Assignment Week – 5

Routing and Nested Components in React

Name: B. Rithwik

HT No: 2303A52330

Batch: 35

Introduction

Single Page Applications (SPAs) require efficient navigation between different views without reloading the entire page. React Router is a widely used routing library that enables client-side navigation in React applications. In an online shopping application, routing helps users move between pages such as Home, Products, and Cart smoothly. Nested routing further enhances the application by allowing detailed product views within the Products page. This assignment explains the implementation of routing and nested components using React Router.

1. Setup

The React application is created using Create React App. React Router DOM is installed to enable client-side routing. Separate components such as Home, Products, and Cart are created to represent different pages of the Online Shop application.

2. Routing in React Router

Main routes are defined using BrowserRouter, Routes, and Route components. Each route maps a specific URL path to a corresponding component. The Home component is rendered at the root path (/), the Products component at /products, and the Cart component at /cart. This structure enables smooth navigation without page reloads.

3. Nested Routing for Product Details

Nested routing is implemented within the Products route to display individual product details. A dynamic route parameter (:id) is used to identify each product uniquely. The ProductDetail component is rendered as a child route inside the Products component. This approach allows product details to be displayed within the Products page itself.

4. Use of useParams() Hook

The useParams() hook is used inside the ProductDetail component to extract the product ID from the URL. This ID is then used to fetch and display the corresponding product information. Dynamic routing ensures that changing the URL parameter updates the displayed product details.

5. Output

When implemented practically, the application behaves as a Single Page Application with multiple pages. Nested routing works correctly, and product details are displayed dynamically based on URL parameters. Navigation between pages occurs without full page reloads.

Conclusion

Routing and nested components play a crucial role in building scalable React applications. React Router simplifies navigation by enabling client-side routing and dynamic URL handling. By implementing nested routes and using the useParams() hook, the Online Shop application efficiently displays product details and enhances user experience. This approach makes React applications suitable for modern web development.