

# Guia de estudos React

---

## O que é React?

---

### Uma biblioteca JavaScript para criar interfaces de usuário

#### Declarativo

React faz com que a criação de UIs interativas seja uma tarefa fácil. Crie views simples para cada estado na sua aplicação, e o React irá atualizar e renderizar de forma eficiente apenas os componentes necessários na medida em que os dados mudam.

Views declarativas fazem com que seu código seja mais previsível e simples de depurar.

#### Baseado em componentes

Crie componentes encapsulados que gerenciam seu próprio estado e então, combine-os para criar UIs complexas.

Como a lógica do componente é escrita em JavaScript e não em templates, você pode facilmente passar diversos tipos de dados ao longo da sua aplicação e ainda manter o estado fora do DOM.

#### Aprenda uma vez, use em qualquer lugar

Você pode desenvolver novos recursos com React sem reescrever o código existente.

O React também pode ser renderizado no servidor, usando Node, e ser usado para criar aplicações mobile, através do [React Native](#).

## Preparando do ambiente de desenvolvimento

---

Instalando [Node](#)

Instalando [Yarn](#)

## Instalando React

no terminal digite o seguinte comando para instalação:

```
npm install -g react
```

## Entendendo o projeto criado

---

Nesse conteúdo você irá criar um projeto de blog pessoal em React utilizando todas as funcionalidades desenvolvidas no conteúdo de back-end: Crud completo da tabela de postagem e tema, sistema de autenticação por token, função de cadastro e login.

## Criando um projeto React

---

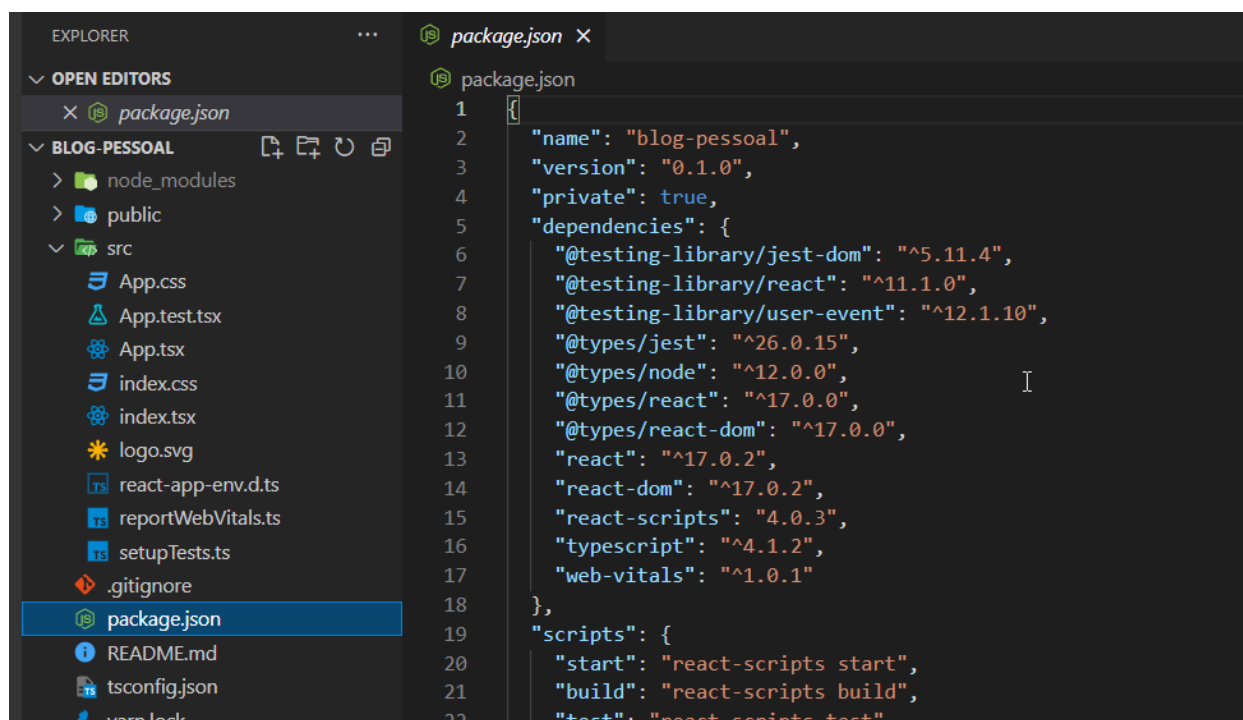
Criaremos um projeto react typescript utilizando o seguinte comando:

```
npx create-react-app blog-pessoal --template typescript
```

blog-pessoal é o nome do projeto que criaremos nesse conteúdo.

