**FrontEnd by React**

**Metodos de importação e exportação**

* Import xxx from ‘xxx’
* Export default xxxx

**Para iniciar diretório com as dependências prontas**

(npx create-react-app frontend)

Após app, dá-se o nome da pasta, que no meu caso foi “frontend”

**Como funciona o HTML no React**

Dentro do código a pagina ela busca o primeiro arquivo .html

E o nessa pasta estará referenciando um id=”root”

<body>

    <noscript>You need to enable JavaScript to run this app.</noscript>

    <div *id*="root"></div>

  </body>

Antes disso temos um arquivo responsável por fazer o link da pagina onde os códigos iram se armazenar

import React from 'react';

import ReactDOM from 'react-dom';

import App from './App';

ReactDOM.render( <*App* />, document.getElementById('root'));

ReactDOM é um pacote para conecção com a DOM. + render imprime o conteúdo em tela, envia via document.\*\*\* onde o método getElementById seleciona root, que é a ID da primeira página que o frontend vai procurar.

Então temos

A function App que é enviada via index.js pela DOM com o ID ‘root’ para index.html importando como <div id=’root’></>

**Conceitos importantes**

**Componente**

É uma função que retorna HTML:

function App() {

  return (

    <h1> helo world </h1>

  );

}

App é a função que retorna todo o código da página.

**JFX** (javaScript + xml)

Quando html está integrado dentro do javaScript

**Propriedades**

É tão igual aos atributos repassados aos componentes:

////Esta função esta me retornando um componente

function App() {

  return (

    <*Header*>

      Semana OmniStack

    </*Header*>

  );

}

export default App;

export default function Header({*children*}) {

    return (

        <header>

            <h1>{children}</h1>

        </header>

    );

}

**Estado**

Informação que será armazenada pelo componente.

Listagens e etc.

Para ser utilizada é preciso importar do React o useState

import React, {useState} from 'react';

Ex: {

const [counter, setCounter] = useState(0);

  function increment() {

    setCounter(counter + 1 )

  }

App de contar,

O useState retorna um array com duas posições

**[valor original, função de possível alteração]**

**OBS:** essa informação no estado n pode ser alterada então nós podemos subscreve-las.

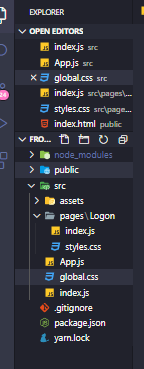
Sempre que um componente precisar armazenar uma informação dentro dele, nós não vamos criar uma variável comum, nós criamos um ESTADO, assim atualizamos as informações

**Pages**

Ele cria uma pasta para cada página, pois junta os arquivos de estilização e a index para importação.

Ex: página de login:

Src/pages/Logon/index.js and styles.css



**Estilização**

Abre um arquivo global do css para resetar as características padrões e fazer do zero as estilizações

Arquivo global.css



Rotas

(Npm install react-router-dom)

Rotas da primeira página