**React js.**

**O que é npm programação?**

**npm** é o gerenciador de pacotes padrão para o ambiente de tempo de execução JavaScript Node. js. Ele consiste em um cliente de linha de comando, também chamado de npm, e um banco de dados online de pacotes públicos e privados pagos, chamado de registro npm.

O **npm** é o Gerenciador de Pacotes do Node **(Node Package Manager)** que vem junto com ele e que é muito útil no desenvolvimento Node. Por anos, o Node tem sido amplamente usado por desenvolvedores JavaScript para compartilhar ferramentas, instalar vários módulos e gerenciar suas dependências.

**git init**

Inicializa um repositório no **github.**

**O que é npm init y?**

**npm init -y**. Simplesmente irá gerar um projeto npm vazio sem passar por um processo interativo. O -y significa sim.

**Instalações de bibliotecas:**

Instalações: "**npm i next@12.0.7 react@17.0.2** [**react-dom@17.0.2**](mailto:react-dom@17.0.2)"

**Criação de arquivo .gitignore**

O arquivo **.gitignore** é um arquivo de texto que diz ao Git quais arquivos ou pastas ele deve ignorar em um projeto. Um arquivo . gitignore local geralmente é colocado no diretório raiz de um projeto**.** Exemplo: **node\_modules**

Não há necessidade de enviar esse arquivo no o **github**, pois ele é muito pesado e o **github** não utilizará esses arquivos.

Configurar script:

**“scripts”: {**

**“dev”: “next dev”**

**}**

Criar pasta **pages.** E depois criar arquivo **index.js.** Criar **função HomePages** (dentro do arquivo index.js),e **exporta-la.**

Criar pasta **src (source:** fonte, origem)fora da pasta **pages.**

Dentro da pasta **src**, criar pasta **components** e criar todas as outras pastas do projeto dentro dela.

Uma função no **react** sempre retorna um html.

A diferença das **tags** do **html** para as **tags** do **react**, é que as **tags** do react tem letra maiúscula. Exemplo: **<Title />** ou **<Title><Title />**

Sempre que usarmos variáveis dentro do html do react, devemos usar chaves e *className*:

<h1 *className*={styles.title}>Meu Título</h1>

**O que é uma Props?**

As **props** são entradas que não podem ser alteradas dentro de um componente. Componentes devem, obrigatoriamente, apenas ler as **props**. Sendo assim, todos os componentes de **React** devem ser “puros” e não podem alterar nenhum valor das **props**.

**O que é Props e State?**

Apesar de ambos guardarem informações que influenciam no resultado da renderização, eles são diferentes por uma razão importante: props são passados para o componente (como parâmetros de funções), enquanto **state** é gerenciado de dentro do componente (como variáveis declaradas dentro de uma função).

Há duas formas de importar arquivos. Uma usando o **export default**. E a outra é usando as chaves, sem o **default**:

Exemplo: **export {NewSubtitle}** ou **export default NewSubtitle**

Propriedades não podem ser alteradas dentro do components. Funcionam como constantes.

Para que o código fique mais limpo, usar o **destructor**. E aí, não será necessário usar a **props**.

function Subtitle ( {text} ) {

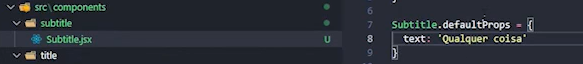
Return (

<h4>{text}</h4>

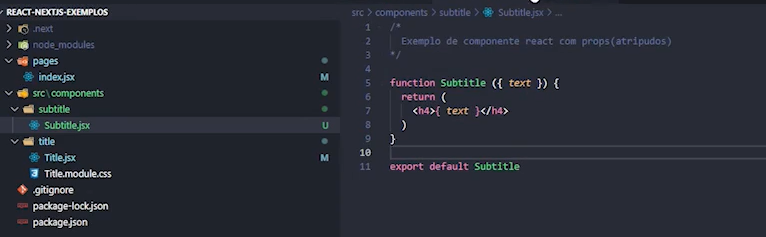
)

}

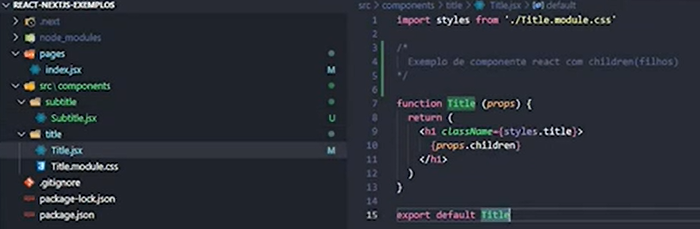
Também podemos passar um texto padrão, para quando não houver um texto definido:



