**Hands-on: 6. ReactJS-HOL**

**Introduction**

React is known for building **single-page applications (SPAs)**, where navigation between different views doesn’t reload the page. This is made possible using **React Router**, a popular library that enables dynamic routing and deep linking in React apps. It allows developers to handle navigation, route management, and URL-based rendering easily and efficiently.

1. **Explain the Need and Benefits of React Router**

* **Need for React Router:**
* SPAs need navigation without full page reloads.
* Manual routing using state is inefficient and hard to manage.
* URL-based navigation improves usability and bookmarkability.
* **Benefits of React Router:**
* Client-side routing without reloading the page
* Dynamic route matching based on parameters
* Nested routing for structured UI rendering
* History management using browser history APIs
* Allows URL parameters and query strings to pass data between pages

1. **Identify the Components in React Router**

React Router provides several key components for implementing routing in your application.

* **Main Components:**

|  |  |
| --- | --- |
| **Component** | **Purpose** |
| BrowserRouter | Sets up the routing context using HTML5 history API |
| Routes | A container for all Route components |
| Route | Defines a path and the component to render |
| Link | Used to navigate between routes without reloading the page |
| NavLink | Like Link, but with styling for active links |
| useNavigate() | Hook to programmatically navigate routes |
| useParams() | Hook to read URL parameters |
| Outlet | Renders nested routes inside parent routes |

1. **List the Types of Router Components**

There are different router types in React Router depending on the environment or navigation strategy:

* **Router Types:**

|  |  |
| --- | --- |
| **Router Type** | **Description** |
| BrowserRouter | Uses the HTML5 history API; suitable for most web apps |
| HashRouter | Uses the URL hash (#) for navigation; good for static file hosting |
| MemoryRouter | Stores history in memory; used mostly for testing and non-browser environments |
| StaticRouter | Used for server-side rendering (SSR) with frameworks like Next.js |

1. **Parameter Passing via URL**

React Router allows passing parameters in the URL to make dynamic routing possible.

* **Step-by-Step Example:**

**1. Define Route with Parameter:**

<Route path="/user/:id" element={<UserProfile />} />

**2. Access the Parameter with useParams():**

import { useParams } from 'react-router-dom';

function UserProfile() {

const { id } = useParams();

return <h2>User ID: {id}</h2>;

}

**3. Linking with Parameter:**

<Link to="/user/123">View User 123</Link>

* **You can also pass query parameters using URLSearchParams:**

<Link to="/search?query=react">Search</Link>

import { useSearchParams } from 'react-router-dom';

const [searchParams] = useSearchParams();

const query = searchParams.get('query');

**Conclusion**

React Router is essential for building navigable and user-friendly single-page applications. It provides a robust and flexible system for handling routing, nested views, and dynamic URLs. By using components like Route, Link, and hooks such as useParams and useNavigate, developers can implement advanced navigation patterns that mimic multi-page apps while still leveraging the speed and simplicity of SPAs.