Titles

Project Name – ShoppyGlobe E-commerce Application

Build by – Shashikant Sahu

Batch – 1st March 2025

Table of Contents

Introduction

ShoppyGlobe is a basic yet fully functional e-commerce web application built using modern React development tools and practices. The app allows users to explore a catalog of products, view detailed information, and manage their shopping cart with ease. It integrates real-time data fetching, global state management using Redux Toolkit, and dynamic routing with React Router. With a focus on modularity, performance optimization (lazy loading), and responsive design (Tailwind CSS), ShoppyGlobe serves as both a practical shopping interface and a showcase of core front-end development skills.

Organized File Structure

Creating a well-organized file structure is crucial for readability, maintainability, and collaboration in any web project. My project is already structured well, and here's a breakdown of each component:

Description of Files:

index.html

- The main HTML file used by Vite.
- It contains a where your React app is injected.
- You typically don't edit this much.

---

main.jsx

- Purpose: Entry point of the React app and app routes are defined
- Usage: Wraps the entire application with Redux Provider and BrowserRouter to enable global state management and routing also React Router to set up paths for Cart, ProductList, ProductDetails and NotFound. Also wraps routes with Suspense for lazy loading.

App.jsx

- Purpose: Main component where defined.
- Usage: Main app to visible the component on UI.

---

components/Header.jsx

- Purpose: Navigation bar component.
- Usage: Displays links to ProductList and Cart, and shows the total number of cart items using Redux state.

---

components/ProductItem.jsx

- Purpose: Displays individual product info in a card layout.
- Usage: Used by ProductList to render each product with title, price, image, and an "Add to Cart" button.

---

components/CartItem.jsx

- Purpose: Displays an individual item inside the cart.
- Usage: Handles quantity change and removal logic using Redux actions.

---

components /ProductList.jsx *(or inside Home)*

- Purpose: Displays a searchable and filterable list of products.
- Usage: Fetches data from API using a custom hook and renders multiple ProductItem components.

---

components /ProductDetails.jsx

- Purpose: Shows detailed information of a single product.
- Usage: Fetches product data using route parameters and allows adding to cart.

---

components /Cart.jsx

- Purpose: Displays all products added to the cart.
- Usage: Uses Redux to manage cart state, supports quantity updates, item removal, and cart total calculation.

---

components /NotFound.jsx

- Purpose: Catch-all route for undefined paths.
- Usage: Improves UX by handling 404 errors gracefully.

---

redux/cartSlice.js

- Purpose: Redux slice to manage cart state.
- Usage: Contains reducers like addToCart, removeFromCart, updateQuantity, and clearCart.

---

redux/store.js

- Purpose: Configures Redux store.
- Usage: Registers cartSlice and provides it to the app via Provider.

---

hooks/useFetchProducts.js

- Purpose: Custom hook for fetching product data.
- Usage: Used inside ProductList to fetch from API on mount and handle loading/error states.

package.json

- Lists project dependencies like react, vite, and react-icons.
- Also contains scripts like npm run dev to start the app.

---

vite.config.js

- Configuration file for Vite.
- Usually left unchanged unless you need to customize the dev server, alias paths, etc.

README.md

- Documentation file explaining:
  - What the app does
  - How to install and run it.

.

Creativity and Presentation

The **ShoppyGlobe** project demonstrates thoughtful UI/UX design along with the use of modern, performance-optimized technologies. Below is a breakdown of the core technologies used in building and styling the application:

Frontend Technologies

Technology	Purpose
React.js	Library for building component-based, declarative UIs.
Redux Toolkit	Simplified Redux state management for cart functionality.
React Router	Enables client-side routing between product list, cart, and detail pages.
JavaScript (ES6+)	Language used to structure and control the app logic.

Styling & Responsiveness

Technology	Purpose
Tailwind CSS	Utility-first CSS framework used to build responsive, clean UI quickly.
Responsive Design	Ensures layout works on desktop, tablet, and mobile screens.

Performance & Tooling

Technology	Purpose
React.lazy + Suspense	Code splitting for lazy-loading components to improve performance.
Vite	Fast development server and build tool for React.
Custom Hooks	Used for fetching API data (useFetchProducts).

