



Routing in React – Full Guide with SPA, Setup, Real Examples & Best Practices

What is Routing?

Routing allows users to **navigate different pages** of your application by changing the URL, without refreshing the whole page (in SPA).

1. Server-side Routing (SSR)

 Concept:

Each time user navigates  new request sent to server  full HTML page returned.

```
<!-- Traditional HTML route -->
<a href="/about">About</a> <!-- Full Page Reload -->
```

✗ Downsides:

- