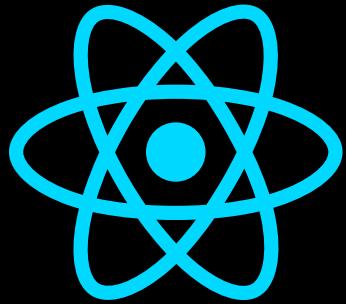


Fundamentos do React JS

O que iremos aprender?



- Introdução ao React
- Por que usar o React
- Iniciando um projeto
- Entendendo a estrutura de pastas do React
- Mão no código - Parte prática
- Importando, exportando e conectando arquivos no React
- Revisando Conceitos
- Colocando em prática!

Introdução ao React

React, também conhecido como React.js ou ReactJS, é uma biblioteca JavaScript de código aberto utilizada para construir interfaces de usuário (UI) interativas e reativas. Ela foi desenvolvida pelo Facebook e é amplamente utilizada para criar componentes de interface de usuário reutilizáveis e eficientes, especialmente em aplicações de página única (Single Page Applications - SPAs).

Um Aplicativo de Página Única, ou SPA (do inglês "Single Page Application"), é um tipo de aplicativo web que funciona dentro de uma única página da web, em contraste com os aplicativos tradicionais que têm várias páginas carregadas quando o usuário navega entre diferentes partes do site.

Introdução ao React

A ideia principal por trás de um SPA é que, quando o usuário interage com o aplicativo, em vez de carregar uma nova página do servidor a cada ação, o aplicativo carrega apenas os dados necessários e atualiza dinamicamente a página existente. Isso cria uma experiência de usuário mais rápida e fluida, pois os recursos e o conteúdo do aplicativo são pré-carregados e podem ser atualizados sem a necessidade de recarregar a página inteira.

Para criar essa interatividade, os SPAs geralmente usam tecnologias como JavaScript, AJAX (Asynchronous JavaScript and XML) e APIs de manipulação do histórico do navegador. Isso permite que as ações do usuário, como clicar em um botão ou preencher um formulário, sejam tratadas sem a necessidade de recarregar a página, resultando em uma experiência mais responsiva e parecida com a de um aplicativo de desktop.

Introdução ao React

No entanto, os SPAs também podem apresentar alguns desafios, como SEO (otimização para mecanismos de busca) e gerenciamento adequado da memória no navegador, já que todo o aplicativo e seus recursos permanecem carregados na mesma página.

Por que usar o React?

Pense no DOM (Document Object Model) como uma representação da sua página web. Quando algo muda na página, o navegador precisa atualizar o DOM **real**, o que pode ser um processo lento.

O React ajuda a tornar isso mais rápido. Ele cria uma cópia do DOM chamada de "Virtual DOM" na memória. É como se fosse uma versão de rascunho da página. Sempre que você faz uma mudança na sua página, essa mudança é primeiro feita no Virtual DOM.

Aqui está a mágica: o React compara o Virtual DOM com o DOM **real** e nota as diferenças. Ele calcula a maneira mais eficiente de aplicar essas mudanças no DOM **real**, e então faz apenas essas mudanças específicas. Isso é muito mais rápido do que recriar toda a página.

Por que usar o React?

Então, em resumo, o React ajuda a evitar atualizar toda a página toda vez que algo muda. Em vez disso, ele atualiza apenas as partes necessárias, graças ao Virtual DOM. Isso torna a aplicação mais rápida e responsiva!

Imagine que você está montando um quebra-cabeça para fazer uma página da web. Normalmente, você tem que lidar com duas coisas diferentes: a aparência da página (como ela se parece) e o comportamento dela (o que ela faz).

Iniciando um Projeto React

Para criarmos um projeto em React, utilizamos o comando `npx create-react-app`.



```
1 npx create-react-app nomedoprojeto
```

O comando `npx create-react-app` é uma forma conveniente de criar um novo projeto React sem a necessidade de instalar globalmente o pacote `create-react-app`.

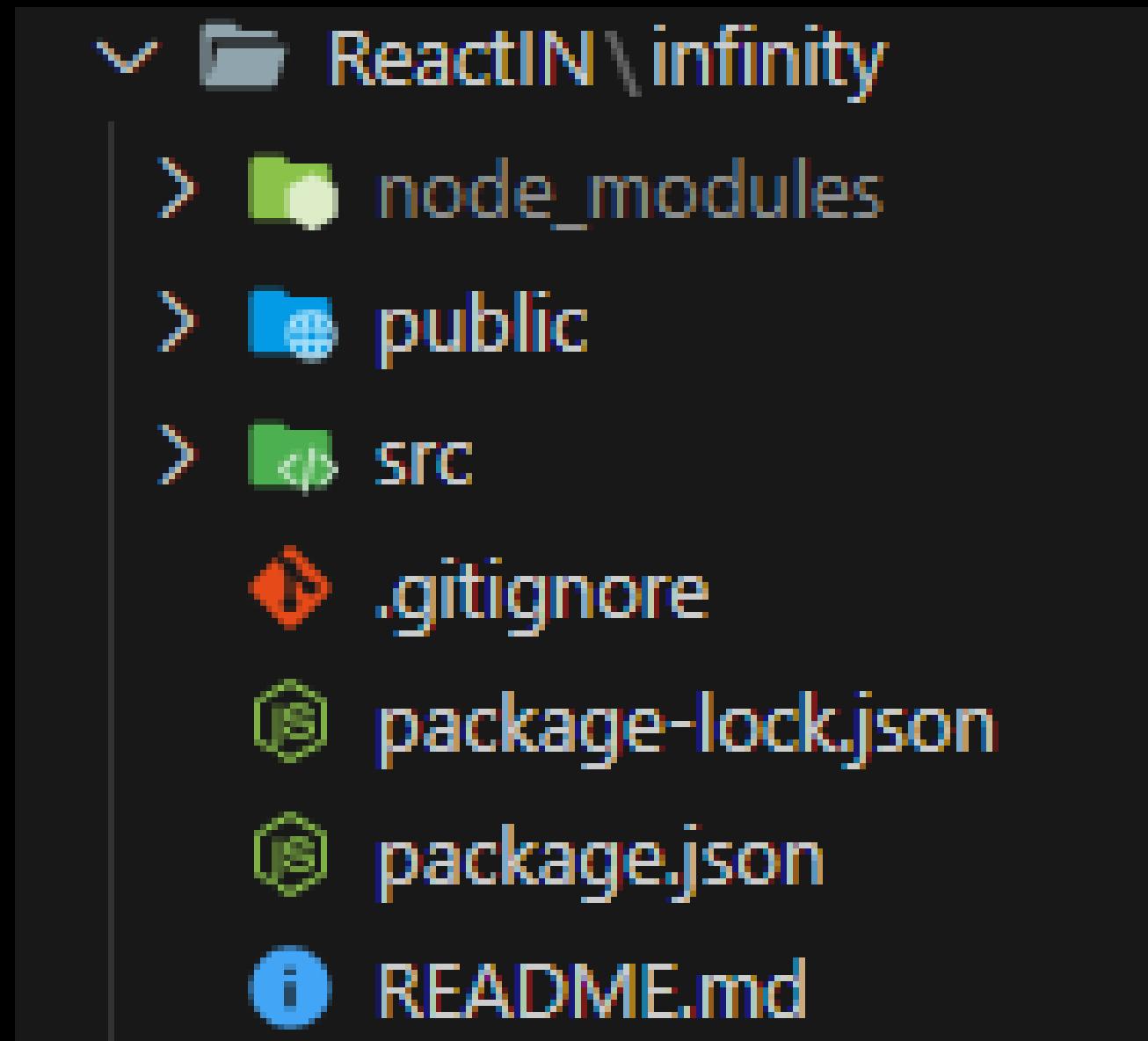
Iniciando um Projeto React

- 1.0 que é npx: npx é um utilitário de linha de comando que vem com o npm 5.2.0 ou superior. Ele é usado para executar pacotes Node.js que não estão instalados globalmente. Quando você executa um comando com npx, ele procura o pacote no seu node_modules/.bin ou instala temporariamente o pacote antes de executá-lo.
- 2.create-react-app: É um pacote npm mantido pela equipe do React. Ele é uma ferramenta de linha de comando que configura automaticamente um ambiente de desenvolvimento React, incluindo configurações como webpack, Babel e scripts de compilação.
- 3.Comando completo: npx create-react-app <nome_do_projeto>. Isso cria um novo projeto React com o nome especificado na pasta atual.

Regras para utilização

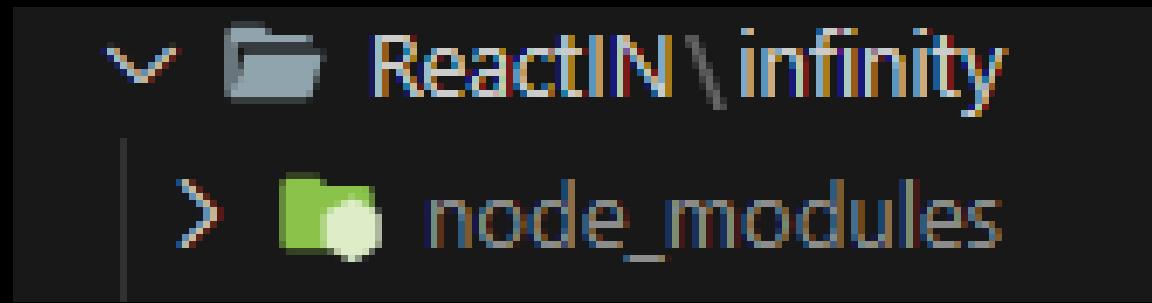
1. Certifique-se de ter o npm instalado em sua máquina. Normalmente, o npm é instalado automaticamente quando você instala o Node.js.
2. Certifique-se de ter uma conexão com a internet, já que o comando npx pode precisar baixar o pacote create-react-app temporariamente.
3. Escolha um nome significativo para o seu projeto e substitua <nome_do_projeto> pelo nome desejado.
4. Certifique-se de estar na pasta na qual deseja criar o projeto ou forneça um caminho absoluto para a pasta desejada.
5. Após a execução do comando, aguarde até que a instalação e a configuração estejam completas. Isso pode levar alguns minutos, dependendo da velocidade da sua conexão com a internet.
6. Após a conclusão, você verá instruções sobre como iniciar o servidor de desenvolvimento e outras tarefas comuns.

Entendendo a Estrutura de pastas do React



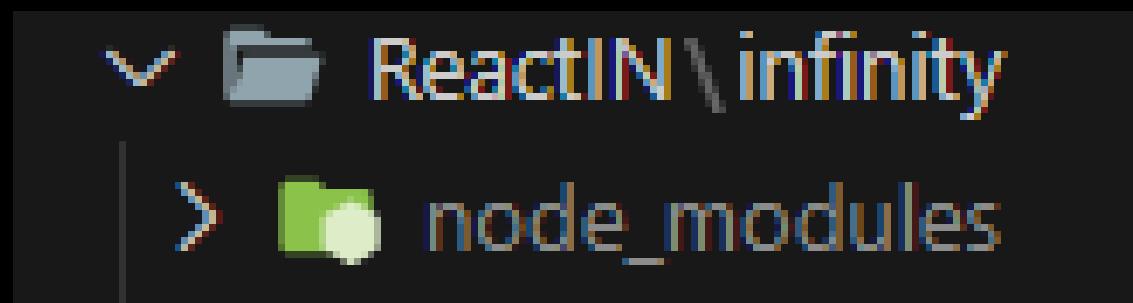
No modelo ao lado, criei uma pasta comum chamada ReactIN, e dentro dessa pasta dei o comando `npx create-react-app infinity`. Toda vez que começamos um projeto em React, virá nessa estrutura de organização. Vamos entender a funcionalidade de cada uma dessas pastas.

Pasta node modules



A pasta `node_modules` é onde o Node.js armazena todas as bibliotecas de terceiros (também conhecidas como pacotes ou módulos) que seu projeto depende. Quando você instala um pacote npm usando o comando `npm install`, o Node.js coloca esse pacote e todas as suas dependências na pasta `node_modules`. Isso é útil porque:

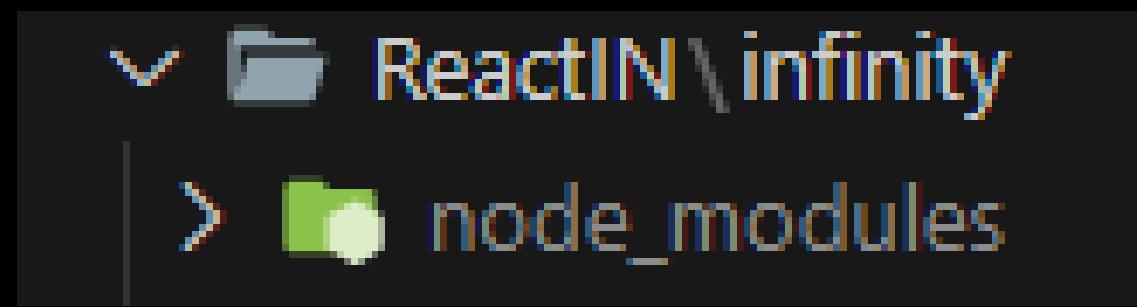
Pasta node modules



Gerenciamento de dependências: Mantém todas as bibliotecas e suas versões em um local específico, evitando conflitos entre diferentes projetos que podem precisar de versões diferentes das mesmas bibliotecas.

Portabilidade: Quando você compartilha seu código com outras pessoas ou implanta em outro ambiente, elas podem instalar todas as dependências necessárias apenas executando npm install, e o Node.js irá automaticamente baixar e instalar as bibliotecas listadas no arquivo package.json do seu projeto, colocando-as na pasta node_modules.

Pasta node modules

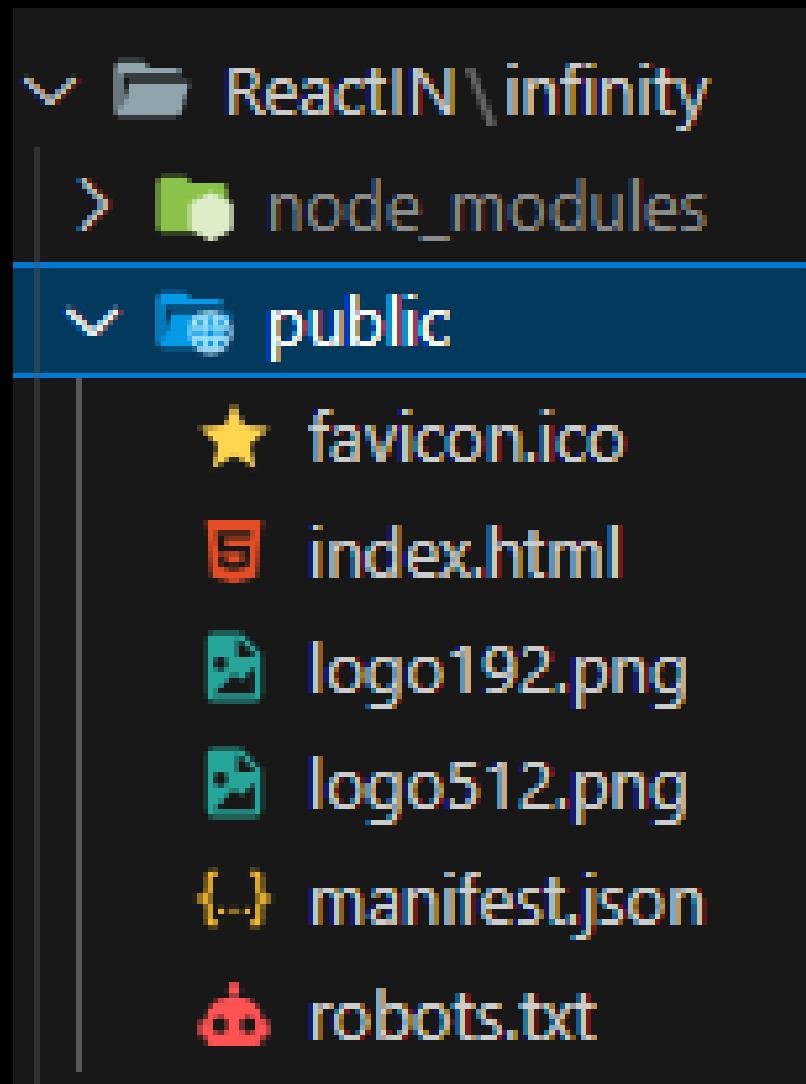


Fácil de gerenciar: Você não precisa se preocupar em incluir manualmente todas as bibliotecas necessárias em seu projeto. O Node.js cuida disso para você, simplificando o processo de desenvolvimento.

ATENÇÃO

No entanto, é importante não modificar diretamente nada na pasta `node_modules`, pois essas são bibliotecas de terceiros e qualquer alteração pode causar problemas de compatibilidade ou ser sobreescrita quando você atualizar essas bibliotecas. Em vez disso, você deve gerenciar suas dependências através do arquivo `package.json` e usar o comando `npm install` para instalar ou atualizar as bibliotecas necessárias.

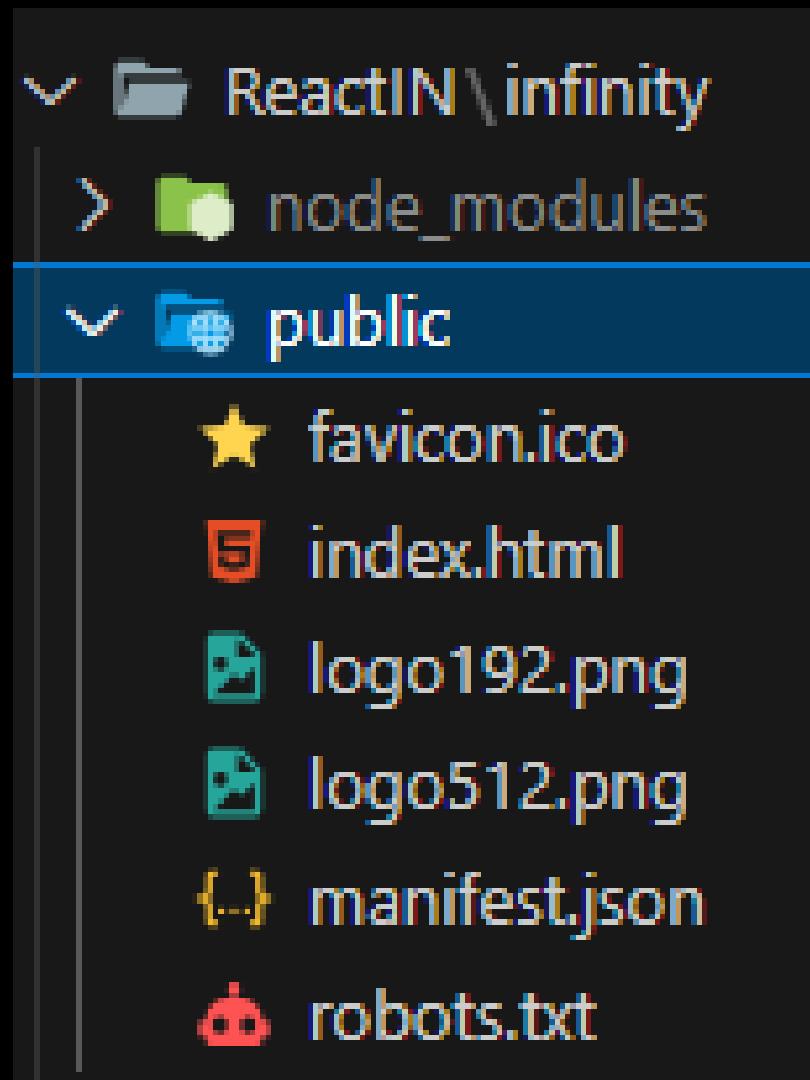
Pasta Public



A pasta public em um projeto React contém os arquivos estáticos que serão servidos diretamente para o navegador. Aqui está uma explicação simples de cada arquivo que geralmente vem dentro dela:

index.html: Este arquivo é o ponto de entrada principal para a sua aplicação React. Ele contém a estrutura básica de uma página HTML, como as tags `<html>`, `<head>` e `<body>`, e também uma div com um id específico onde a sua aplicação React será renderizada. Geralmente, você não precisa modificar este arquivo diretamente, pois o React manipula a renderização dinâmica da aplicação dentro dessa div.

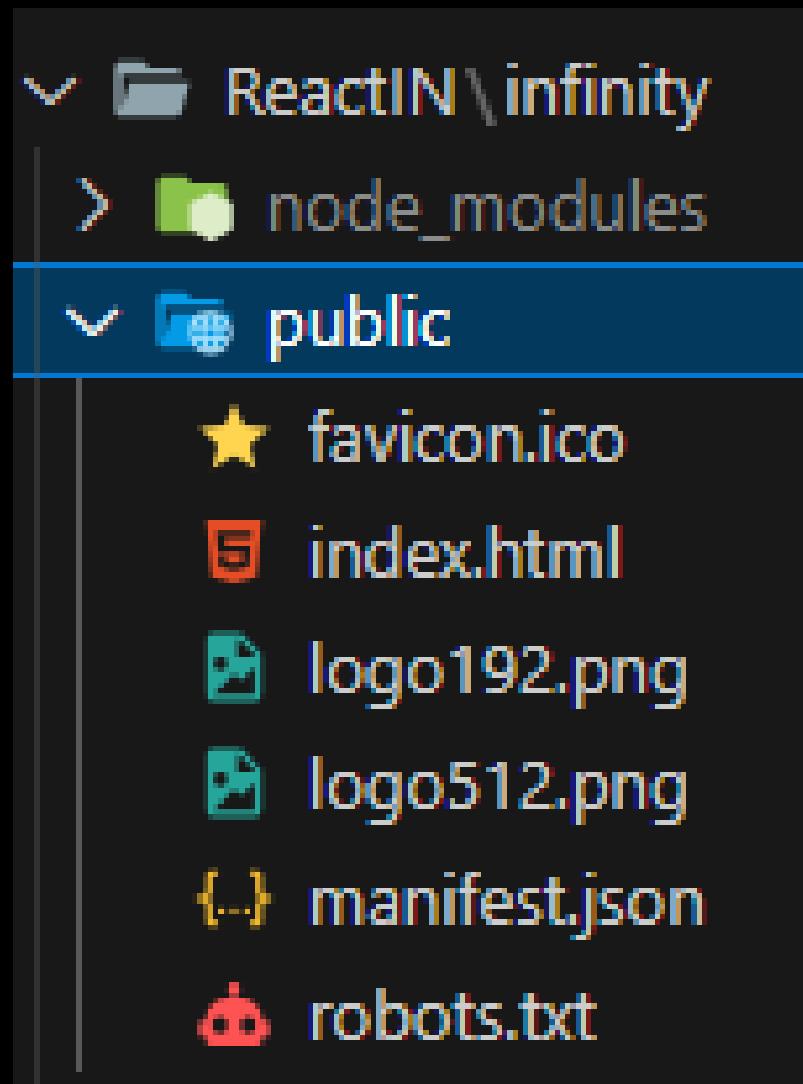
Pasta Public



favicon.ico: Este é o ícone da sua aplicação, que aparece na aba do navegador ou na barra de favoritos. O React cria automaticamente este arquivo para você, mas você pode substituí-lo por um ícone personalizado se desejar.

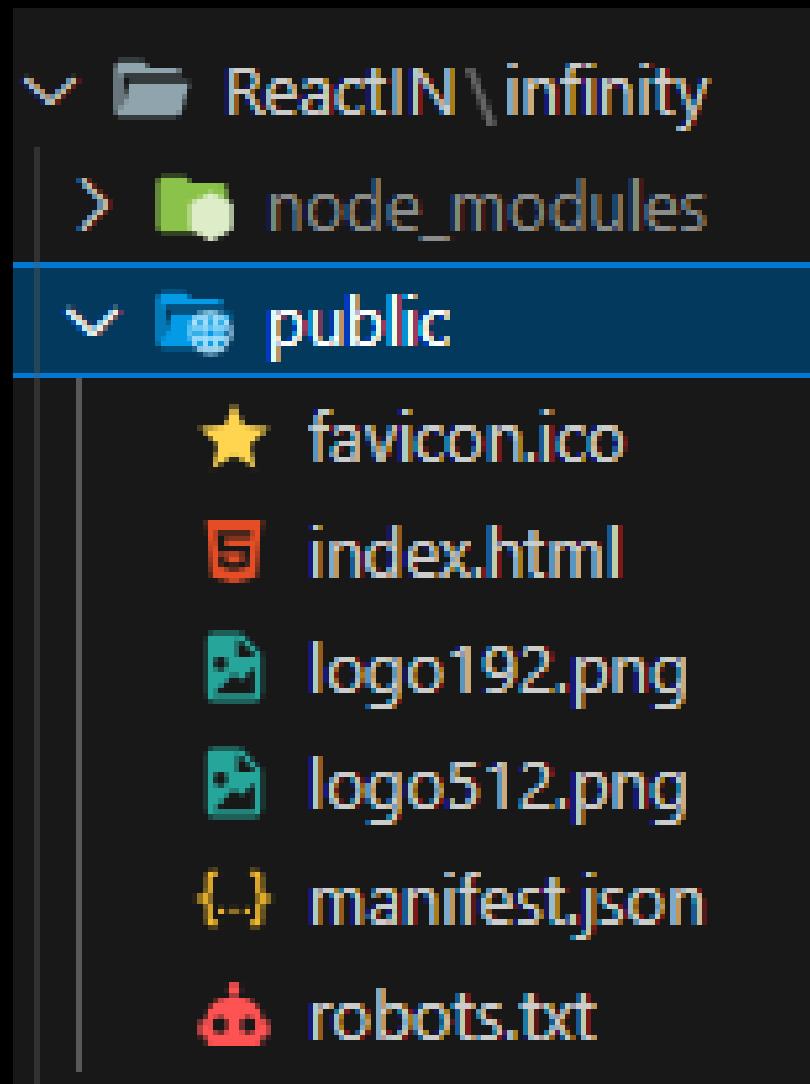
manifest.json: Este arquivo contém metadados sobre sua aplicação, como o nome, descrição, cor de tema e ícones usados em diferentes dispositivos e navegadores. Ele é útil para configurar a aparência e o comportamento da aplicação quando adicionada à tela inicial de dispositivos móveis ou desktop.

Pasta Public



robots.txt: Este arquivo é utilizado pelos motores de busca para entender quais partes da sua aplicação eles devem ou não rastrear. Ele fornece diretrizes sobre como os robôs dos mecanismos de busca devem interagir com o seu site. Por exemplo, você pode indicar quais páginas devem ser rastreadas ou não pelos motores de busca.

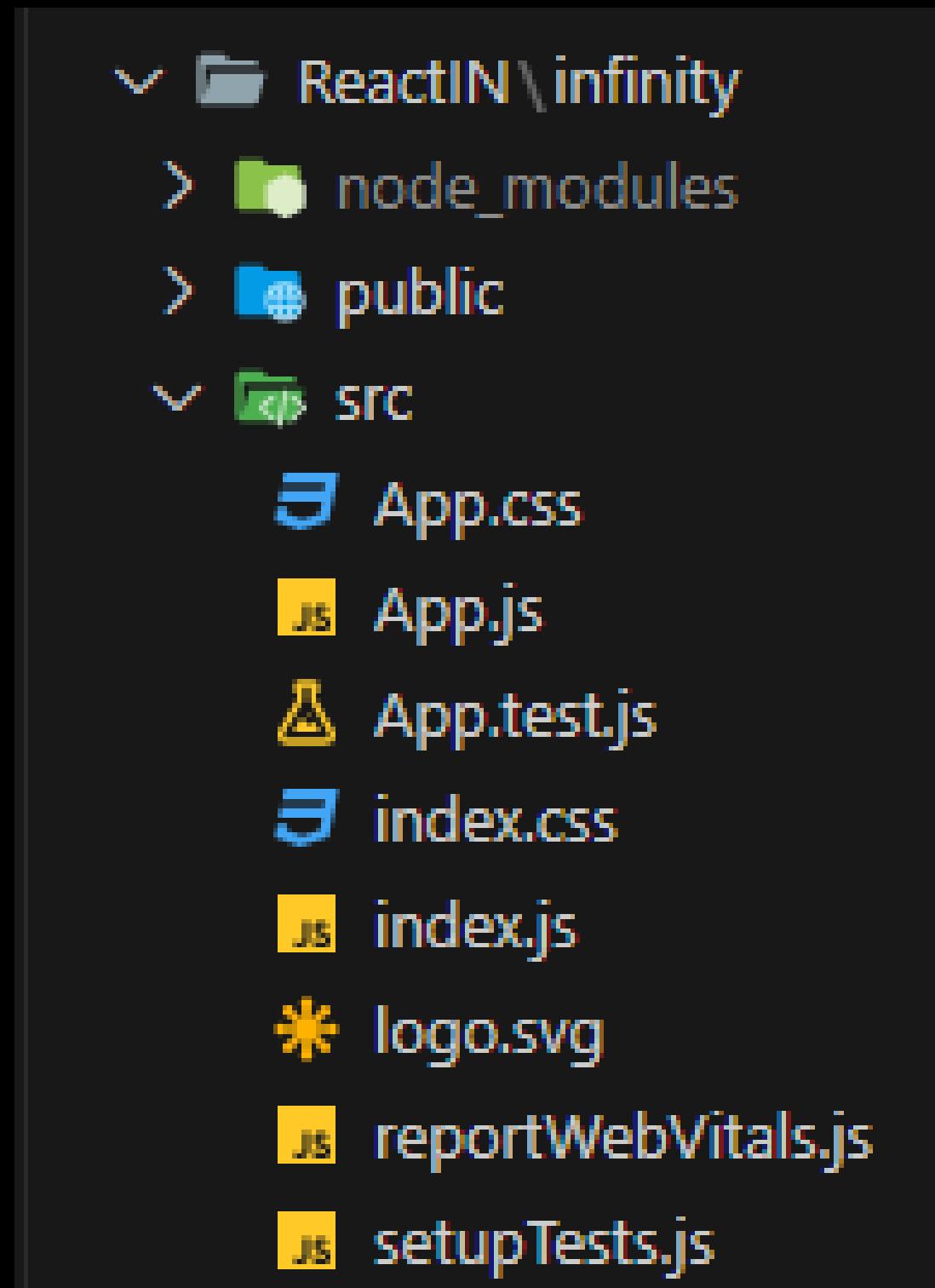
Pasta Public



arquivos de imagem, vídeo, fontes, etc.: Você pode colocar outros arquivos estáticos, como imagens, vídeos, fontes e qualquer outro recurso que sua aplicação precise, dentro desta pasta. Esses arquivos podem ser referenciados em seu código JavaScript ou HTML para serem exibidos ou utilizados em sua aplicação.

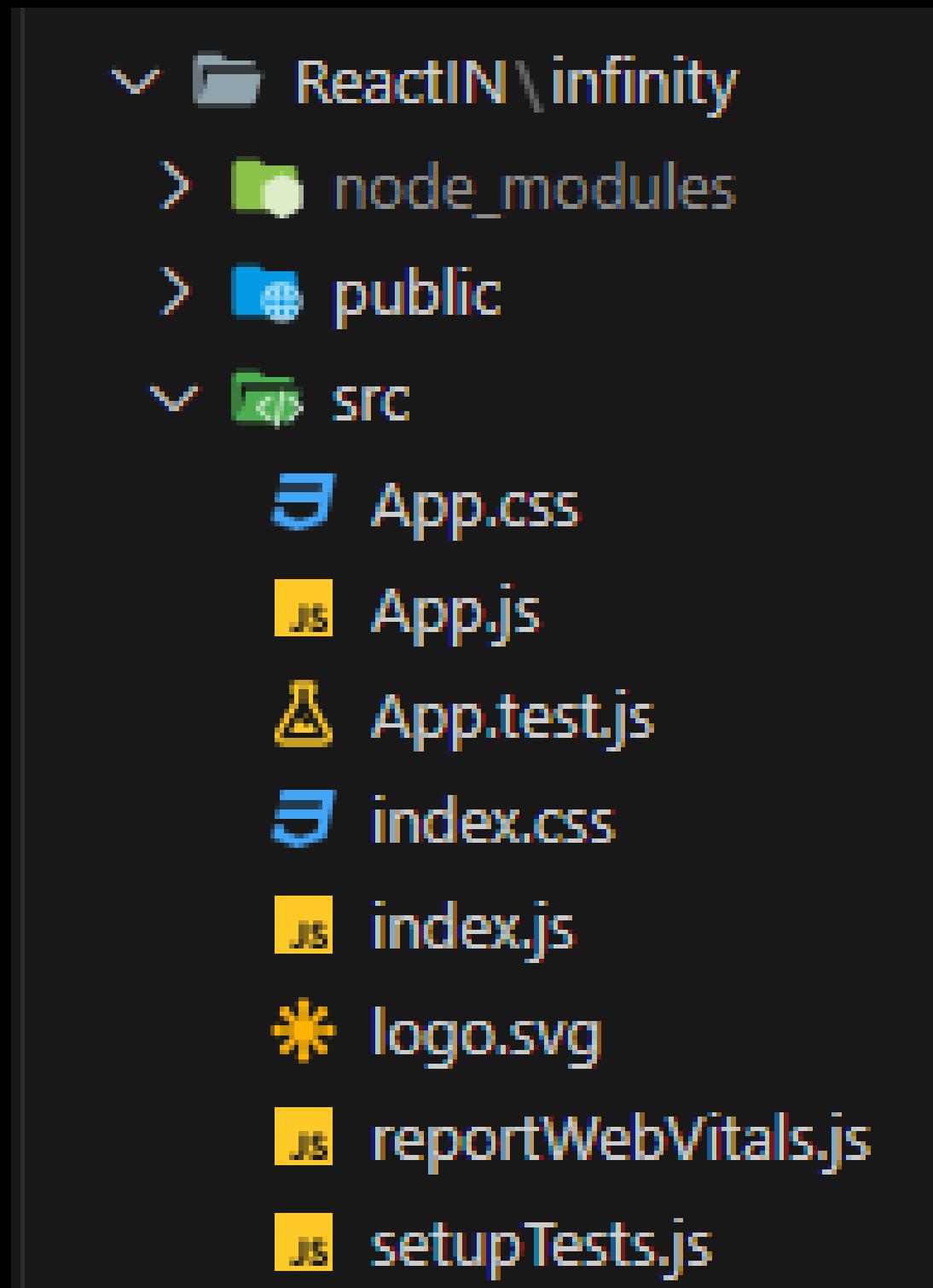
Em resumo, os arquivos da pasta public fornecem os recursos básicos necessários para iniciar sua aplicação e garantir que ela seja acessível e indexada corretamente pelos motores de busca.