Code

index.html code :-

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" type="image/svg+xml" href="/vite.svg" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>ShoppyGlobe E-commerce Application</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.jsx"></script>
  </body>
</html>
```

main.jsx code :-

```
import React, { lazy, StrictMode, Suspense } from "react";
import { createRoot } from "react-dom/client";
import "./index.css";
import App from "./App.jsx";
import { createBrowserRouter, RouterProvider } from "react-router-dom";
import NotFound from "./component/NotFound.jsx";
import ProductList from "./component/ProductList.jsx";

// this function is used for individual rendering when neeeded
const Cart = React.lazy(() => import("./component/Cart.jsx"));
const ProductDetails = React.lazy(() =>
  import("./component/ProductDetails.jsx")
);

// for routing from one to another component without refreshing the page
const appRouter = createBrowserRouter([
  {
    path: "/",
    element: <App />,
```

```
    children: [
      {
        path: "/",
        element: <ProductList />,
      },
      {
        path: "/product-details/:id",
        element: (
          <Suspense
            fallback={
              <h1 className="text-3xl text-center mt-30 py-30 font-semibold">
                Loading...
              </h1>
            }
          >
            <ProductDetails />
          </Suspense>
        ),
      },
      {
        path: "/cart",
        element: (
          <Suspense
            fallback={
              <h1 className="text-3xl text-center mt-30 py-30 font-semibold">
                Loading...
              </h1>
            }
          >
            <Cart />
          </Suspense>
        ),
      },
    ],
    errorElement: <NotFound />,
  },
]);
```

```
createRoot(document.getElementById("root")).render(
  <StrictMode>
    <RouterProvider router={appRouter} />
  </StrictMode>
);
```

App.jsx Code:-

```
import { Outlet } from "react-router-dom";
import "./App.css";
import Header from "./component/Header";
import { Provider } from "react-redux";
import store from "./utils/store";

function App() {
  return (
    // provider act as bridge between redux and react
    <Provider store={store}>
      <Header />
      {/*<Outlet> component from React Router is used to render the child routes within a parent route  */}
      <Outlet />
    </Provider>
  );
}

export default App;
```

Components/ProductList.jsx Code:-

```
import { useState } from "react";
import useFetchProducts from "../utils/UseFetchProducts";
import ProductItem from "./ProductItem";

function ProductList() {
  const { products, loading, error } = useFetchProducts();
  const [search, setSearch] = useState("");
```

```
  // filtering for search inputs
  const filteredProducts = products.filter((product) =>
    product.title.toLowerCase().includes(search.toLowerCase())
  );
  // console.log(filteredProducts);
  return (
    <>
      <div className="px-1 py-10 mt-10">
        <h1 className="text-center font-bold text-4xl mb-2">Products</h1>
        {/* search input UI  */}
        <div className="text-center">
          <input
            type="text"
            placeholder="search products..."
            value={search}
            onChange={(e) => setSearch(e.target.value)}
            className="border py-1 px-4 w-3/4 mx-5 md:w-1/2 rounded "
          />
        </div>
        <div className="grid md:grid-cols-2 lg:grid-cols-4 my-10 mx-2 md:mx-10 lg:mx-0">
          {filteredProducts.length > 0 ? (
            // returning all the data by using map method
            filteredProducts.map((product) => (
              <ProductItem key={product.id} product={product} />
            ))
          ) : (
            <p className="col-span-full text-center text-gray-600">
              No Products Found !
            </p>
          )}
        </div>
      </div>
    </>
  );
}

export default ProductList;
```

Components/ProductItem.jsx Code:-

```
import { useDispatch } from "react-redux";
import { addToCart } from "../utils/cartSlice";
import { Link } from "react-router-dom";
import { MdFileDownloadDone } from "react-icons/md";


function ProductItem(props) {
  const dispatch = useDispatch();


  // data is fetching by using props from ProductList parent page
  const handleAddToCart = () => {
    dispatch(addToCart(props.product));
  };
  return (
    <>
      {/* individual item to show in ProductList page   */}
        <div className=" m-5 md:m-5 lg:m-2 py-4 px-3 bg-zinc-200 hover:scale-98 transition duration-200 ease-out rounded-xl">
        <Link to={`/product-details/${props.product.id}`}>
          <div className="m-3 flex justify-center items-center">
            <img
              src={props.product.thumbnail}
              alt="product_image"
              className=" h-auto w-[50%] md:w-[80%]  bg-white rounded p-5 my-1"
            />
          </div>
          <div className="px-2 md:px-0">
            <h2 className="text-center ">{props.product.title}</h2>
            <div className="flex justify-between mb-2 text-sm">
              <h2 className=" ">Brand: {props.product.brand}</h2>
              <h3>Stock:{props.product.stock}</h3>
            </div>
            <div className="flex justify-between mb-2 text-sm">
              <h3>Price: ${props.product.price}</h3>
              <h3>Rating: {props.product.rating}</h3>
            </div>
            <p className="text-zinc-700 mb-2 text-sm">
```

