ReactJS: framework de JavaScript para fazer front-end de sites web. Baseado em componentes, apenas atualiza os componentes.

JSX: extensão sintaxe de JavaScript, simula HTML.

Hooks ReactJS: gerenciadores de estado de um componente, onde tem uma variável que precisa ser atualizada e mostrada na tela. EX: useState (Primeira variável: valor em si. Segunda variável: função para atualizar a primeira variável.), useEffect (primeiro parâmetro: função chamada, segundo campo é um array, quando o array sofrer alteração a função é chamada).

Função Padrão:

function nome(){

Return ‘isso é uma função js’

}

Imprimir no prompt: console.log(nome())

No prompt: node nome-do-arquivo

Arrow Function: escopo do this é diferente, mais abrangente

Const arrow = () => {

Return ‘isso é uma função js’

}

Console.log(arrow())

Função map:

Cont arrayMap = [10,5,8,56]

Const arrayModificado = Array.map((item,index,array)=>{

Return item+2

})

Console.log(arrayModificado)

Função filter: pode retornar array de mesmo tamanho ou tamanho inferior.

Const arrayModificado = arrayMap.filter((item,index,array)=>{

Return item < 100

})

... :

Const obj = {nome: ‘João’, idade: 22}

Const newObj = {..., p1: 5} adiciona um atributo (p1: 5) ao objeto

Obs: Funciona com array também

Desestruturação:

Const array = [1,5,6]

Const [a,b,c] = array

Ao invés de:

Const a = [1]

Const b = [5]

Const obj = {nome: ‘João’, idade: 22}

Const {nome,idade} = obj

RESTful API:

API: é uma interface de programação de aplicação que define as regras que você precisa seguir para se comunicar com outro sistema de software. Comunicação com as aplicações em formas de funções. EX: página Web e banco de dados.

O cliente envia solicitação para o servidor, o servidor autentica o cliente e confirma que o cliente tem o direito de fazer solicitação. O servidor recebe a solicitação e a processa internamente. O servidor retorna uma resposta ao cliente.

Métodos de requisição:

Método HTTP: informa ao servidor o que ele precisa fazer com o recurso.

GET: os clientes usam para acessar recursos utilizados no URL especificado no servidor. Acessa site e passa informação ao site por meio da URL.

POST: enviar dados ao servidor, incluem a representação de dados com a solicitação. Manda dados dentro da requisição.

PUT: atualizar recursos existentes no servidor.

DELETE: remover recurso.

REST: Estado de transferência representacional. Fácil de identificar: onde na URL, o que no método, mensagens auto-descritivas.

NodeJS: Node é um runtime que executa JS fora do navegador. A ideia era executar operações de rede fora de um web browser.

Express: Framework de Node.Js, para back-end, focado em construir RESTful APIs, usado em Web e Mobile.