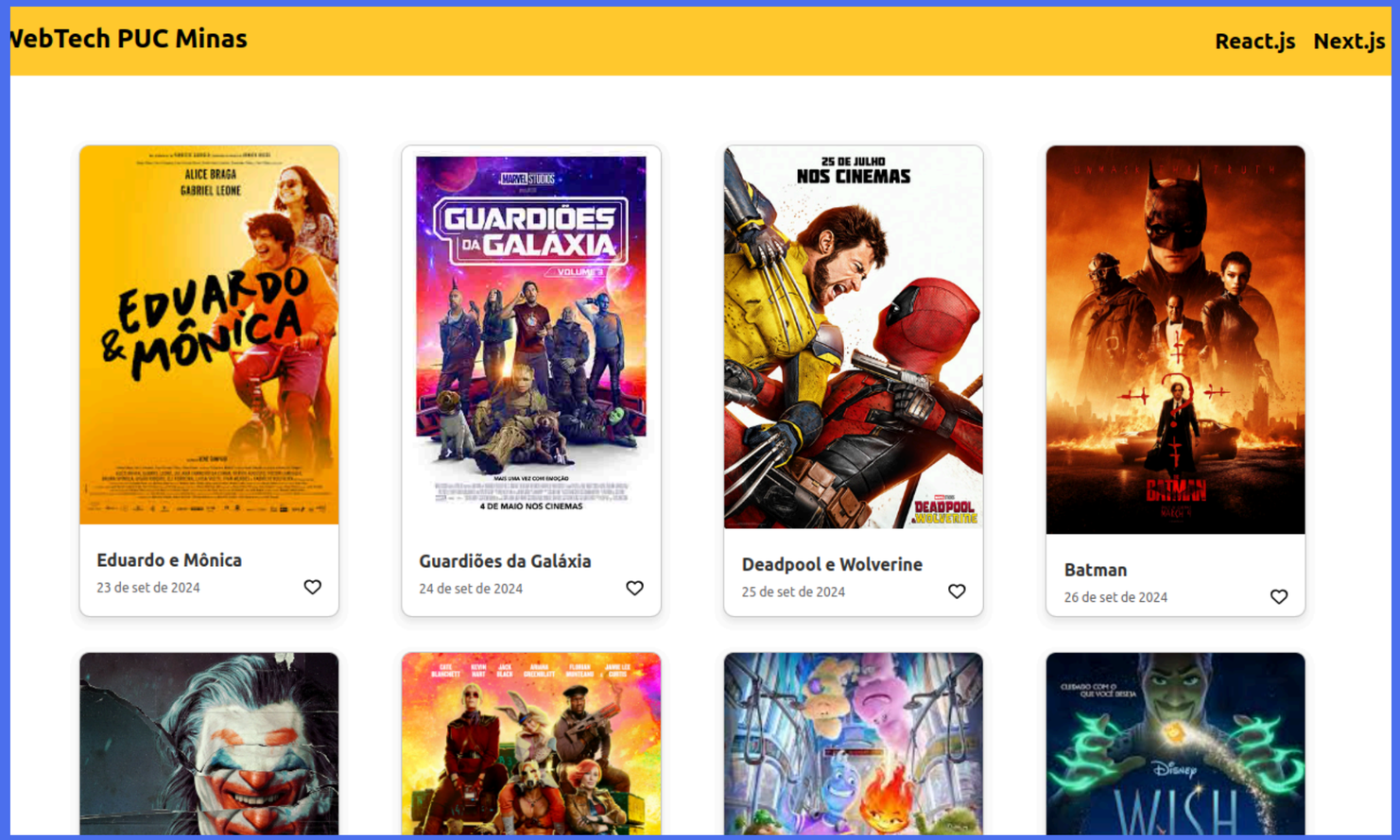
Organização

Como será trabalhado o curso?

02 **React.js: principais conceitos**

03 **Next.js: principais conceitos**

Relembrando React...



01 O que é React?

02 Componentes

03 Rotas

04 Props

05 Estados

Linha do tempo da carreira

1990

Fundamentos da web

HTML (91), CSS(96) e JavaScript(95)

Frameworks e bibliotecas como AJAX e jQuery

2013

React.js

Criado pelo Facebook como forma de facilitar a implementação da rede social.

Introdução de componentes e UI reativa.