```
            {props.product.description.split(" ").slice(0, 13).join(" ")}...
          </p>
        </div>
      </Link>
      <div className="">
        {/* button and icon for add to Cart  */}
        <button
          onClick={handleAddToCart}
          className="flex justify-center items-center gap-1 w-full rounded p-1 bg-zinc-400 hover:bg-zinc-600
font-medium"
        >
          <MdFileDownloadDone /> Add to Cart
        </button>
      </div>
    </div>
  </>
 );
}

export default ProductItem;
```

Components/ProductDetails.jsx Code:-

```
import { useEffect, useState } from "react";
import { IoArrowBack } from "react-icons/io5";
import { Link, useParams } from "react-router-dom";

function ProductDetails() {
 const { id } = useParams(); // it is dynamic parameter for a particular/current URL.
 const [product, setProduct] = useState(null);
 const [loading, setLoading] = useState(true);
 const [error, setError] = useState(null);

 // fetching particular data by using "id" from the external API by using useEffect.
 useEffect(() => {
  // console.log("Fetching product with ID:", id);
  const fetchProduct = async () => {
```

```
    try {
      if (!id) throw new Error("Invalid Product ID");
      const response = await fetch(`https://dummyjson.com/products/${id}`);
      if (!response.ok) throw new Error("Failed to fetch product details");
      const data = await response.json();
      setProduct(data);
    } catch (error) {
      setError(error.message);
    } finally {
      setLoading(false);
    }
  };

  fetchProduct();
}, [id]);

if (loading) return <p className="text-center mt-20">Loading...</p>;
if (error) return <p className="text-center mt-20 text-red-500">{error}</p>;

return (
  <>
    {/* Heading of ProductDetails  */}
    <h1 className="text-center mt-20 m-3 p-1 text-2xl underline ">
      <em>{`Book Details with id: ${id}`}</em>
    </h1>
    {/* ProductDetails Detail  */}
    <div className="my-10 mx-10 md:mx-25 lg:m-10 bg-zinc-200 text-black rounded-xl shadow-xl py-5 px-
2 md:p-6 flex flex-col lg:flex-row">
      <div className=" lg:w-[30%] flex justify-center items-center mb-2">
        <img
          src={product.thumbnail}
          alt={product.title}
          className="bg-white p-5 lg:p-0 w-auto h-72  lg:h-80 object-cover rounded "
        />
      </div>
      <div className="lg:p-10 lg:w-[70%]">
        <h2 className=" md:text-2xl lg:text-3xl font-bold mb-4 text-center">
          {product.title}
```

```
</h2>
<div className="mb-2 flex flex-col md:flex-row md:justify-between">
  <h2 className="">
    <strong>Brand:</strong> {product.brand}
  </h2>
  <h3>
    <strong>Warranty:</strong> {product.warrantyInformation}
  </h3>
</div>
<div className="mb-2 flex flex-col md:flex-row md:justify-between">
  <h2 className="">
    <strong>Shipping:</strong> {product.shippingInformation}
  </h2>
  <h3>
    <strong>ReturnPolicy:</strong> {product.returnPolicy}
  </h3>
</div>
<div className="mb-2 flex justify-between">
  <h2 className="">
    <strong>Price:</strong> ${product.price}
  </h2>
  <h3>
    <strong>In Stock:</strong> {product.stock}
  </h3>
</div>

<h3 className="mb-2">
  <strong>Rating:</strong> {product.rating}
</h3>

<p className="mb-2">
  <strong>Description:</strong> {product.description}
</p>

<div className="mt-10 flex justify-center">
  {/* Link to go back to ProductList  */}
  <Link to="/">
      <button className="bg-zinc-600 hover:bg-zinc-800 text-white px-6 py-2 rounded-md flex items-
```

```
center gap-1">
              {/* React Icon is used to show the icon on UI  */}
              <IoArrowBack /> Back to Home
            </button>
          </Link>
        </div>
      </div>
    </div>
  </>
 );
}
export default ProductDetails;
```

Components/PageNotFoound.jsx Code:-

```
import { useRouteError } from "react-router-dom";

function NotFound() {
 // using react-router-dom hook for handling the error routes of page-not-found
 const error = useRouteError();
 // console.log(error);
 return (
  <>
   <div className="h-screen flex flex-col justify-center items-center gap-5">
    <h1 className="text-red-700 text-3xl md:text-5xl font-bold">Oops!</h1>
    <h1 className="text-xl">
     {/* fetching the data dynamically  */}
     {error.status}--{error.statusText}
    </h1>
    <h2>{error.data}</h2>
    <p>Sorry, the page you're looking for does not exist.</p>
    <button className="bg-blue-500 hover:bg-blue-700 px-4 py-2 rounded-xl font-bold">
     <a href="/"> HomePage</a>
    </button>
   </div>
  </>
 );
```

```
}

export default NotFound;
```

Components/Header.jsx Code:-

```
import { useState } from "react";
import { RiShoppingBag3Fill } from "react-icons/ri";
import { FaCartPlus } from "react-icons/fa6";
import { FaBars, FaTimes } from "react-icons/fa";
import { useSelector } from "react-redux";
import { selectCartItems } from "../utils/cartSlice";
import { Link } from "react-router-dom";

function Header() {
  const [menuOpen, setMenuOpen] = useState(false);
  const cartItems = useSelector(selectCartItems);
  const itemCount = cartItems.reduce((total, item) => total + item.quantity, 0);

  return (
    <>
      <div className="fixed w-full top-0 z-10 bg-zinc-400 text-white font-bold flex justify-between items-center
px-10 py-2">
        {/* Logo */}
        <h1 className="text-xl text-blue-500 text-shadow-lg/50">
          <Link href="/" className="flex items-center justify-center gap-1">
            <RiShoppingBag3Fill className="shadow-lg/70" /> ShoppyGlobe
          </Link>
        </h1>

        {/* Hamburger Menu Icon */}
        <button
          className="md:hidden text-white text-2xl"
          onClick={() => setMenuOpen(!menuOpen)}
        >
          {menuOpen ? <FaTimes /> : <FaBars />}
        </button>
```

```
        {/* Navigation Menu */}
        <nav
          className={` absolute md:relative top-11 md:top-0
          right-0 w-28 md:right-25 bg-zinc-400 md:bg-transparent transition-all
           duration-300 md:flex md:items-center md:gap-10
           ${menuOpen ? "block" : "hidden"}`}
        >
          <ul className="flex flex-col md:flex-row md:gap-10 py-1 px-4">
            <li className="hover:text-blue-700">
              <Link to="/">ProductList</Link>
            </li>
            <li className="hover:text-blue-700">
              <Link to="/cart" className="flex justify-center items-center gap-1">
                <sup>{itemCount}</sup>
                <FaCartPlus /> Cart
              </Link>
            </li>
          </ul>
        </nav>
      </div>
    </>
  );
}

export default Header;
```

Components/Cart.jsx Code:-

```
import { useDispatch, useSelector } from "react-redux";
import { clearCart, selectCartItems } from "../utils/cartSlice";
import { Link } from "react-router-dom";
import CartItem from "./CartItem";

function Cart() {
  const cartItems = useSelector(selectCartItems);
  const dispatch = useDispatch();
```

```
// reduce method is performed to sum the price of items
const total = cartItems.reduce(
  (sum, item) => sum + item.price * item.quantity,
  0
);

const handleClearCart = () => {
  dispatch(clearCart());
};

// if no cart present then this block execute.
if (cartItems.length === 0) {
  return (
    <div className="p-10 h-screen flex flex-col justify-center items-center">
      <h2 className="text-2xl font-bold">Your cart is empty</h2>
      <Link to="/" className="text-blue-500 underline">
        Go shopping
      </Link>
    </div>
  );
}

return (
  <div className="max-w-4xl mx-auto p-6 mt-16">
    <h2 className="text-center text-2xl font-bold mb-4">Your Cart</h2>
    {cartItems.map((item) => (
      // calling cartItem here to display in cart page and item is used as prop in child components
      <CartItem item={item} />
    ))}

