**🔧 Day 8: React Routing with react-router-dom**

**✅ What You'll Build:**

A **Desi Multi-Page React App** with:

* ✅ Home Page (with your To-Do app!)
* ✅ About Page (About the project or you)
* ✅ Contact Page (Fake form for fun)
* ✅ Navbar with smooth routing

**🔍 Step-by-Step Plan:**

**🔹 Step 1: Install react-router-dom**

Run this in your project directory:

bash

CopyEdit

npm install react-router-dom

**🔹 Step 2: Create Pages Directory & Files**

Inside your src/, create a folder:

bash

CopyEdit

mkdir src/pages

Now create 3 files:

bash

CopyEdit

touch src/pages/Home.jsx

touch src/pages/About.jsx

touch src/pages/Contact.jsx

**🔹 Step 3: Create Basic Pages**

**📄 Home.jsx**

jsx

CopyEdit

import App from "../App";

export default function Home() {

return <App />;

}

**📄 About.jsx**

jsx

CopyEdit

export default function About() {

return (

<div className="min-h-screen bg-orange-50 flex flex-col items-center justify-center p-10">

<h1 className="text-4xl font-bold text-orange-800 mb-4">About Desi To-Do</h1>

<p className="text-lg text-gray-700 max-w-xl text-center">

Desi To-Do is your friendly task manager built with 💖 using React and Tailwind.

It's simple, colorful, and full of masala 🌶️ — just like life should be!

</p>

</div>

);

}

**📄 Contact.jsx**

jsx

CopyEdit

export default function Contact() {

return (

<div className="min-h-screen bg-yellow-50 flex flex-col items-center justify-center p-10">

<h1 className="text-4xl font-bold text-orange-700 mb-4">📞 Contact Us</h1>

<p className="text-lg text-gray-700">Want to connect? Drop a lassi emoji 🥤 and we’ll get back!</p>

</div>

);

}

**🔹 Step 4: Create Navbar.jsx (Shared Nav Component)**

**📄 Navbar.jsx**

jsx

CopyEdit

import { Link } from "react-router-dom";

export default function Navbar() {

return (

<nav className="bg-orange-100 py-4 shadow-md flex justify-center gap-6 text-orange-800 font-semibold text-lg">

<Link to="/" className="hover:text-orange-600">🏠 Home</Link>

<Link to="/about" className="hover:text-orange-600">📘 About</Link>

<Link to="/contact" className="hover:text-orange-600">📞 Contact</Link>

</nav>

);

}

**🔹 Step 5: Setup Router in main.jsx**

jsx

CopyEdit

import React from "react";

import ReactDOM from "react-dom/client";

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

import Home from "./pages/Home";

import About from "./pages/About";

import Contact from "./pages/Contact";

import Navbar from "./Navbar";

import "./index.css";

ReactDOM.createRoot(document.getElementById("root")).render(

<React.StrictMode>

<BrowserRouter>

<Navbar />

<Routes>

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

<Route path="/about" element={<About />} />

<Route path="/contact" element={<Contact />} />

</Routes>

</BrowserRouter>

</React.StrictMode>

);

**✅ Done! You now have:**

* Navbar working across pages
* Clean routes
* Home = your awesome Desi To-Do App
* About + Contact = extra spice 🌶️

**🌐 Bonus Challenge (Optional):**

Add **active link highlight** using NavLink, or create animated transitions using Framer Motion!

**🧭 What is React Router?**

**React Router** is the official routing library for React. It lets you create **Single Page Applications (SPAs)** with multiple pages, without actually refreshing the browser.

🧠 Think of it as the **Google Maps** of your React App — it helps your app "navigate" to different pages (/about, /contact, etc.) without ever leaving the single HTML file.

**🔧 Core Concepts of React Router**

| **Concept** | **What it does** | **Example** |
| --- | --- | --- |
| BrowserRouter | Wraps your whole app to enable routing | <BrowserRouter>...</BrowserRouter> |
| Routes | A container for all your <Route /> components | <Routes>...</Routes> |
| Route | Defines a path and which component to show | <Route path="/about" element={<About />} /> |
| Link | Used to navigate between routes without reload | <Link to="/about">About</Link> |
| useNavigate() | Programmatic navigation (JS-based routing) | navigate("/contact") |

**🧩 Component Breakdown**

**1. BrowserRouter**

* It **enables routing** using HTML5 history API.
* Put this **only once** around your entire app, typically in main.jsx or index.jsx.

jsx

CopyEdit

<BrowserRouter>

<App />

</BrowserRouter>

**2. Routes and Route**

* Routes: Think of it as a **container** for all your Routes.
* Route: Maps a **URL path** to a **component**.

jsx

CopyEdit

<Routes>

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

<Route path="/about" element={<About />} />

</Routes>

**3. Link**

* Acts like <a href="..."> but without full page reload.
* Used for **client-side navigation**.

jsx

CopyEdit

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

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

**4. useNavigate()**

* A hook to **navigate with JavaScript**.

jsx

CopyEdit

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

const ContactForm = () => {

const navigate = useNavigate();

const handleSubmit = () => {

// After form submission

navigate('/thank-you');

};

};

**🧠 Visual Map**

php-template

CopyEdit

App.jsx

├── <Navbar />

└── <Routes>

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

├── <Route path="/about" element={<About />} />

└── <Route path="/contact" element={<Contact />} />

Clicking a <Link to="/about">About</Link> updates the URL and loads <About /> without refreshing the whole page!

**⚠️ Common Mistakes (That You Faced Too 😉)**

| **Mistake** | **Fix** |
| --- | --- |
| Using Route outside of Routes | Always wrap Routes inside <Routes> |
| Forgetting to import Routes or Route | Always import from 'react-router-dom' |
| Not wrapping the app in BrowserRouter | Wrap your root app with it |
| Writing <Route path="/" component={Home} /> | Use element={<Home />} in React Router v6+ |