2016

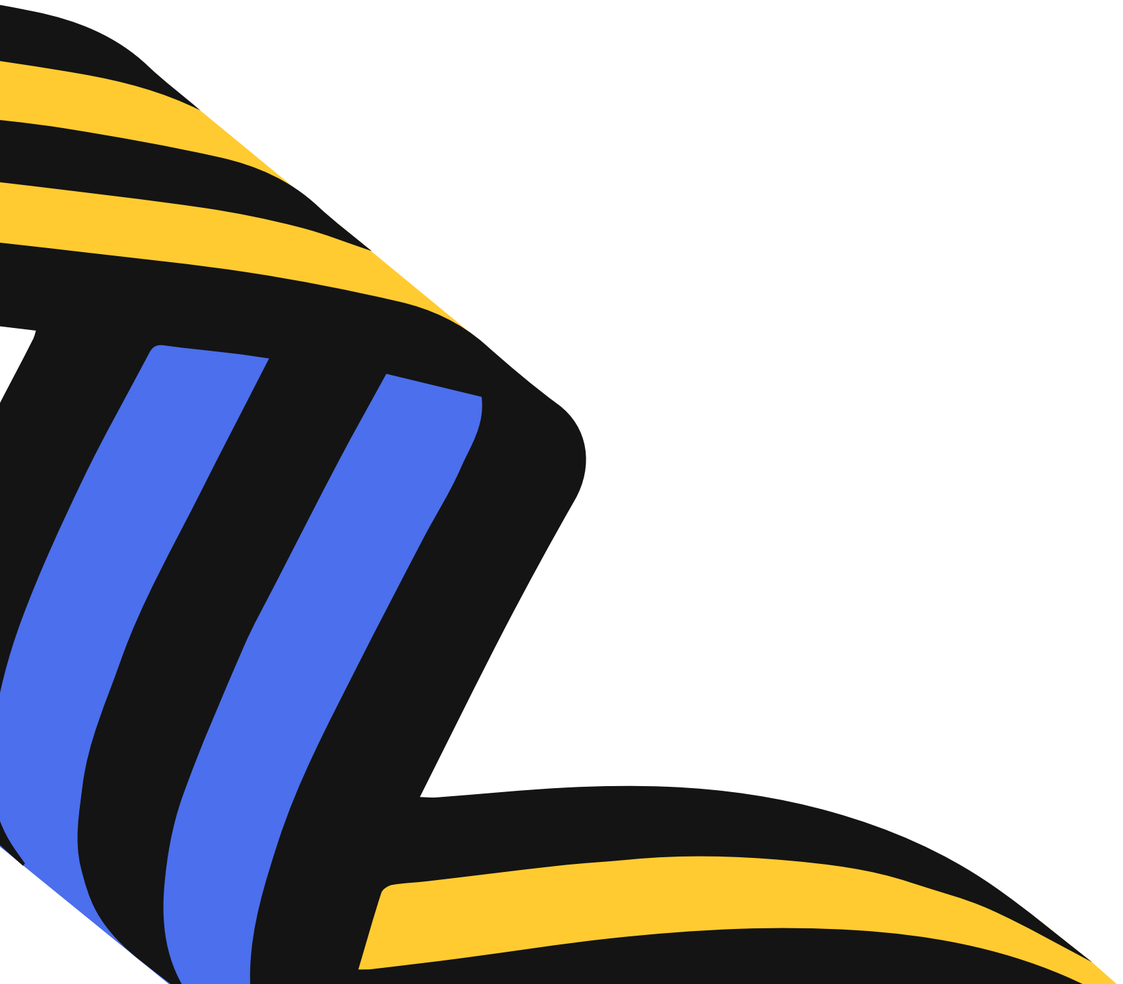
Next.js

Melhoria da experiência de desenvolvimento e otimização, ajudando questões como SEO e performace.

O que é o Next.js?

Next.js é um **framework React** que possibilita renderização no servidor, roteamento simples, **otimização automática** de páginas e melhorias significativas em **performance** e **SEO**

MELHORIAS



— Renderização (Client/Server:)

Next.js fornece algumas formas de construção do site de forma a otimizar carregamento, SEO e interatividade.

— Roteamento:

Next.js oferece um sistema de roteamento baseado em arquivos, onde cada arquivo pages.js dentro de uma no arquivo app é automaticamente transformado em uma rota.

— Otimizações:

Next.js fornece alguns componente, que otimizam automaticamente o site, resultando em uma performance superior.

00

Setup Padrão

Iniciando um novo projeto em React!