    <div className="text-right mt-6">
      <p className="text-xl font-bold mb-2">Total: ${total.toFixed(2)}</p>
      {/* button for clearing the cart at once by clicking on it  */}
      <button
        onClick={handleClearCart}
        className="bg-red-500 text-white mb-10 px-4 py-2 rounded hover:bg-red-700"
      >
        Clear Cart
```

```
        </button>
      </div>
    </div>
  );
}

export default Cart;
```

Components/CartItem.jsx Code:-

```
import { useDispatch } from "react-redux";
import { removeFromCart, updateQuantity } from "../utils/cartSlice";

function CartItem({ item }) {
  const dispatch = useDispatch();

  // a function for changing the quantity dynamically
  const handleQuantityChange = (id, quantity) => {
    if (quantity > 0) {
      dispatch(updateQuantity({ id, quantity }));
    }
  };

  const handleRemove = (id) => {
    dispatch(removeFromCart(id));
  };

  return (
    <>
      {/* individual cart item by using the props(item) from the parent component   */}
      <div
        key={item.id}
        className="flex justify-between items-center mb-4 border-b pb-2"
      >
        <div className="flex items-center gap-4">
          <img
            src={item.thumbnail}
```

```
            alt={item.title}
            className="w-24 h-24 object-cover rounded"
          />
          <div>
            <h3 className="font-semibold md:text-lg">{item.title}</h3>
            <p>
              ${item.price} x {item.quantity}
            </p>
            <input
              type="number"
              min="1"
              value={item.quantity}
              onChange={(e) =>
                handleQuantityChange(item.id, Number(e.target.value))
              }
              className="border px-2 py-1 rounded mt-1 w-16"
            />
          </div>
        </div>
        <button
          onClick={() => handleRemove(item.id)}
          className="text-red-600 hover:underline"
        >
          Remove
        </button>
      </div>
    </>
  );
}

export default CartItem;
```

utils/cartSlice.js Code:-

```
import { createSlice } from "@reduxjs/toolkit";

// these is a slice store
const cartSlice = createSlice({
  name: "cart",
  initialState: {
    cartItems: [],
  },
  reducers: {
    addToCart: (state, action) => {
      const existing = state.cartItems.find(
        (item) => item.id === action.payload.id
      );
      if (existing) {
        existing.quantity += 1;
      } else {
        state.cartItems.push({ ...action.payload, quantity: 1 });
      }
    },
    removeFromCart: (state, action) => {
      state.cartItems = state.cartItems.filter(
        (item) => item.id !== action.payload
      );
    },
    updateQuantity: (state, action) => {
      const { id, quantity } = action.payload;
      const item = state.cartItems.find((item) => item.id === id);
      if (item) {
        item.quantity = quantity;
      }
    },
    clearCart: (state) => {
      state.cartItems = [];
    },
  },
});
```

```
export const { addToCart, removeFromCart, updateQuantity, clearCart } =
  cartSlice.actions;
export const selectCartItems = (state) => state.cart.cartItems;
export default cartSlice.reducer;
```

utils/store.js Code:-

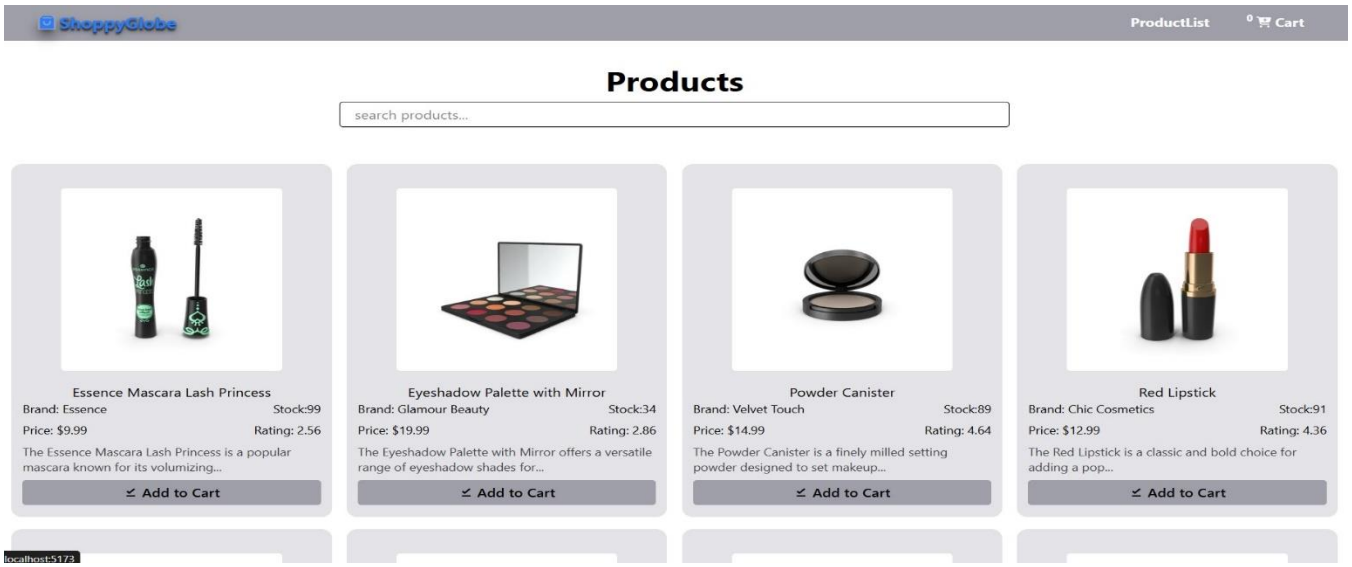
```
import { configureStore } from "@reduxjs/toolkit";
import cartReducer from "./cartSlice.js";