## Estrutura de básica de um projeto React

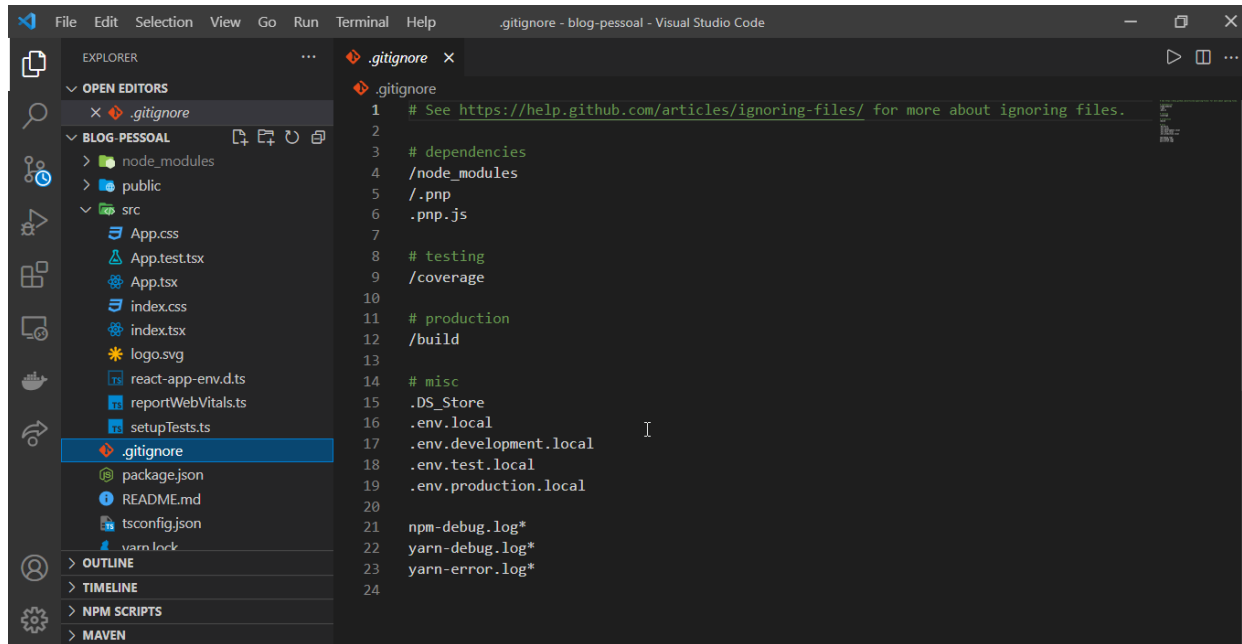
---



**Arquivo package.json** um arquivo com estrutura json que declara quais dependência estará sendo instalada no projeto React.

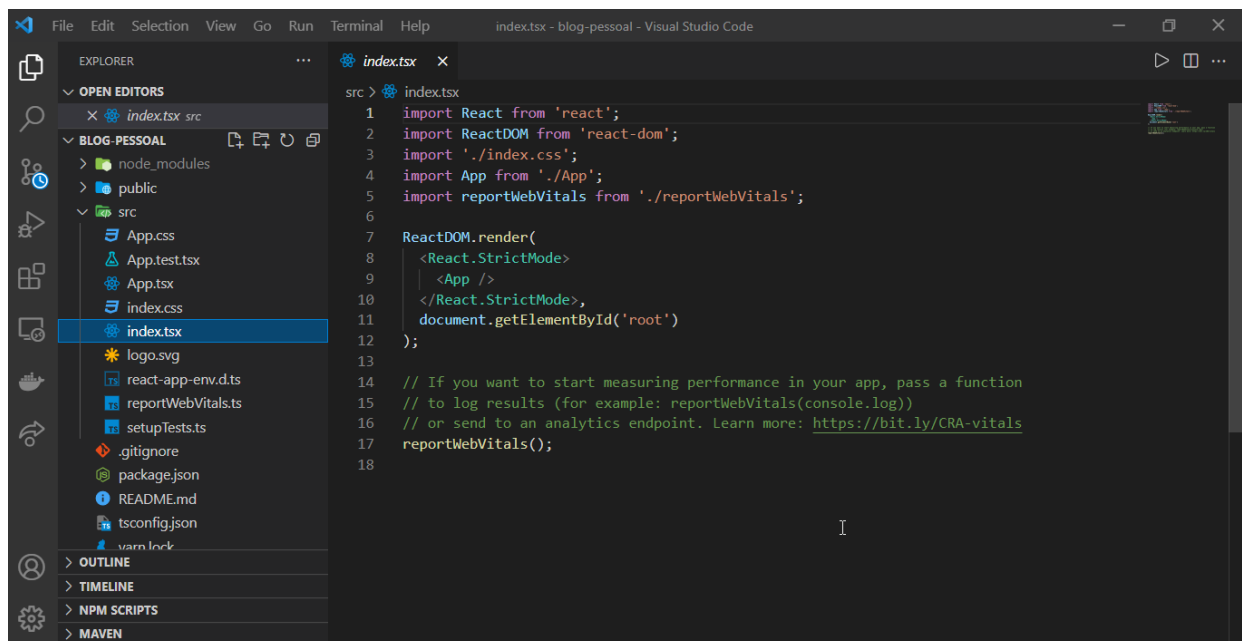
**Pasta node-module** uma pasta que guarda as dependencias que são instaladas no projeto react