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

```
npx create-next-app@latest 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-next.git
```

```
cd workshop-next
```

```
npm install
```

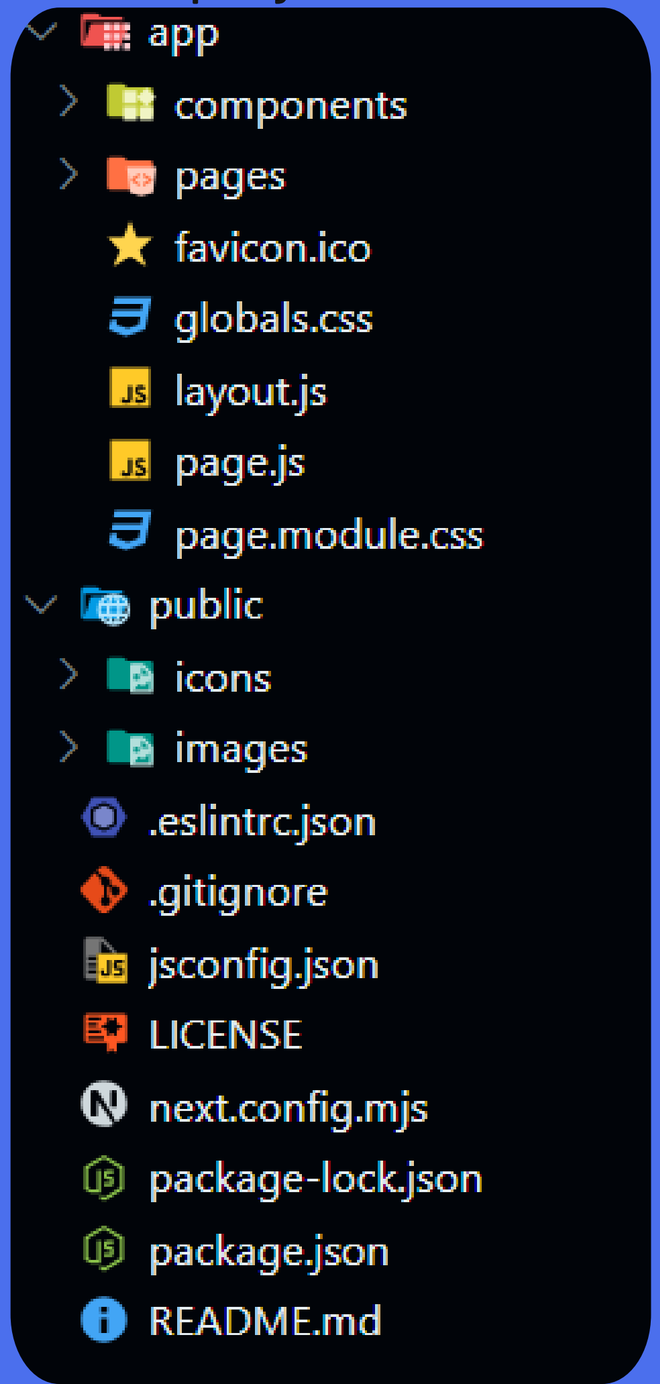
```
npm run dev
```

Organização das pastas no Next:

Padrão:

```
web_tech_page/
├── app/
│   ├── layout.js
│   └── page.js
├── node_modules/
├── package.json
├── next.config.mjs
├── README.md
├── package-lock.json
├── jsconfig.json
└── .gitignore
```

Nosso projeto no Git:



WebTech PUC Minas React.js Next.js

É Assim Que Acaba

07/08/2024

O Vingador da Iugoslávia 2

23/09/2024

A Substância

07/09/2024

Thunderbolts*

30/04/2025

É Assim Que Acaba

Data de Lançamento: 07/08/2024
Avaliação: 6.8

Descrição:
Lily Bloom, uma mulher que supera uma infância traumática para começar uma nova vida em Boston e perseguir o sonho de sua vida de abrir seu próprio negócio. Um encontro casual com o charmoso neurocirurgião Ryle Kincaid desencadeia uma conexão intensa, mas à medida que os dois se apaixonam profundamente, Lily começa a ver lados de Ryle que a lembram do relacionamento de seus pais. Quando o primeiro amor de Lily, Atlas Corrigan, reaparece repentinamente em sua vida, seu relacionamento com Ryle é abalado, e Lily percebe que deve aprender a confiar em sua própria força para fazer uma escolha impossível para o seu futuro.

02

Roteamento

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

02

Roteamento

Feito via **organização** dentro de arquivos presentes em **pastas**

- Simples (/about)
- Aninhada (/about/info)
- Aninhada dinâmica ([category]/[id].js)

01 Crie um **pasta “next”** no arquivo **app/**

02 Mova a **página Next** para dentro da pasta criado

03 Altere o nome da **página** para **page.js**

03

Otimização

- Links
- Imagens
- Fontes

Usando </Link>

01 Importe a biblioteca de rotas do next

02 Substitua o parâmetro **to** por **href**

Links / Navbar

```jsx
import style from './Navbar.module.css';
import Link from 'next/link';

function Navbar() {
  return (
    <header className={style.navbar}>
      <h1><Link href="/">WebTech PUC Minas</Link></h1>
      <nav>
        <ul>
          <li>
            <Link href="/">React.js</Link>
          </li>
          <li>
            <Link href="/next">Next.js</Link>
          </li>
        </ul>
      </nav>
    </header>
  );
}

export default Navbar;
```

