* O React JS é uma biblioteca JavaScript para a criação de interfaces no usuário (atua no frontend renderizado diretamente no navegador do cliente, ou seja, não é processado no backend). É uma junção de HTML, CSS e JS

Framework x Biblioteca

* **Biblioteca**: compartilha soluções por meio de **funções ou métodos** para facilitar algum ponto do código/desenvolvimento. Utilizando uma biblioteca, não é necessário fazer tudo do zero, tendo em vista que podemos utilizar recursos já criados

Exemplos de bibliotecas:

- Moment.js (datas)

- Chart.js (gráficos)

- Voca

- mo.js

- **React** (podemos utilizar somente as funcionalidades que precisamos)

* **Framework**: ferramenta que ajuda a ter como único objetivo focar em desenvolver o projeto, não em detalhes de configurações. Tem sua documentação própria e alguns **padrões de desenvolvimento que devem ser seguidos**

Exemplo de frameworks:

- Angular

- Vuex

- Ionic

- Next

- Express

- LoopBack

Criando Projeto React

* O **npx** é um pacote do npm que sempre irá baixar a versão mais recente do repositório que você estiver fazendo o download
* Para criar um projeto em React, usamos o seguinte comando no terminal: **npx create-react-app nome-do-projeto** (esse comando faz todas as instalações e baixa todas as dependências necessárias para a criação do projeto)
* O React trabalha com a estrutura PWA (renderização do lado do cliente)
* Cada página da nossa aplicação é um **componente**, então conforme vamos navegando no nosso site, ele renderiza o componente específico da página que a gente quer
* Um **componente** é uma **função retornando um HTML**
* O HTML dentro de uma função chama-se **jsx (escrever tags HTML dentro de um arquivo JS)**
* Um componente deve ser sempre criado com **letra maiúscula**
* Para rodar o projeto no React, digitamos no terminal: **npm run start** ou **yarn run start**

Componentes de Classes x Funcionais

* Componente Funcional: normalmente se tem a função e ela exportada para quem for importar seu componente
* Componente de classe: eram importados o React e o Component dentro do React e era criada uma **class** export default para exportar por padrão
* Todo componente lida com **estado (state)**
* Hoje em dia os componentes de classes não são mais usados
* Não possível retornar duas tags HTML dentro do return de uma function, pois o jsx só retorna um componente

Para fazer isso, devemos colocar as tags dentro de uma tag vazia (<> ... </>):

function App() {

return (**<>**

<div className="App">Olá React</div>

<button></button>

**</>**

);

}

* É obrigatório fechar todas as tags no React, independente se ela tem conteúdo ou não
* Todo o projeto é colocado dentro da pasta **src**
* Ao criar um arquivo com o nome **index**, a importação desse arquivo é facilitada, pois temos que citar somente o repositório em que ele se encontra e não o nome do arquivo, já que o editor busca arquivos com esse nome
* Digitar **rfc** (React Functional Components) cria um modelo de componente automaticamente no VSCode, porém essa estrutura importa o React por padrão, o que já não é mais necessário a partir da versão 17 da lib
* Para exibir uma variável ou qualquer código JS no React, colocamos a informação dentro de **chaves ({ }) quando estamos dentro do return da function**
* Para fazer uma importação, devemos citar o componente que desejamos importar e sua origem (caminho) que deve estar **entre aspas**

Exemplo:

import Button from './components/Button'

Onde:

- Button: componente

- './components/Button': origem

Ciclo de Vida

1. Constructor: passo em que é iniciada a construção do componente, onde será levantado tudo que será necessário para construí-lo
2. Antes desse componente ser mostrado por usuário, ele passa por um processo de montagem dele, se é necessário fazer alguma ação antes dele ser montado e mostrado ao usuário
3. Passa por um processo de saber quando ele precisar ser **atualizado** (toda que vez que o estado de um componente é atualizado, ele é rerenderizado juntos com seus **filhos**)
4. Componente sai da tela e “**morre**” (é possível manipular o momento em que isso aconteça)