**Arquivo .gitignore** um arquivo que determina quais pastas e arquivos serão ignorados pelo git ao subir para o servidor.



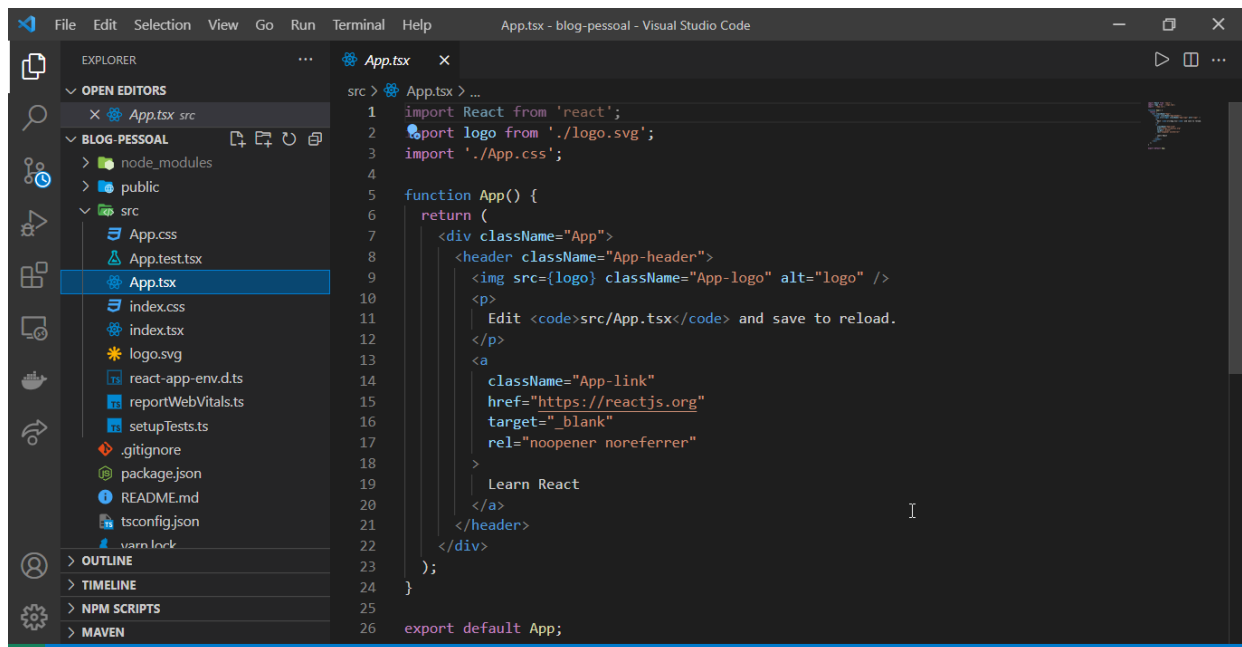
```
.gitignore
1 # See https://help.github.com/articles/ignoring-files/ for more about ignoring files.
2
3 # dependencies
4 /node_modules
5 /.pnp
6 .pnp.js
7
8 # testing
9 /coverage
10
11 # production
12 /build
13
14 # misc
15 .DS_Store
16 .env.local
17 .env.development.local
18 .env.test.local
19 .env.production.local
20
21 npm-debug.log*
22 yarn-debug.log*
23 yarn-error.log*
24
```

**Arquivo index.tsx** arquivo que executa o método render que renderiza os codigos react jsx.



```
src > index.tsx
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4 import App from './App';
5 import reportWebVitals from './reportWebVitals';
6
7 ReactDOM.render(
8   <React.StrictMode>
9     <App />
10   </React.StrictMode>,
11   document.getElementById('root')
12 );
13
14 // If you want to start measuring performance in your app, pass a function
15 // to log results (for example: reportWebVitals(console.log))
16 // or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
17 reportWebVitals();
18
```