Usando </Image>

01 Importe a biblioteca de imagens do next

02 Substitua a tag **img** por **Image**

03 Configure o a permissão de imagens externas

Imagens/Card

```
import style from './card.module.css';
import Image from 'next/image';

function Card({ banner, title, date, id }) {
  const [show, setShow] = useState(false);
  const router = useRouter();

  const handleClick = () => {
    setShow(!show);
  };

  const handleCardClick = () => {
    if (id) {
      router.push(`./next/${id}`);
    }
  };

  return (
    <div className={style.card} onClick={handleCardClick}>
      <figure>
        <Image src={banner} alt={title} width={640} height={360} layout="responsive" />
      </figure>
      <article>
        <h1>{title}</h1>
        <div>
          <time>{date}</time>
          {show ? (
            <AiFillHeart className={style.icon} onClick={handleClick} />
          ) : (
            <AiOutlineHeart className={style.icon} onClick={handleClick} />
          )}
```

01 Importe a biblioteca de fonts do next selecionando as fontes escolhidas

02 Crie as variáveis atribuindo os parâmetros

03 Atribua as fontes nos componentes desejados

Fonts/layout.js

```
import Navbar from "./components/Navbar";
import { Roboto, Poppins } from 'next/font/google';

const roboto = Roboto({
  weight: ['400', '700'],
  subsets: ['latin'],
  display: 'swap',
});

const poppins = Poppins({
  weight: ['700'],
  subsets: ['latin'],
  display: 'swap',
});

export const metadata = {
  title: "Create Next App",
  description: "Generated by create next app",
};

export default function RootLayout({ children }) {
  return (
    <html lang="en">

      <body className={roboto.className}>
        <Navbar className={poppins.className} />
        {children}
      </body>
    </html>
  );
}
```

Renderização

Rapidez

SEO

Experiência **usuário**

Client side

Execução do código no **navegador** do usuário. Permite **interatividade imediata** e mais **controle** sobre a interface. Pode ser **rápida e responsiva**, mas pode enfrentar **limitações** de performance quando há grande carga de dados ou operações intensas.

Static site

Execução do código no **client**. O servidor fornece uma **página HTML mínima**, enquanto o JavaScript no navegador gera e renderiza todo o conteúdo. Isso permite interações dinâmicas e uma **experiência de usuário responsiv**a, mas pode resultar em um **desempenho inicial mais lento**.

Server side

Execução do código no **servidor**. As páginas são **renderizadas** antes de serem enviadas ao navegador do usuário. Melhorando o **tempo de carregamento** inicial e o SEO, mas pode ser mais lento ao responder a interações **complexas**.

Client Side

```
"use client";
import { useEffect, useState } from "react";

export default function NextPage() {
  const [movies, setMovies] = useState([]);
  const [loading, setLoading] = useState(true);
  const apiKey = process.env.NEXT_PUBLIC_TMDB_API_KEY;

  useEffect(() => {
    const fetchMovies = async () => {
      try {
        const res = await fetch(
          `https://api.themoviedb.org/3/trending/movie/day?api_key=${apiKey}&language=pt-BR&page=1`
        );
        const data = await res.json();
        setMovies(data.results || []);
      } catch (error) { console.error("Erro ao buscar os filmes:", error);
      } finally { setLoading(false); }
    };
    fetchMovies();
  }, [apiKey]);

  const formatDate = (dateString) => {
    const [year, month, day] = dateString.split("-");
    return `${day}/${month}/${year}`;
  };
```

Server Side

```
import Image from 'next/image';
import styles from './page.module.css';

export default async function MovieDetailsPage({ params }) {
  const movieId = params.slug;
  const apiKey = process.env.TMDB_API_KEY;

  const res = await fetch(
    `https://api.themoviedb.org/3/movie/${movieId}?api_key=${apiKey}&language=pt-BR`
  );

  if (!res.ok) {
    throw new Error('Falha ao buscar os filmes');
  }

  const data = await res.json();
  const movieDetails = data;

  const formatDate = (dateString) => {
    if (!dateString) return "";
    const [year, month, day] = dateString.split("-");
    return `${day}/${month}/${year}`;
  };
```

05

Boas Práticas

Modularização

Modularize o código em pequenos componentes reutilizáveis

SSR e SSG

Use server-side rendering (SSR) ou static generation (SSG) conforme a necessidade

Organização

Organize as rotas de forma lógica e mantenha a estrutura de pastas clara

Priorize

Priorize a otimização de imagens e carregamento preguiçoso

SEO

Cuide do SEO ao implementar Meta Tags e Sitemap

06

Novas tendências

01 **Renderização Híbrida**

02 **Middleware**

03 **Internacionalização**

04 **TypeScript**

OBRIGADO(A)!