**COMO UTILIZAR USE CONTEXT**

Compartilhar o estado globalmente entre as páginas **Menu** (onde você adiciona, edita e exclui itens) e **Delivery** (onde você vê os itens disponíveis e escolhe para o pedido).

**Estrutura do projeto**:

├── App.js

├── Store.js ← (Aqui criamos o Context)

├── Menu.js ← (Página onde gerencia o cardápio)

├── Delivery.js ← (Página onde os itens do cardápio são listados)

**1- Criando o Contexto Global (Store.js)**

Aqui criamos o **Context API** para armazenar os itens do menu e permitir que outras páginas acessem e modifiquem os dados.

import { createContext, useContext, useState } from "react";

// Criando o contexto

const MenuContext = createContext();

// Criando o Provider que envolverá toda a aplicação

export function MenuProvider({ children }) {

const [menu, setMenu] = useState([

{ id: 1, name: "Hambúrguer", price: 15 },

{ id: 2, name: "Pizza", price: 30 },

{ id: 3, name: "Suco", price: 8 },

]);

// Função para adicionar novo item ao menu

const addItem = (newItem) => setMenu([...menu, newItem]);

// Função para remover item do menu

const removeItem = (id) => setMenu(menu.filter((item) => item.id !== id));

return (

<MenuContext.Provider value={{ menu, addItem, removeItem }}>

{children}

</MenuContext.Provider>

);

}

// Hook para facilitar o uso do contexto

export function useMenu() {

return useContext(MenuContext);

}

**2- Envolvendo a aplicação com o MenuProvider (App.js) -** Aqui garantimos que todas as páginas tenham acesso ao menu global.

import React from "react";

import { MenuProvider } from "./Store";

import Menu from "./Menu";

import Delivery from "./Delivery";

function App() {

return (

<MenuProvider>

<h1>Restaurante</h1>

<Menu />

<Delivery />

</MenuProvider>

);

}

export default App;

**3- Página do Menu (Menu.js) -** Aqui você pode adicionar ou remover itens do cardápio.

import React, { useState } from "react";

import { useMenu } from "./Store";

function Menu() {

const { menu, addItem, removeItem } = useMenu();

const [newItem, setNewItem] = useState("");

const handleAddItem = () => {

if (newItem.trim() !== "") {

addItem({ id: Date.now(), name: newItem, price: Math.floor(Math.random() \* 50) + 10 });

setNewItem("");

}

};

return (

<div>

<h2>Cardápio</h2>

<ul>

{menu.map((item) => (

<li key={item.id}>

{item.name} - R$ {item.price}{" "}

<button onClick={() => removeItem(item.id)}>Remover</button>

</li>

))}

</ul>

<input

type="text"

placeholder="Novo item"

value={newItem}

onChange={(e) => setNewItem(e.target.value)}

/>

<button onClick={handleAddItem}>Adicionar</button>

</div>

);

}

export default Menu;

**4- Página de Delivery (Delivery.js) -** Aqui os itens do menu são exibidos e podem ser adicionados ao pedido.

import React, { useState } from "react";

import { useMenu } from "./Store";

function Delivery() {

const { menu } = useMenu();

const [pedido, setPedido] = useState([]);

const adicionarAoPedido = (item) => {

setPedido([...pedido, item]);

};

return (

<div>

<h2>Delivery</h2>

<h3>Escolha seu pedido:</h3>

<ul>

{menu.map((item) => (

<li key={item.id}>

{item.name} - R$ {item.price}{" "}

<button onClick={() => adicionarAoPedido(item)}>Adicionar ao Pedido</button>

</li>

))}

</ul>

<h3>Seu Pedido:</h3>

<ul>

{pedido.map((item, index) => (

<li key={index}>{item.name} - R$ {item.price}</li>

))}

</ul>

</div>

);

}

export default Delivery;

**✅ Como funciona esse código?**

1. O **MenuProvider** mantém o estado global dos itens do cardápio (menu).
2. A página **Menu.js** usa o contexto para **adicionar e remover itens do cardápio**.
3. A página **Delivery.js** acessa o menu global e **permite escolher itens para o pedido**.
4. Qualquer alteração feita no **Menu** (adicionar ou remover) é refletida automaticamente no **Delivery**.

Isso garante que o cardápio e o pedido estejam sempre sincronizados.