Pasta src



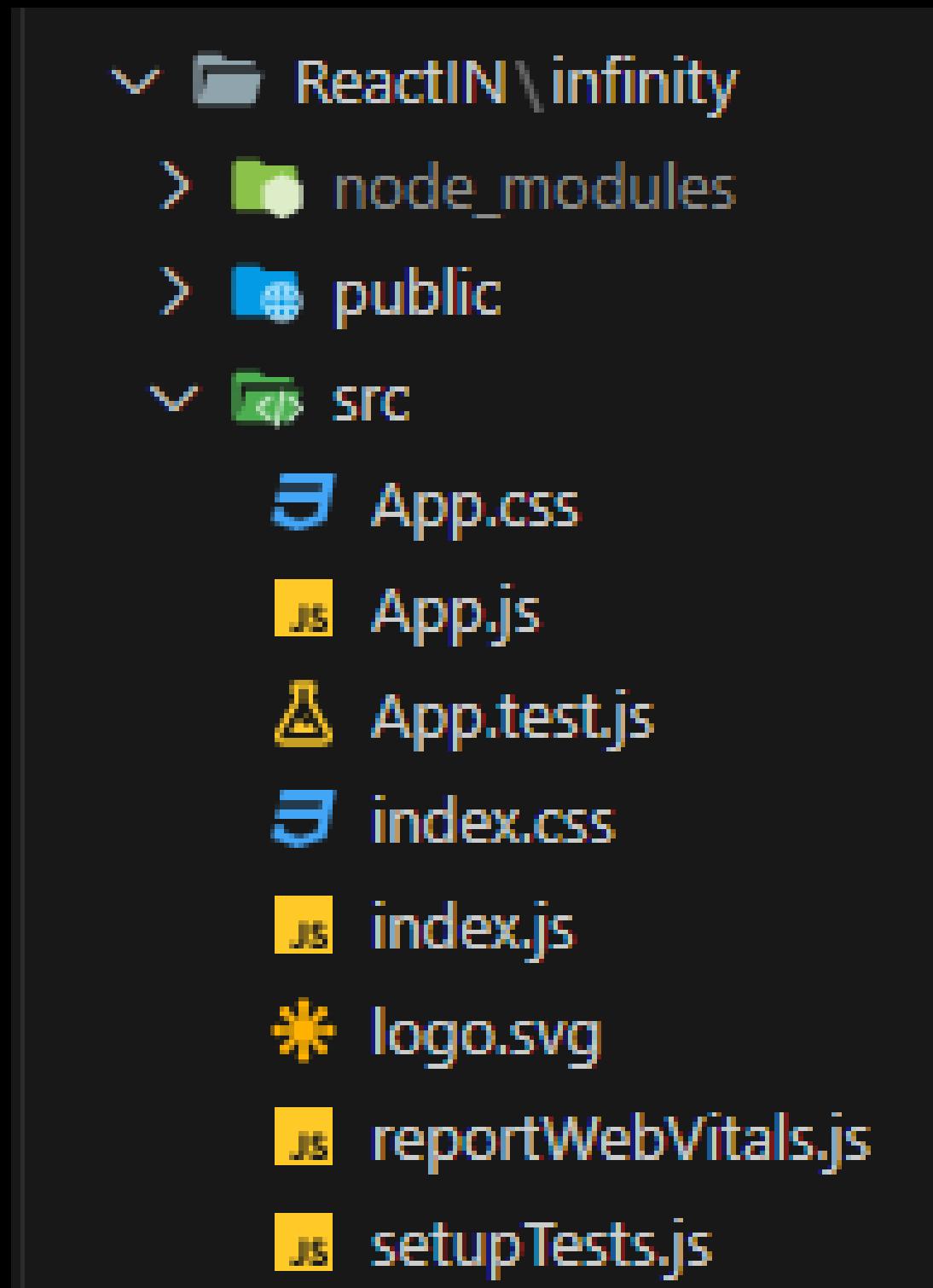
A pasta `src` em um projeto React é onde você irá escrever o código-fonte da sua aplicação. Ela contém os arquivos que serão compilados e transformados em código JavaScript que será executado pelo navegador. Aqui está uma explicação simples de cada arquivo comum encontrado dentro dela:

Pasta src



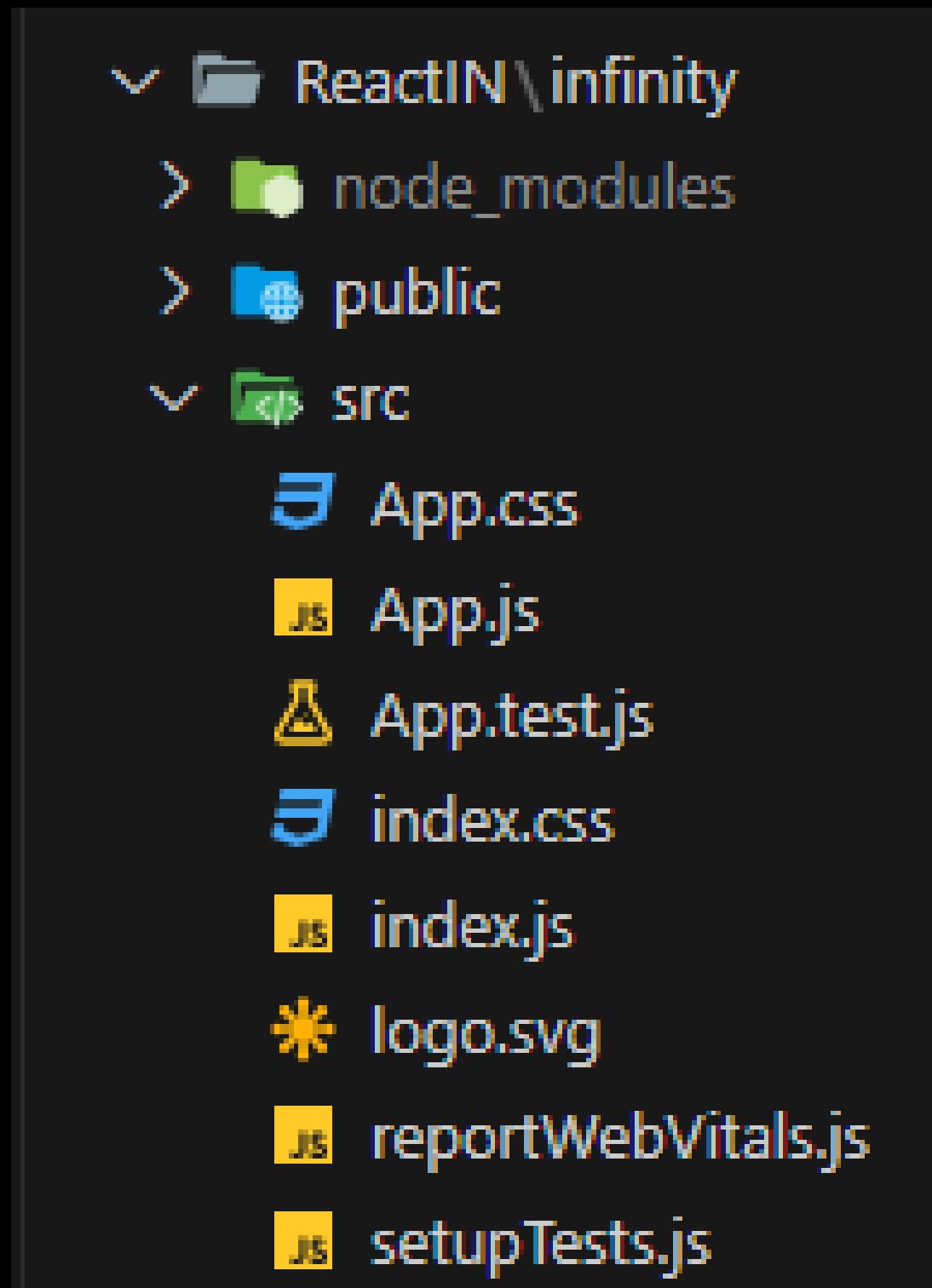
`App.css` (ou `App.scss`): Este é um arquivo de estilo CSS (ou SCSS, se você estiver usando Sass) associado ao componente App. Ele contém estilos específicos para o componente App e pode ser usado para estilizar os elementos dentro deste componente. Você pode adicionar regras de estilo aqui para controlar a aparência do seu aplicativo.