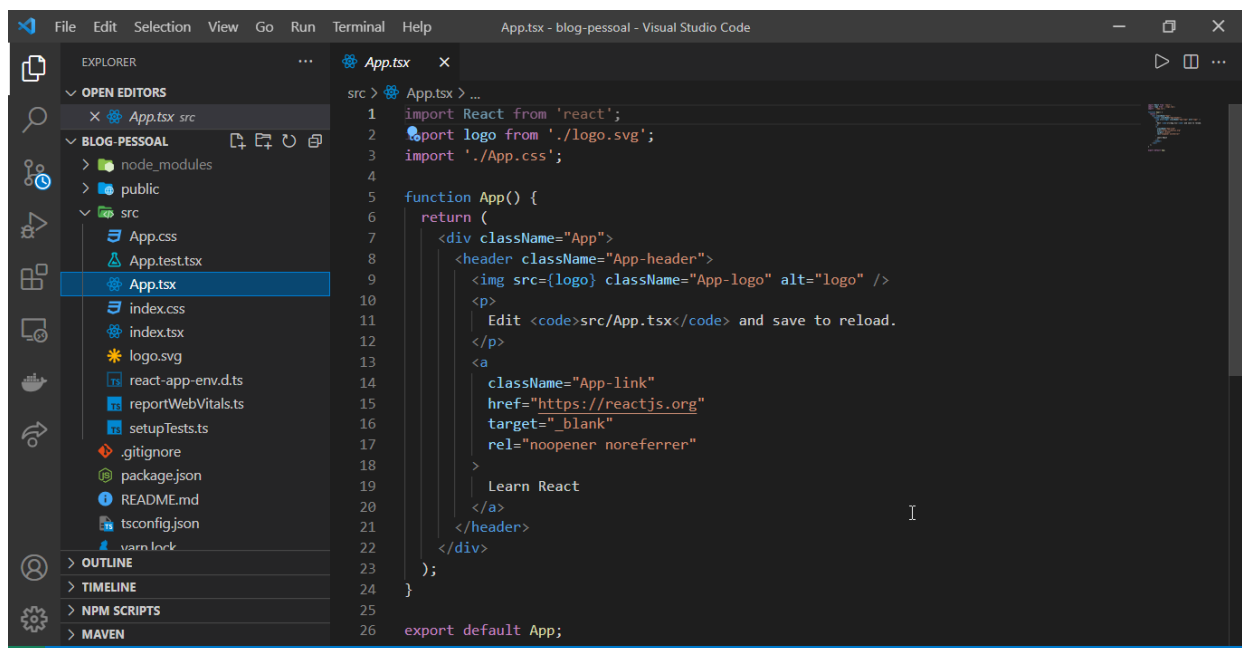
**Arquivo App.tsx** exemplo de componente react, com seu propria estrutura de tags e logica.



```
src > App.tsx > ...
1 import React from 'react';
2 import logo from './logo.svg';
3 import './App.css';
4
5 function App() {
6   return (
7     <div className="App">
8       <header className="App-header">
9         <img src={logo} className="App-logo" alt="logo" />
10        <p>
11          Edit <code>src/App.tsx</code> and save to reload.
12        </p>
13        <a
14          className="App-link"
15          href="https://reactjs.org"
16          target="_blank"
17          rel="noopener noreferrer"
18        >
19          Learn React
20        </a>
21      </header>
22    </div>
23  );
24 }
25
26 export default App;
```

## Estrutura de um componente React

Componentes permitem você dividir a UI em partes independentes, reutilizáveis e pensar em cada parte isoladamente



```
src > App.tsx > ...
1 import React from 'react';
2 import logo from './logo.svg';
3 import './App.css';
4
5 function App() {
6   return (
7     <div className="App">
8       <header className="App-header">
9         <img src={logo} className="App-logo" alt="logo" />
10        <p>
11          Edit <code>src/App.tsx</code> and save to reload.
12        </p>
13        <a
14          className="App-link"
15          href="https://reactjs.org"
16          target="_blank"
17          rel="noopener noreferrer"
18        >
19          Learn React
20        </a>
21      </header>
22    </div>
23  );
24 }
25
26 export default App;
```

Um componente React do tipo função é formado por:

Uma função javascript;

Um método return onde são renderizado as tags html;

E um export: onde permite ser exporta esse component para outro arquivo;