React

React = biblioteca front-end criada pelo FaceBook em 2013. Possui 2 premissas, componentização (aproveitar o possível dos componentes para criar novos, mantendo o padrão e alterando apenas o necessário) e gerenciamento de estado (react é reativo, o gerenciamento de estado diz respeito a reagir às novidades).
DRY = Don't Repeat Yourself.
SRP = Single Responsibility Principle.
npx create-react-app --template typescript --use-npm nome_projeto = digitando no terminal npx executa o react, ao invés de instalar no computador; typescript define a linguagem que será usada; npm diz qual será o gerenciador de versões.
cd nome projeto = digitando no terminal, entra na pasta do projeto.
npm run start = digitando no terminal, inicia o projeto.
Arquivos = o react mostra alguns arquivos no vscode:
Tsconfig.json = faz a compilação de Type pra JavaScript.
Readme.md = informa o objetivo do projeto, documentação, resume tudo.
Package.json = lista dependências, informações, versão do programa, autoria, scripts.
Package-lock.json = guarda o histórico das instalações.
Gitignore = lista o que deve ser ignorado no github.
SRC = principal, guarda os códigos em si. Possui as pastas: app.css (guarda o css), app.test.tsx (guarda o teste), app.tsx (guarda a exibição da tela), index.css (guarda os arquivos de reset), index.tsx, logo.svg (guarda a logo do react), react-app-env.d.ts, reportWebVitals.ts, setupTest.ts.
Node-modules = lista os módulos instalados pelo Node.
Projeto Novo = para começar um novo projeto no react, excluir do src o app.css, app.test.tsx, logo.svg, reportWebVitals.ts e setupTest.ts. No index.tsx excluir a importação do reportWebVitals.ts e sua utilização nas últimas linhas do código. No app.tsx excluir a importação do app.css e logo.svg, além do header inteiro. Criar pastas separadas pra componentes.
Props = adicionam propriedades a um componente, permitem individualizar cada um.
Imagens = importar as imagens usadas para a pasta do código.
Normalize = css reset do react, precisa instalar ele no terminal e importar no código pra funcionar.
ESLint = usado pra padronizar o código. Para iniciar, digitar no terminal do vscode npm init @eslint/config, e depois responder as perguntas geradas. Digitando npx eslint ./nome_pasta - -fix, ele aplica a padronização na pasta.
Alt + shift + F = identa automaticamente para a nova configuração de espaços.
.eslintrc.json = arquivo de configurações do ESLint, onde é possível mudar as padronizações do código.

Código:

```
import React from 'react'; //Importa o React para ser usado no código.

type?: tipo //Torna o tipo especificado opcional.

class Nome_Componente extends React.Component //Cria uma classe com um componente
```

```
react. Pra chamar no app.tsx, colocar dentro da div como <Nome_Componente/>.
    {render()
        {return
                (<button>Nome do Botão </button>)}} //Nesse caso, o componente é um
botão.

function Nome_Componente() //Cria uma função com um componente react. Pra chamar no
app.tsx, colocar dentro da div como <Nome_Componente/>.
    {return
        (<button>Nome do Botão </button>)} //Nesse caso, o componente é um botão.

const array = [{ item: "Nome 1" }, { item: "Nome 2" }, { item: "Nome 3"}]; //Cria um
aray renderizado automaticamente.
array.map(lista => <p> {array.item} </p>); //O que for adicionado no array será
colocado, nesse caso, no parágrafo.

class Nome_Componente extends React.Component //Cria uma classe com um componente
react e uma props. Pra chamar no app.tsx, colocar dentro da div como
<Nome_Componente>Nome_Prop<Nome_Componente/>.
    {render()
        {return
               (<button>{this.props.children} </button>)}} //Nesse caso, o componente
props é um botão.

function Nome_Funcao() //Cria uma função com vários componentes react, nesse caso,
botões.
    {return
        (<>
            <button>Botão 1<button/>
            <button>Botão 2<button/>
            <button>Botão 3<button/>
        </>)}

function  Nome_Funcao()  //Cria  uma  função  com  componente  react  que  atualiza
automaticamente caso algo seja inserido no setItens.
    {const [itens, setItens] = useState
        ([{ item: "Nome 1" },
        { item: "Nome 2" },
        { item: "Nome 3"}]);
    return
         (setItens([{ item: "Nome 1" }, { item: "Nome 2" }, { item: "Nome 3"}, { item:
"Nome 4"}]))}
```