**IP Exp 8**

**Shashwat Tripathi  
D15A 64**

**Batch C**

**Aim:** The Basics of React.(React features, React application building, components, props, state, etc.)

**Router:**

The Router is a component in React (commonly provided by libraries like React Router) that enables navigation and routing within a single-page application. It allows you to define different "routes" for your application, each corresponding to a unique URL. When the URL changes, the router renders the appropriate component. This is essential for creating multi-page experiences within a single-page application.

**Hooks:**

Introduced in React 16.8, Hooks are functions that allow you to "hook into" React state and lifecycle features in functional components. They provide a way to manage state, side effects, and more, without writing class components. Popular Hooks include useState for managing component state and useEffect for handling side effects like data fetching and subscriptions.

**Ref:**

Refs provide a way to access and interact with the DOM (or other React elements) directly. They are primarily used for accessing or modifying the properties of a rendered component. Refs are essential when working with non-React libraries or for handling focus and media playback.

**Context API:**

The Context API is a built-in part of React that allows you to share data between components without having to pass props manually through each level of the component tree. It's particularly useful for global state management, theming, and localization.

**Redux:**

Redux is a popular state management library for React applications. It enforces a predictable and centralized state management approach. Redux provides a store to hold the application's state and allows components to dispatch actions to modify that state. It's commonly used for complex applications with shared state and a need for time-travel debugging.

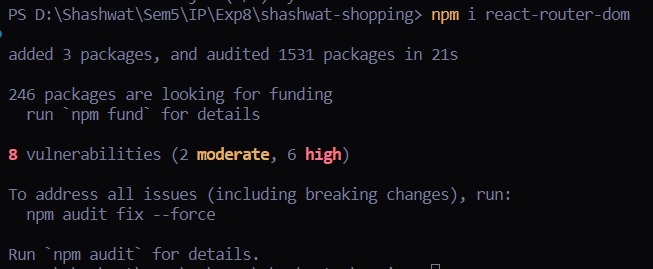
**Higher-Order Components (HOCs):**

HOCs are functions that accept a component and return a new component with additional props and behavior. They are a way to reuse component logic. HOCs can be used for features like authentication, logging, and access control, without repeating the same code in multiple places.

**Error Boundary:**

Error Boundaries are special React components that catch JavaScript errors during rendering or in lifecycle methods of child components and display a fallback UI instead of crashing the whole application. They are useful for improving the robustness of your application by gracefully handling errors.

**Implementation:**



import React, { useState, useRef } from "react";

import { BrowserRouter as Router, Routes, Route, Link } from "react-router-dom";

function App() {

// State to manage the shopping cart

const [cart, setCart] = useState([]);

// Ref for input field

const quantityInputRef = useRef();

// Function to add an item to the cart

const addItemToCart = (item) => {

setCart([...cart, item]);

quantityInputRef.current.value = ""; // Clear the quantity input

};

return (

<Router>

<div className="App" style={appStyle}>

<h1>Shashwat Online Shopping Center</h1>

<nav style={navStyle}>

<ul style={ulStyle}>

<li style={liStyle}>

<Link to="/">Home</Link>

</li>

<li style={liStyle}>

<Link to="/products">Products</Link>

</li>

<li style={liStyle}>

<Link to="/cart">Cart ({cart.length} items)</Link>

</li>

</ul>

</nav>

<Routes>

<Route path="/" element={<Home />} />

<Route

path="/products"

element={

<ProductList

addItemToCart={addItemToCart}

quantityInputRef={quantityInputRef}

/>

}

/>

<Route path="/cart" element={<Cart cart={cart} />} />

</Routes>

</div>

</Router>

);

}

// Home component

function Home() {

return <div>Welcome to Shashwat's online store!</div>;

}

// ProductList component

function ProductList({ addItemToCart, quantityInputRef }) {

const products = [

{ id: 1, name: "Product 1", price: 10 },

{ id: 2, name: "Product 2", price: 20 },

{ id: 3, name: "Product 3", price: 30 },

];

return (

<div>

<h2>Products</h2>

<ul>

{products.map((product) => (

<li key={product.id}>

{product.name} - ${product.price}

<input

type="number"

ref={quantityInputRef}

defaultValue="1"

min="1"

max="10"

style={inputStyle}

/>

<button onClick={() => addItemToCart(product)}>Add to Cart</button>

</li>

))}

</ul>

</div>

);

}

// Cart component

function Cart({ cart }) {

return (

<div>

<h2>Cart</h2>

<ul>

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

<li key={index}>{item.name}</li>

))}

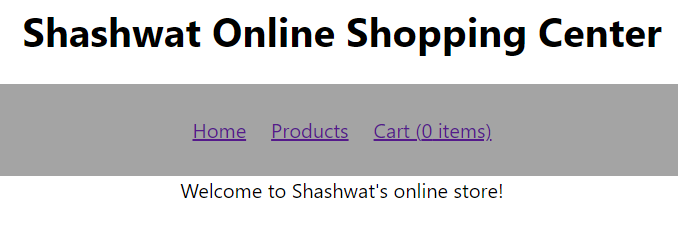
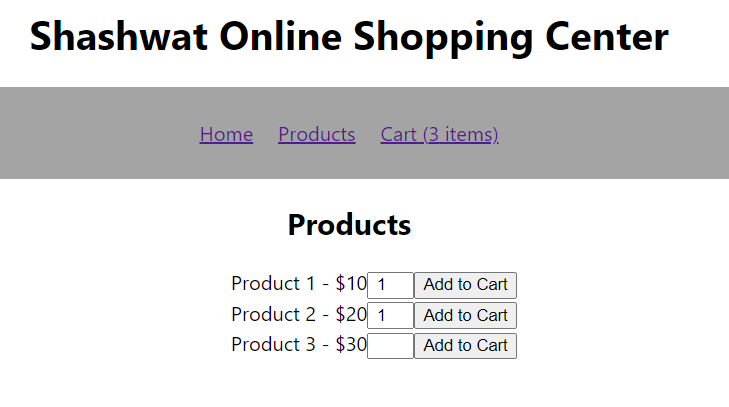
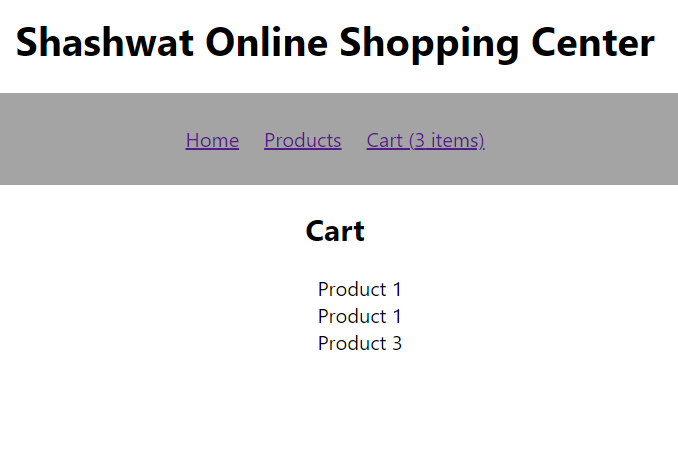
</ul>

</div>

);

}

export default App;



**Conclusion:**

Thus, we have understood the advanced features of JavaScript and implemented that in our programs.