**CARRITO DE COMPRAS COMPLETO BARKERY**

Carpeta src + Carpeta actions +

**Archivo Actions.js**

export const TYPES = {

    ADD\_TO\_CART: "ADD\_TO\_CART",

    REMOVE\_ONE\_ITEM: "REMOVE\_ONE\_ITEM",

    REMOVE\_ALL\_ITEMS: "REMOVE\_ALL\_ITEMS",

    CART\_CLEAR: "CART\_CLEAR",

}

Carpeta src + carpeta components + carpeta products +

**Archivo CartItem.js**

const CartItem = ({item, deleteFromCart}) => {

    const {name, price, id, quantity} = item;

  return (

    <div className='cart-item'>

        <h4>{name}</h4>

        <h5>${price} x {quantity} = {price \* quantity}</h5>

        <button onClick={() => deleteFromCart (id,)}>Eliminar uno</button>

        <button onClick={() => deleteFromCart (id, true)}>Eliminar todos</button>

        </div>

  );

};

export default CartItem

**Archivo Product.js**

const Product = ({product, addToCart}) => {

    const {name, price, id} = product;

      return (

        <div className='product'>

            <h4>{name}</h4>

            <h5>${price}</h5>

            <button onClick={() => addToCart(id)}>Agregar</button>

        </div>

      )

    }

    export default Product

**Archivo ShoppingCart.js**

"use client";

import { useReducer, useState, useEffect } from 'react';

import { shoppingReducer } from '@/reducer/ShoppingReducer';

import Product from './Product';

import CartItem from './CartItem';

import ShoppingModal from '../ShoppingModal';

import { TYPES } from '@/actions/actions';

import  HeaderMain from '../HeaderMain';

import axios from 'axios';

const { ADD\_TO\_CART, REMOVE\_ONE\_ITEM, REMOVE\_ALL\_ITEMS, CART\_CLEAR } = TYPES;

const ShoppingCart = ({ setCartCount }) => {

  const [state, dispatch] = useReducer(shoppingReducer, { products: [], cart: [] });

  const { products, cart } = state;

  const [isModalOpen, setIsModalOpen] = useState(false);

  const [modalAction, setModalAction] = useState(null);

  const [selectedProductId, setSelectedProductId] = useState(null);

  useEffect(() => {

    const fetchProducts = async () => {

      try {

      const response = await axios.get('http://localhost:5000/products');

        dispatch({ type: 'SET\_PRODUCTS', payload: response.data });

      } catch (error) {

        console.error('Error al cargar productos:', error);

      }

    };

    fetchProducts();

  }, []);

  const openModal = (action, productId = null) => {

    setModalAction(action);

    setSelectedProductId(productId);

    setIsModalOpen(true);

  };

  const addToCart = (id) => {

    openModal('add', id);

  };

  const deleteFromCart = (id, all = false) => {

    if (all) {

      dispatch({ type: REMOVE\_ALL\_ITEMS, payload: id });

      setCartCount((prevCount) => prevCount - 1);

    } else {

      dispatch({ type: REMOVE\_ONE\_ITEM, payload: id });

    }

  };

  const clearCart = () => dispatch({ type: CART\_CLEAR });

  const confirmAction = async () => {

    setIsModalOpen(false);

    if (modalAction === 'add' && selectedProductId) {

      dispatch({ type: ADD\_TO\_CART, payload: selectedProductId });

      setCartCount((prevCount) => prevCount + 1);

    } else if (modalAction === 'purchase') {

      try {

        const response = await axios.post('/api/confirm-purchase', { cart });

        if (response.status === 200) {

          alert('Compra confirmada');

          clearCart();

          setCartCount(0);

        }

      } catch (error) {

        console.error('Error confirmando la compra:', error);

      }

    }

  };

  const totalProducts = cart.reduce((total, item) => total + item.quantity, 0);

  useEffect(() => {

    setCartCount(totalProducts);