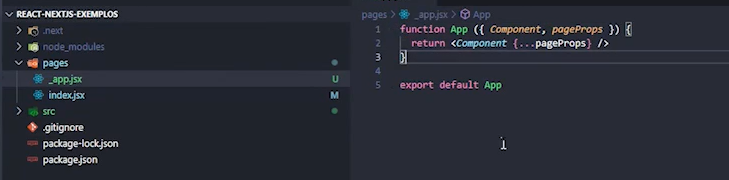
Exemplo de componente react com props (atributos):



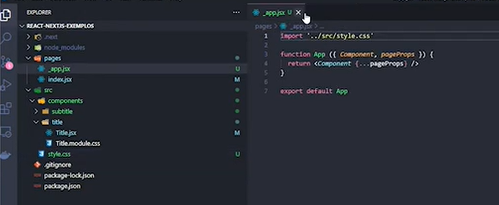
Exemplo de componente react com children (filhos):



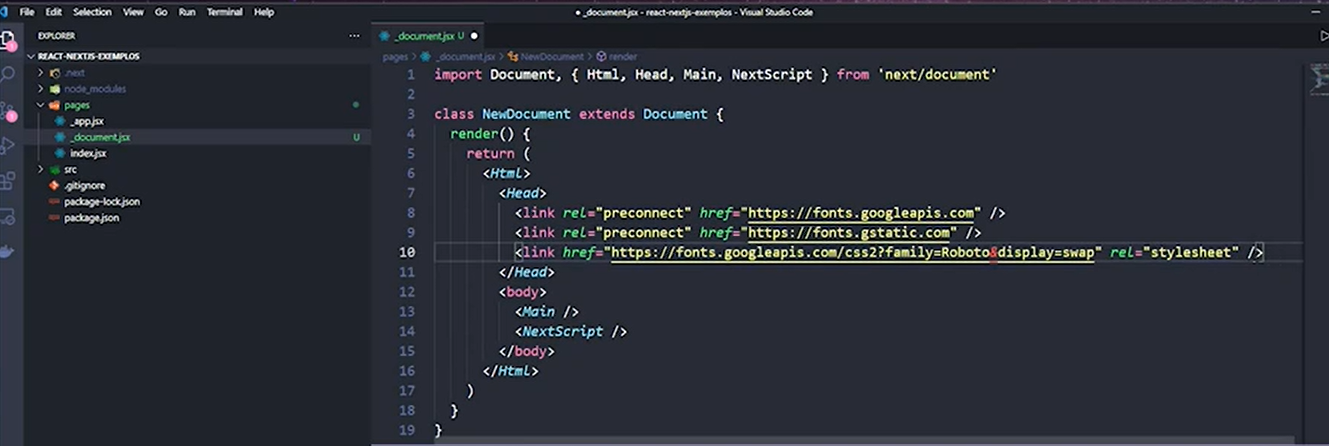
Padrão do next js



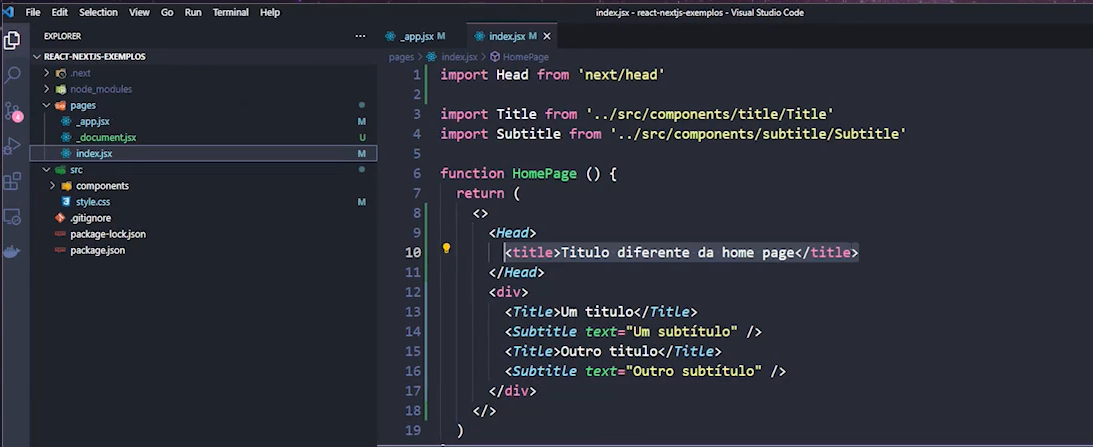
Criar o css global em src:



Arquivo \_document.jsx



Configurações globais:



Sempre que quisermos mostrar alguma coisa para o usuário na tela devemos usar o **useState**. Devemos criar a variável com **const** dentro da função.



O primeiro termo é a variável e o segundo é a função que altera essa variável.