Pasta src



App.js: Este é um componente React que serve como o componente raiz da sua aplicação. Ele é responsável por definir a estrutura geral da sua interface de usuário e por incluir outros componentes conforme necessário. Você irá modificar este arquivo para adicionar ou modificar os elementos e a lógica da sua aplicação.

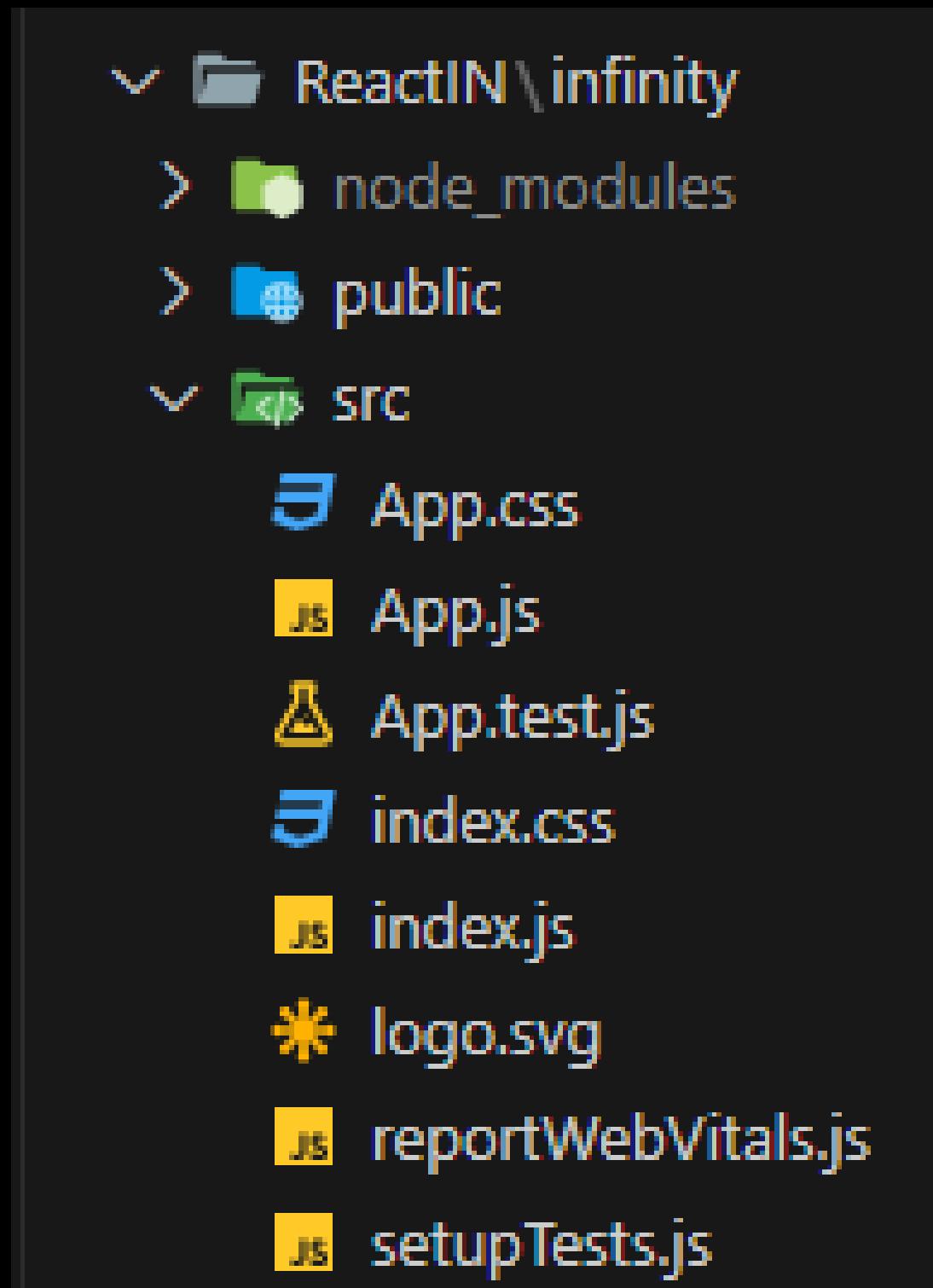
Pasta src



A página `App.test.js` em um projeto React serve para escrever testes automatizados para o componente principal `App` da sua aplicação.

Os testes automatizados são uma prática essencial no desenvolvimento de software, pois permitem verificar se o seu código está funcionando corretamente em diferentes cenários e situações, além de ajudar a detectar e corrigir erros mais rapidamente.

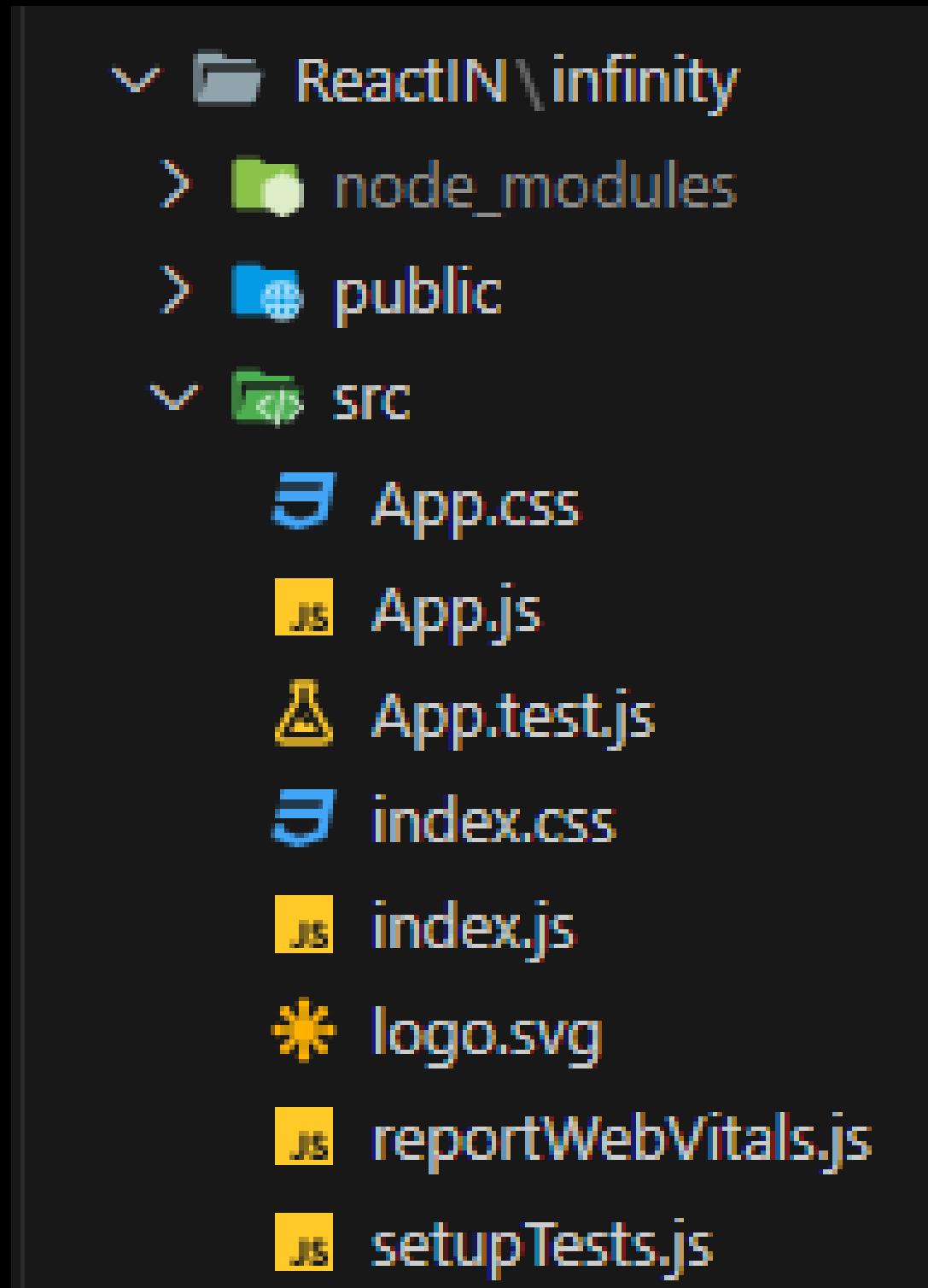
Pasta src



A página `index.css` em um projeto React serve para definir estilos globais que afetam toda a aplicação.