// all the data are stored in a reducer(cartSlice). it is a redux toolkit srote
const store = configureStore({
  reducer: {
    cart: cartReducer,
  },
});
export default store;
```

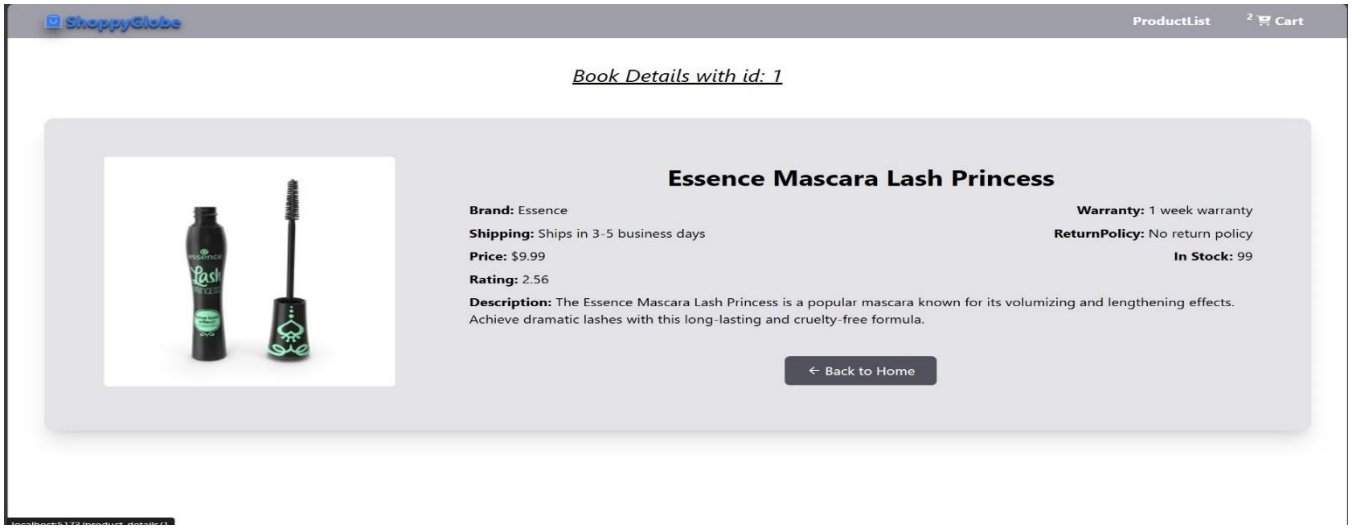
utils/useFetchProducts.js Code:-

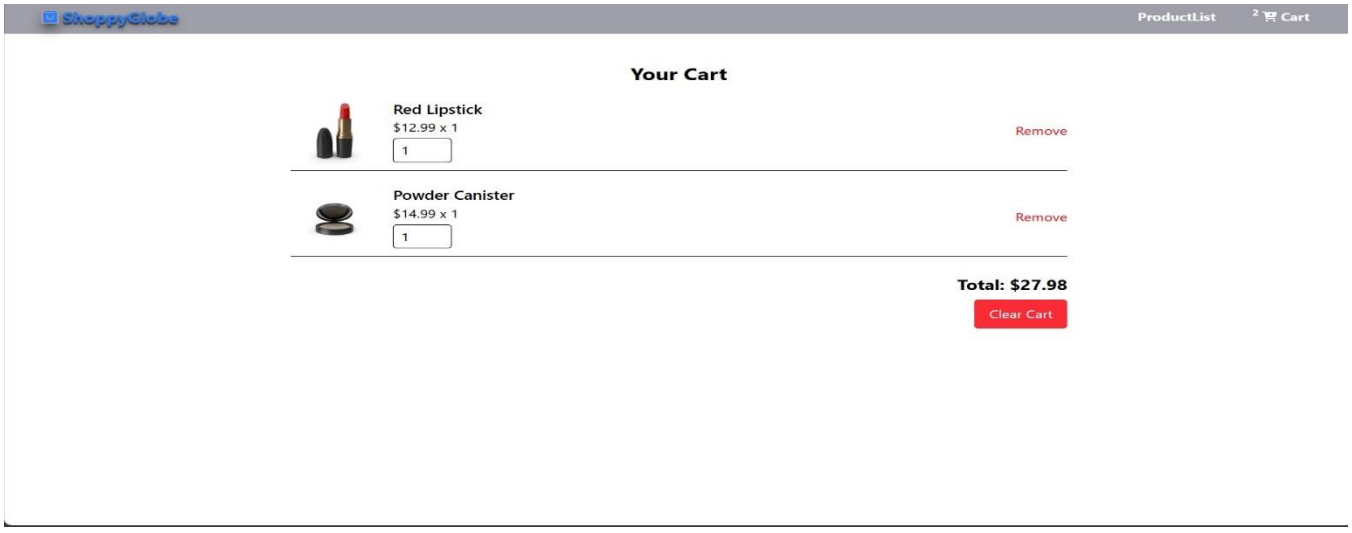
```
import { useEffect, useState } from "react";