  }, [cart]);

  return (

    <>

      <h2>Carrito de Compras</h2>

      <h3>Productos</h3>

      <div className="box grid-responsive">

        {products.map((product) => (

          <Product key={product.id} product={product} addToCart={addToCart} />

        ))}

      </div>

      <h3>Carrito</h3>

      <div className="box">

        {cart.map((item, i) => (

          <CartItem key={i} item={item} deleteFromCart={deleteFromCart} />

        ))}

      </div>

      <button onClick={() => openModal('purchase')}>Comprar</button>

      <button onClick={clearCart}>Limpiar Carrito</button>

      <ShoppingModal

        isOpen={isModalOpen}

        onClose={() => setIsModalOpen(false)}

        onConfirm={confirmAction}

      />

    </>

  );

}

export default ShoppingCart;

**Archivo HeaderMain.tsx** (con modificaciones)

import React from 'react';

import { BiUser } from 'react-icons/bi';

import { BsSearch } from 'react-icons/bs';

import { HiOutlineShoppingBag } from 'react-icons/hi';

import { FaHeart } from "react-icons/fa6";

import Image from 'next/image';

const HeaderMain = **({ cartCount }: { cartCount: number })** => {

  return (

    <div className='border-b border-red-400 py-5 bg-red-400  pb-5 pt-5'>

      <div className='container sm:flex justify-between items-center my-0'>

        <Image src="/barkery-logo.jpg"

        width={90}

        height={70}

        alt="Barkery Logo"

        />

        <div className='font-bold text-2xl text-center pb-4 sm:pb-0 text-white'>

        </div>

        <div className='w-full sm:w-[300px] md:w-[70%] relative'>

          <input className='border-gray-300 border p-2 px-4 rounded-lg w-full' type='text' placeholder='Buscar productos...' />

          <BsSearch className='absolute right-0 top-0 mr-3 mt-3 text-gray-500' size={20}/>

        </div>

        <div className='hidden lg:flex gap-4 text-blackish text-[30px]'>

          <BiUser/>

          <div className='relative'>

            <FaHeart/>

          <div className='bg-red-600 rounded-full absolute top-0 right-0 w-[18px] h-[18px] text-[12px] text-white grid place-items-center translate-x-1 -translate-y-1'>0</div>

        </div>

        <div className='relative'>

            <HiOutlineShoppingBag/>

          <div className='bg-red-600 rounded-full absolute top-0 right-0 w-[18px] h-[18px] text-[12px] text-white grid place-items-center translate-x-1 -translate-y-1 '>0

          <span className="absolute top-0 right-0 bg-red-500 text-white text-xs rounded-full px-2 py-1">

            {cartCount}

          </span>

          </div>

        </div>

        </div>

      </div>

    </div>

  )

}

export default HeaderMain

**Archivo Start.tsx** (con modificaciones)

"use client";

import React, { useState } from 'react';

import HeaderTop from './HeaderTop';

import HeaderMain from './HeaderMain';

import Navbar from './NavBar';

import Hero from './Slider';

import Footer from './Footer';

import ShoppingCart from './products/ShoppingCart';

const Start = () => {

  const [cartCount, setCartCount] = useState(0);

  return (

    <>

      <HeaderTop />

      <HeaderMain cartCount={cartCount} />

      <section className='bg-bgImage bg-repeat bg-cover bg-bottom w-full h-screen'>

        <Navbar />

        <Hero />

        <Footer />

        <ShoppingCart setCartCount={setCartCount}/>

      </section>

    </>

  );

}

export default Start;

Carpeta src + carpeta db

**Archivo db.json** (para visualizar el funcionamiento – hay q reemplazarla por los productos de la página)

{

    "products": [

      { "id": 1, "name": "Producto A", "price": 50 },

      { "id": 2, "name": "Producto B", "price": 100 },

      { "id": 3, "name": "Producto C", "price": 150 },

      { "id": 4, "name": "Producto D", "price": 200 }

    ],

    "cart": []