`index.js`: Este é o ponto de entrada principal da sua aplicação React. Ele é responsável por importar o componente raiz da sua aplicação e renderizá-lo no DOM. Geralmente, você não precisa modificar este arquivo, pois ele é criado automaticamente e mantido pelo Create React App ou pela configuração inicial do seu projeto.

Pasta src

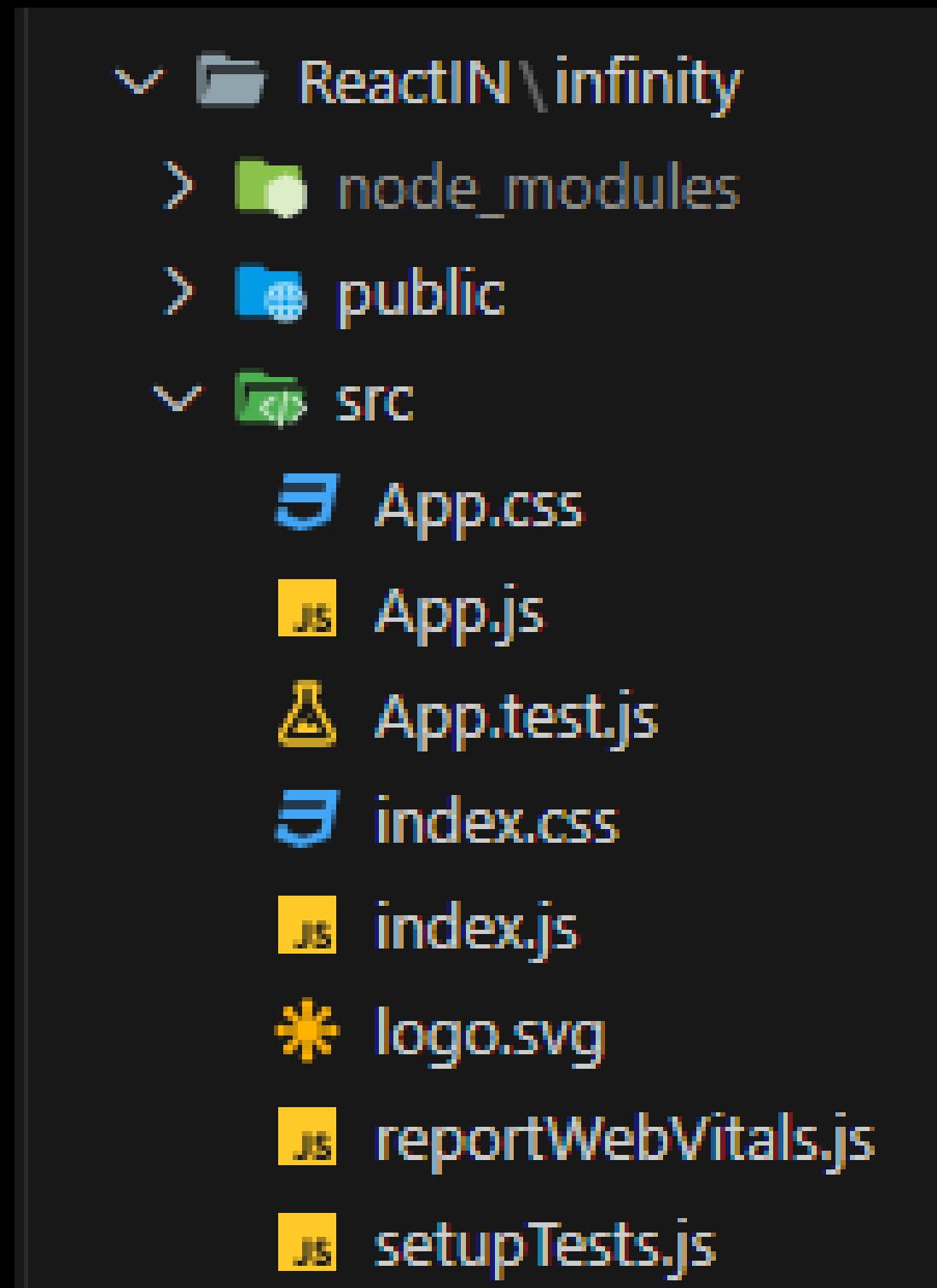


O `reportWebVitals` e o `setupTests.js` são ambos utilizados em projetos React para melhorar a qualidade e o desempenho da aplicação, mas têm propósitos diferentes:

`reportWebVitals`:

- Registra métricas de desempenho da sua aplicação.
- Principalmente usado para monitorar e analisar o desempenho ao longo do tempo.
- Ajuda a identificar e corrigir problemas de lentidão na aplicação.

Pasta src



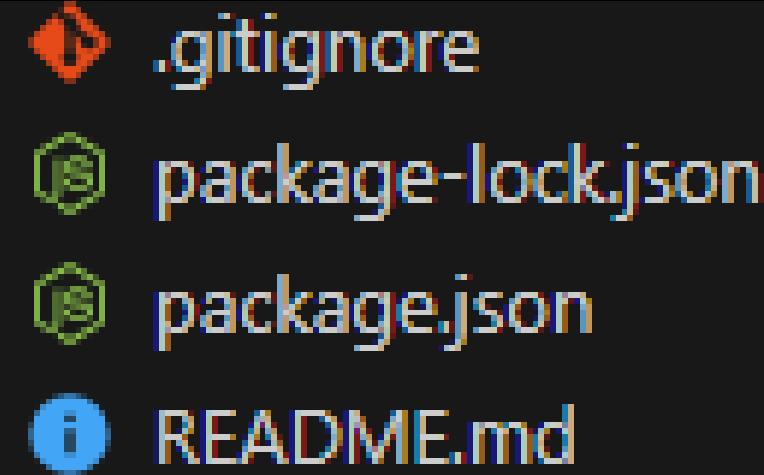
setupTests.js:

- Configura o ambiente de testes da sua aplicação.
- Usado para preparar o ambiente de teste antes de executar os testes automatizados.
- Garante que os testes sejam executados de forma eficiente e confiável.

Diferença principal:

- O reportWebVitals monitora o desempenho da aplicação, enquanto o setupTests.js configura o ambiente de testes.

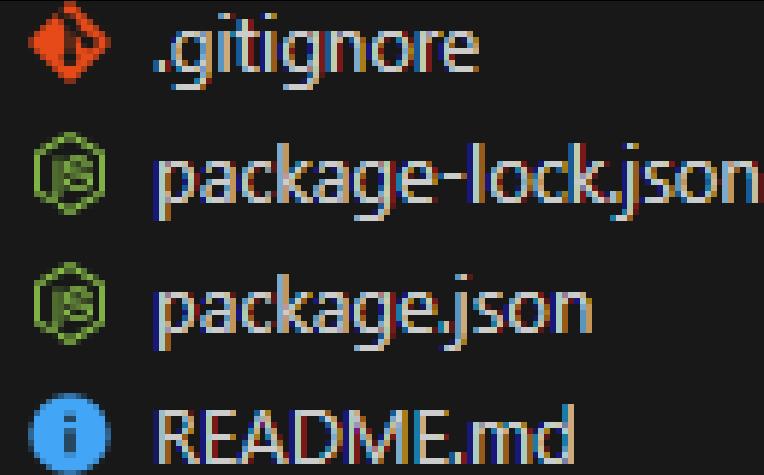
Gitignore, package-lock.json e package.json



A página `.gitignore` em um projeto React é um arquivo que especifica quais arquivos e pastas devem ser ignorados pelo Git, um sistema de controle de versão amplamente utilizado para o desenvolvimento de software colaborativo.

O arquivo `package-lock.json` é um arquivo gerado automaticamente pelo npm quando você instala ou atualiza as dependências de um projeto. Ele serve para garantir que as versões exatas das dependências instaladas sejam reproduzidas em diferentes ambientes de desenvolvimento, garantindo assim que todos os desenvolvedores tenham as mesmas versões de dependências instaladas.

