Report Day: 7.1

Submitted By: Ayesha Abid

Submitted To: Ali Hyder Hidayat

Group: Frontend Development @ProSensia

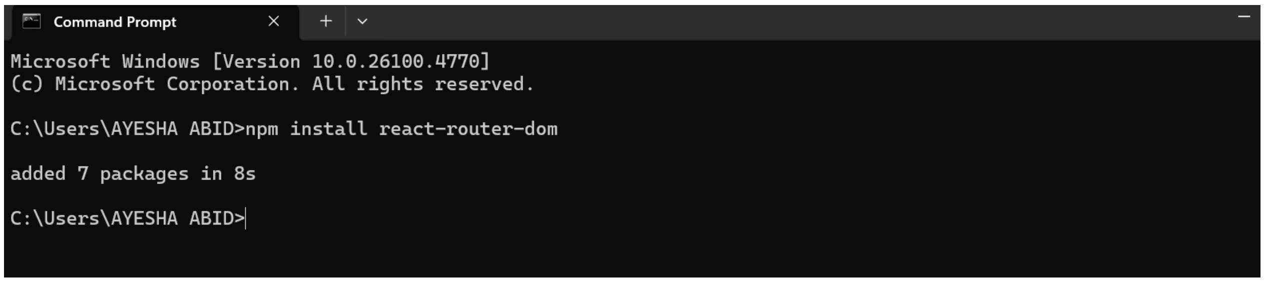
Dated: 04/08/25

Task:

React Router: Pages and Navigation

1. Introduction

React Router is a **standard library for routing in React**. It enables navigation between different components/pages, allows the use of browser URL to determine what is rendered, and manages dynamic routing in React applications.



```
Microsoft Windows [Version 10.0.26100.4770]
(c) Microsoft Corporation. All rights reserved.

C:\Users\AYESHA ABID>npm install react-router-dom

added 7 packages in 8s

C:\Users\AYESHA ABID>
```

2. Core Concepts of React Router

a. Router Types

- <BrowserRouter> – Uses the HTML5 history API to sync UI with the URL.
- <HashRouter> – Uses hash portion of the URL (e.g., /#/home).

b. Pages as Components

Each "page" is essentially a React component that is rendered when the URL matches.

c. Routes and Route Matching

To define routes, use the <Routes> and <Route> components.

3. Navigation in React Router

a. Link Component

<Link> replaces anchor tags for internal navigation.

jsx

```jsx
import { Link } from 'react-router-dom';

<Link to="/">Home</Link>
<Link to="/about">About</Link>
```

b. useNavigate() Hook

Programmatic navigation using the useNavigate hook.

4. Nested Routes

Allows child components to render within a parent route.

jsx

```jsx
<Route path="/dashboard" element={<Dashboard />}>
  <Route path="profile" element={<Profile />} />
</Route>
```

5. Route Parameters

Dynamic segments in the URL like /user/:id.

jsx

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

6. 404 Not Found Page

To handle unmatched routes:

<Route path="*" element={<NotFound />} />

7. Redirects

Use navigate() inside useEffect or conditional logic:

```
if (!userLoggedIn) {
  navigate('/login');
}
```

8. Protected Routes (Private Routing)

Create a wrapper that checks authentication before rendering:

```
function PrivateRoute({ children }) {
  return isAuthenticated ? children : <Navigate to="/login" />;
}
```

9. Lazy Loading Routes

Split code and load routes when needed.

```
import { lazy, Suspense } from 'react';

const Home = lazy(() => import('./Home'));

<Routes>
  <Route path="/" element={
    <Suspense fallback={<div>Loading...</div>}>
      <Home />
    </Suspense>
  } />
</Routes>
```

10. Navigation State & Location

- useLocation() gets current URL info.
- useNavigationType() (v6.4+) tells how navigation occurred (POP, PUSH).

```
import { useLocation } from 'react-router-dom';

const location = useLocation();
console.log(location.pathname); // e.g., "/about"
```

Conclusion

React Router is essential for managing pages and navigation in React apps. It allows you to build seamless, dynamic single-page applications with URL-based routing. By mastering routing, nested routes, programmatic navigation, and route protection, you can create powerful and user-friendly React applications.