Day 15: React Router Basics – Pages & Navigation

1. Why Do We Use React Router?

React is used for building single-page applications (SPA). By default, React apps render only one HTML file (`index.html`). Navigation between pages happens inside the same page without refreshing. To handle this navigation, **React Router** is used.

- **Without React Router**: You would have to manually show/hide components.
- **With React Router**: You can create multiple "pages" with different URLs and navigate smoothly without page reload.

👉Example: `/home`, `/about`, `/contact`

---

2. Installing React Router

To use React Router in your project:

```
npm install react-router-dom
```

---

3. Basic Setup of React Router

Import Required Components

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

Example: Setting Up Pages

```
function Home() {
  return <h1>Home Page</h1>;
}

function About() {
  return <h1>About Page</h1>;
}

function Contact() {
  return <h1>Contact Page</h1>;
}

export default function App() {
  return (
    <BrowserRouter>
      <nav>
        <Link to="/">Home</Link> |
        <Link to="/about">About</Link> |
        <Link to="/contact">Contact</Link>
      </nav>

      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/about" element={<About />} />
        <Route path="/contact" element={<Contact />} />
      </Routes>
    </BrowserRouter>
  );
}
```

👉Here: - `<BrowserRouter>` wraps the app to enable routing. - `<Routes>` holds all the `<Route>` definitions. - `<Route>` maps a **path** (`/about`) to a **component** (`<About />`). - `<Link>` allows navigation without reloading the page.

---

4. Navigation Methods

4.1 Using `<Link>` (Preferred)

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

4.2 Using `useNavigate()` Hook

For programmatic navigation:

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

function Home() {
  const navigate = useNavigate();

  return (
    <button onClick={() => navigate("/about")}>Go to About</button>
  );
}
```

4.3 Redirecting

If you want to redirect after an action (e.g., login):

```
<Route path="/" element={<Navigate to="/home" />} />
```

---

5. Nested Routes

You can create routes inside routes.

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

Now `/dashboard/profile` and `/dashboard/settings` will render inside `Dashboard`.

---

6. When to Use React Router

- When you need **multiple pages** in your SPA.
- When you want **smooth navigation without reloads**.
- When your app grows and has **different views/screens**.

---

7. Day 15 Exercise 📝

Task:

1. Create a React app with **3 pages**: `Home`, `About`, and `Contact`.
2. Add a navigation bar with links to each page using `<Link>`.
3. Use `useNavigate()` in the `Home` page to navigate to `About` when a button is clicked.
4. Bonus: Add a **nested route** inside `About` → `Team` page (URL: `/about/team`).

👉🏼By completing this exercise, you'll understand how to: - Set up React Router. - Create multiple pages. - Navigate using both `<Link>` and `useNavigate()`. - Use nested routes effectively.