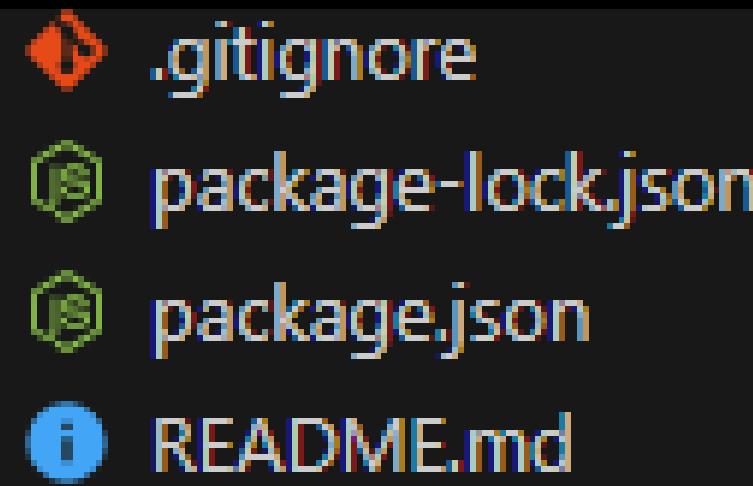
Gitignore, package-lock.json e package.json



A página `.gitignore` em um projeto React é um arquivo que especifica quais arquivos e pastas devem ser ignorados pelo Git, um sistema de controle de versão amplamente utilizado para o desenvolvimento de software colaborativo.

O arquivo `package-lock.json` é um arquivo gerado automaticamente pelo npm quando você instala ou atualiza as dependências de um projeto. Ele serve para garantir que as versões exatas das dependências instaladas sejam reproduzidas em diferentes ambientes de desenvolvimento, garantindo assim que todos os desenvolvedores tenham as mesmas versões de dependências instaladas.

Gitignore, package-lock.json, package.json e README.md



O package.json é um arquivo usado em projetos Node.js para listar dependências, configurar o projeto, definir scripts de execução e fornecer informações sobre o projeto. Ele é essencial para gerenciar pacotes e automatizar tarefas no desenvolvimento de software.

O README.md é um arquivo de texto usado para fornecer informações sobre o projeto, incluindo sua descrição, instruções de instalação, uso e outras informações importantes para os colaboradores e usuários do projeto. Ele serve como uma documentação rápida e acessível para entender o propósito e funcionamento do projeto.

Vamos começar?

Mão no código!

Antes de começarmos de fato a mexer no nosso projeto em react, vamos ver como ele vem por padrão? Basta entrar na sua pasta usando o comando `cd <nomedapasta>` em seguite `npm start`.



```
1 cd nomedoprojeto
```

```
2 npm start
```

Mão no código!

Agora começaremos a modificação do projeto. Vamos com calma e devagar já que são muitas pastas. Dentro da pasta nodemodules, não alteramos NADA !

Pasta public

Apagaremos todos os arquivos com excessão do index.html;
Dentro do index.html, apagaremos as linhas: 5, 12 e 17.

Pasta src

Apagaremos os arquivos: `reportWebVitals`, `App.test.js`, `SetupTestes.js` e `logo.svg`. APAGAR ESSES ARQUIVOS VAI RESULTAR EM UM ERRO, É NORMAL, AINDA NÃO TERMINAMOS. Os únicos arquivos que devem permanecer são o `app.css`, `app.js`, `index.js` e `index.css`.

Os erros que estão dando é porque dentrodos arquivoc que sobraram, está sendo importado os arquivos que apagamos, e como eles não existem mais, resulta em erros.

Dentro do `App.js`:

Apague a linha 1 e tudo o que está dentro do parênteses da palavra `return()`

Pasta src - App.js

ATENÇÃO

Apagar tudo o que está dentro do `return` na pasta `app.js` também resultará em um erro. É NORMAL. No React, um fragmento (ou Fragment) é uma maneira de agrupar múltiplos elementos sem adicionar um nó extra ao DOM (Documento Objeto do Modelo). Em outras palavras, você pode usar fragmentos para retornar vários elementos adjacentes de um componente, sem precisar criar um elemento extra de container.

Mas o que é um fragment?

Pasta src - App.js

```
1 import './App.css';
2
3 function App() {
4   return (<>
5     </>);
6 }
7
8 export default App;
```

fragment são esses sinais do html que estamos acostumados. Mas sem necessariamente ter um nome para a tag. Não esqueça: Ao criar HTML no react, precisa sempre estar envolto em um elemento pai, em uma tag.

Pasta src - index.js

Na pasta index.js, iremos apagar as seguintes linhas:

```
1 import reportWebVitals from './reportWebVitals';
```

Apagamos o arquivo reportWebVitals, então obviamente tentar importá-lo resultará em um erro. Apagaremos também a chamada da função na última linha do arquivo index.js:

```
reportWebVitals();
```

Como importar arquivos no React

Para importarmos uma imagem, por exemplo, após colocarmos a imagem em uma pasta, importamos da seguinte forma:

```
1 import logo from "../../assets/logo-jornal.png";
2 //logo é um nome fictício criado por mim;
3 //Dentro do src da imagem, ao invés de passar o
4 //caminho da imagem, passamos a palavra logo entre
5 // chaves
```

```
<img src={logo} alt="#" />
```

Como importar arquivos no React

Para importar arquivos css, por exemplo, usamos apenas o `import`, e passamos o caminho entre aspas:

```
1 import "./Login.css";
```

Ao criarmos uma função ou componente no React, ela precisa ter a capacidade de “ser chamada” em outro local. Para isso, usamos o `export`, no final da função:

```
1 //conectando um arquivo css, adicionando estilização
2 //a minha página
3 import './App.css';
4
5 //criando a função home, que vai renderizar minha página home
6 function Home() {
7   return (<>
8     <h1>Conteúdo da minha home</h1>
9   </>);
10 }
11
12 //tornando o meu conteúdo
13 //possível de ser chamado
14 //em outra página.Quando você
15 // usa export default em um
16 //arquivo React, está
17 //exportando por padrão o
18 //componente principal ou a
19 // função do componente do
20 //arquivo.
21 export default Home;
```

Revisando conceitos

Agora que removemos aquilo que não será utilizado na aula de hoje, vamos recapitular os conceitos vistos em sala de aula para, em seguida, começarmos a atividade!

Revisando conceitos

O que é React?

React é uma biblioteca JavaScript de código aberto usada para construir interfaces de usuário interativas e reativas em aplicativos da web. Desenvolvida pelo Facebook, React é altamente popular devido à sua eficiência, facilidade de uso e reutilização de componentes.

Revisando conceitos

O que é SPA?

Uma Single Page Application (SPA) é um tipo de aplicação web que funciona dentro de uma única página da web, sem precisar recarregar a página inteira durante o uso. Em vez disso, as interações do usuário são tratadas dinamicamente no navegador, geralmente usando JavaScript. Isso significa que, quando você navega em uma SPA, o conteúdo muda sem a necessidade de carregar uma nova página.

Revisando conceitos

O que é o DOM Virtual?

O DOM (Document Object Model) é uma representação da estrutura de um documento HTML. Quando você visita uma página da web, o navegador cria uma estrutura de árvore com todos os elementos HTML da página. O DOM é essa estrutura, e os navegadores usam o DOM para renderizar e manipular a página.

O DOM Virtual é uma técnica usada em muitos frameworks de JavaScript, como React e Vue.js. Em vez de manipular diretamente o DOM do navegador, esses frameworks criam uma representação virtual da árvore DOM em JavaScript. Isso permite que o framework faça alterações no DOM virtual de forma eficiente e, em seguida, atualize apenas as partes relevantes da página no navegador, em vez de recriar toda a árvore DOM.

Revisando conceitos

O que é a sintaxe JSX?

No React, os elementos precisam sempre estar encapsulados devido à sintaxe JSX e às regras de renderização. Aqui estão algumas razões principais:

Sintaxe JSX: JSX é uma extensão de sintaxe JavaScript que permite escrever código HTML-like dentro do JavaScript. No entanto, o JSX requer que os elementos tenham um único elemento pai. Isso significa que você não pode retornar múltiplos elementos irmãos diretamente de um componente sem encapsulá-los dentro de um elemento pai. Por exemplo:

Revisando conceitos

```
1 // Isso não é permitido em JSX
2 return (
3   <h1>Título</h1>
4   <p>Parágrafo</p>
5 );
6
7 // Em vez disso, você precisa encapsulá-los em um elemento pai, como um div:
8 return (
9   <div>
10    <h1>Título</h1>
11    <p>Parágrafo</p>
12  </div>
13 );
```

Revisando conceitos

Como criar um projeto React?

Para criar um projeto React, usamos o comando `create-react-app` nome do projeto.

Como colocar o projeto React pra rodar?

navegue até a pasta com o comando `cd nome da pasta`, em seguida, use o comando, `npm start`

Revisando conceitos

Como importar uma imagem no react?

Usamos a palavra `import` seguida de um nome a nossa escolha, em seguida a palavra `from`, e, entre aspas, o caminho até a imagem!

Como tornar nossas funções exportáveis?

Usando o comando `export default nomeafunção` ao final da página!

Colocando em prática

Crie seu currículo no react. Vamos criar o HTML dentro do return do app.js e a estilização é toda feita no app.css, como vimos anteriormente nas aulas de front!