function useFetchProducts() {
  const [products, setProducts] = useState([]);
  const [loading, setLoading] = useState(true);
  const [error, setError] = useState(null);

  // fetching the data from external API using useEffect
  useEffect(() => {
    const fetchProducts = async () => {
      try {
        const response = await fetch("https://dummyjson.com/products");
        if (!response.ok) throw new Error("Failed to fetch products");
```

```
        const data = await response.json();
        setProducts(data.products);
      } catch (error) {
        setError(error.message);
      } finally {
        setLoading(false);
      }
    };
    fetchProducts();
  }, []);

  return { products, loading, error };
}
export default useFetchProducts;
```

Screenshot :-


ShoppyGlobe
ProductList
0 Cart
Products
search products...
Essence Mascara Lash Princess
Brand: Essence
Stock:99
Price: $9.99
Rating: 2.56
The Essence Mascara Lash Princess is a popular mascara known for its volumizing...
Add to Cart
Eyeshadow Palette with Mirror
Brand: Glamour Beauty
Stock:34
Price: $19.99
Rating: 2.86
The Eyeshadow Palette with Mirror offers a versatile range of eyeshadow shades for...
Add to Cart
Powder Canister
Brand: Velvet Touch
Stock:89
Price: $14.99
Rating: 4.64
The Powder Canister is a finely milled setting powder designed to set makeup...
Add to Cart
Red Lipstick
Brand: Chic Cosmetics
Stock:91
Price: $12.99
Rating: 4.36
The Red Lipstick is a classic and bold choice for adding a pop...
Add to Cart
localhost:5173


ShoppyGlobe
ProductList
2 Cart
Book Details with id: 1
Essence Mascara Lash Princess
Brand: Essence
Warranty: 1 week warranty
Shipping: Ships in 3-5 business days
ReturnPolicy: No return policy
Price: $9.99
In Stock: 99
Rating: 2.56
Description: The Essence Mascara Lash Princess is a popular mascara known for its volumizing and lengthening effects. Achieve dramatic lashes with this long-lasting and cruelty-free formula.
← Back to Home


ShoppyGlobe
ProductList
2 Cart
Your Cart
Red Lipstick
$12.99 x 1
1
Remove
Powder Canister
$14.99 x 1
1
Remove
Total: $27.98
Clear Cart

- **GitHub Link:-** **https://github.com/shashikant021/ShoppyGlobe**