



React 18 Routing (Web)

Objective

You'll build a small app with these pages:

- **Home, About, Contact**
 - **SelectItem** and **DispItem** (dynamic route)
 - **Dashboard** (nested routes)
 - **NotFound (404)** page
- and learn to use **React Router v6** step by step.
-



Step 1 – Create the project (Vite + React)

Open your terminal and run:

```
npm create vite@latest app-router -- --template react
cd app-router
npm install
npm install react-router-dom
npm run dev    # start the dev server
```



This sets up a React 18 project using Vite and installs React Router v6.



Step 2 – Folder structure

```
app-router/
├── src/
│   ├── main.jsx
│   ├── App.jsx
│   ├── components/
│   │   ├── NavBar.jsx
│   │   └── DashboardLayout.jsx
│   └── pages/
│       ├── Home.jsx
│       ├── About.jsx
│       ├── Contact.jsx
│       ├── SelectItem.jsx
│       ├── DispItem.jsx
│       └── NotFound.jsx
```

✿ Step 3 – main.jsx (root with BrowserRouter)

```
import React from "react";
import ReactDOM from "react-dom/client";
import { BrowserRouter } from "react-router-dom";
import App from "../App.jsx";

ReactDOM.createRoot(document.getElementById("root")).render(
  <React.StrictMode>
    <BrowserRouter>
      <App />
    </BrowserRouter>
  </React.StrictMode>
);
```

React Router must wrap your app at the root level.

🌀 Step 4 – NavBar.jsx (Navigation links)

```
import { NavLink, Link } from "react-router-dom";

export default function NavBar() {
  const linkStyle = ({ isActive }) => ({
    marginRight: 12,
    textDecoration: isActive ? "underline" : "none"
  });

  return (
    <nav style={{ padding: 12, borderBottom: "1px solid #ddd" }}>
      <NavLink to="/" style={linkStyle}>Home</NavLink>
      <NavLink to="/about" style={linkStyle}>About</NavLink>
      <NavLink to="/contact" style={linkStyle}>Contact</NavLink>
      <Link to="/select">Select Item</Link>
    </nav>
  );
}
```

- ✓ Use <NavLink> for active page highlighting.
 - ✓ <Link> and <NavLink> prevent full page reloads.
-

🏠 Step 5 – Basic pages

Create three simple components:

```
// src/pages/Home.jsx
```

```

export default function Home() {
  return <h2>Home</h2>;
}

// src/pages/About.jsx
export default function About() {
  return <h2>About</h2>;
}

// src/pages/Contact.jsx
export default function Contact() {
  return <h2>Contact</h2>;
}

```



Step 6 – Dynamic Routes: SelectItem & DispItem

SelectItem.jsx

```

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

export default function SelectItem() {
  const items = ["Tomato", "Potato", "Orange"];
  return (
    <section>
      <h2>Select Item</h2>
      <ul>
        {items.map((name) => (
          <li key={name}>
            <Link to={`/item/${name}`}>Display {name}</Link>
          </li>
        ))}
      </ul>
    </section>
  );
}

```

DispItem.jsx


```

import { useParams, useNavigate } from "react-router-dom";

export default function DispItem() {
  const { name } = useParams(); // read dynamic :name
  const navigate = useNavigate(); // for programmatic navigation

  return (
    <section>
      <h2>Display data for: {name}</h2>
      <button onClick={() => navigate("/select")}>Back to Select</button>
    </section>
  );
}

```

 `useParams()` extracts URL parameters,
`useNavigate()` lets you redirect via code.

Step 7 – 404 Page and `App.jsx` (route map)

`NotFound.jsx`

```
export default function NotFound() {  
  return <h2>404 - Page Not Found</h2>;  
}
```

`App.jsx`

```
import { Routes, Route } from "react-router-dom";  
import NavBar from "../components/NavBar";  
import Home from "../pages/Home";  
import About from "../pages/About";  
import Contact from "../pages/Contact";  
import SelectItem from "../pages/SelectItem";  
import DispItem from "../pages/DispItem";  
import NotFound from "../pages/NotFound";  
  
export default function App() {  
  return (  
    <div>  
      <NavBar />  
      <Routes>  
        <Route path="/" element={<Home />} />  
        <Route path="/about" element={<About />} />  
        <Route path="/contact" element={<Contact />} />  
        <Route path="/select" element={<SelectItem />} />  
        <Route path="/item/:name" element={<DispItem />} />  
        <Route path="*" element={<NotFound />} />  
      </Routes>  
    </div>  
  );  
}
```

★ `path="*" catches any unknown route and renders the 404 page.`

Step 8 – Nested Routes & Shared Layout

`DashboardLayout.jsx`

```
import { Link, Outlet } from "react-router-dom";
```

```
export default function DashboardLayout() {
  return (
    <div style={{ display: "grid", gridTemplateColumns: "200px 1fr", gap: 16
  }}>
      <aside>
        <h3>Dashboard</h3>
        <nav>
          <Link to="profile">Profile</Link><br/>
          <Link to="settings">Settings</Link>
        </nav>
      </aside>
      <main>
        <Outlet /> {/* Child route renders here */}
      </main>
    </div>
  );
}
```

Add to App.jsx

```
<Route path="/dashboard" element={<DashboardLayout />}>
  <Route path="profile" element={<h2>Profile</h2>} />
  <Route path="settings" element={<h2>Settings</h2>} />
</Route>
```

✓ <Outlet /> allows **nested routes** to share a common layout.

Step 9 – Programmatic Navigation (useNavigate)

You can navigate from code (e.g., after form submit):

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

export default function About() {
  const navigate = useNavigate();

  const onBack = () => navigate("/");

  return (
    <section>
      <h2>About</h2>
      <button onClick={onBack}>Go Home</button>
    </section>
  );
}
```

Step 10 – Run & Test

Start the app:

```
npm run dev
```

Open your browser and test:

Path	Expected Page
/	Home
/about	About
/contact	Contact
/select	SelectItem (list of items)
/item/Tomato	DispItem showing Tomato
/anythingwrong	404 – Page Not Found



Extras & Tips

- ✓ Use `<NavLink>` for automatic active styles
- ✓ Keep components small and focused
- ✓ Co-locate route components near their features
- ✓ Use semantic, lowercase URLs
- ✓ Add unit tests for important navigation flows