  }

Carpeta src + carpeta pages + carpeta appi

**Archivo App.js**

import "../../app/carrito.css";

export default function App({ Component, pageProps }) {

  return <Component {...pageProps} />;

}

**Archivo ConfirmCompra.js**

export default function handler(req, res) {

    if (req.method === 'POST') {

     return res.status(200).json({ message: 'Compra confirmada con éxito' });

    }

    res.setHeader('Allow', ['POST']);

    res.status(405).end(`Method ${req.method} Not Allowed`);

  }

Carpeta src + carpeta reducer

**Archivo ShoppingInitialState.js**

export const shoppingInitialState = {

    products: [

       { id: 1, name: 'Producto A', price: 10 },

       { id: 2, name: 'Producto B', price: 50 },

       { id: 3, name: 'Producto C', price: 100 },

       { id: 4, name: 'Producto D', price: 150 },

       { id: 5, name: 'Producto E', price: 200 },

    ],

    cart: [],

};

**Archivo ShoppingReducer.js**

import { TYPES } from '@/actions/actions';

export const shoppingReducer = (state, action) => {

  switch (action.type) {

    case 'SET\_PRODUCTS':

      return { ...state, products: action.payload };

    case TYPES.ADD\_TO\_CART: {

      const product = state.products.find((p) => p.id === action.payload);

      const itemInCart = state.cart.find((item) => item.id === product.id);

      return itemInCart

        ? {

            ...state,

            cart: state.cart.map((item) =>

              item.id === product.id

                ? { ...item, quantity: item.quantity + 1 }

                : item

            ),

          }

        : {

            ...state,

            cart: [...state.cart, { ...product, quantity: 1 }],

          };

    }

    case TYPES.REMOVE\_ONE\_ITEM: {

      const itemToRemove = state.cart.find((item) => item.id === action.payload);

      return itemToRemove.quantity > 1

        ? {

            ...state,

            cart: state.cart.map((item) =>

              item.id === action.payload

                ? { ...item, quantity: item.quantity - 1 }

                : item

            ),

          }

        : {

            ...state,

            cart: state.cart.filter((item) => item.id !== action.payload),

          };

    }

    case TYPES.REMOVE\_ALL\_ITEMS:

      return {

        ...state,

        cart: state.cart.filter((item) => item.id !== action.payload),

      };

    case TYPES.CART\_CLEAR:

      return { ...state, cart: [] };

    default:

      return state;

  }

};

**Arch package.json**

“scripts”

"server": "json-server --watch src/db/db.json --port 5000"

Carpeta app +

**Archivo carrito.css** (lo separe del global.css x las dudas de algún conflicto)

.box {

  padding: 1rem;

  margin: 1rem;

  background-color: aqua;

}

.grid-responsive {

  display: grid;

  grid-template-columns: repeat(auto-fit, minmax (200px, 1fr));

  gap: 1rem;

}

.product {

  padding: 1rem;

  background-color: blueviolet;

  color: white;

}

.cart-item {

  background-color: crimson;

  padding: 1rem;

      color: white;

}

.modal-overlay {

  position: fixed;

  top: 30%;

  left: 30%;

  width: 350px;

  height: 250px;

  background-color: white;

  display: flex;

  justify-content: center;

  align-items: center;

  z-index: 1000;

  border-radius: 20px;

  border-color: black;

}

.modal-content {

  background-color: white;

  padding: 20px;

  border-radius: 20px;

  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

  width: 300px;

  max-width: 80%;

  text-align: center;

}

.modal-buttons {

  display: flex;

  justify-content: space-around;

  margin-top: 20px;

}

.modal-buttons button {

  padding: 10px 20px;

  border: none;

  background-color: #0070f3;

  color: white;

  border-radius: 5px;

  cursor: pointer;

  transition: background-color 0.3s ease;

}

.modal-buttons button:hover {

  background-color: #005bb5;

}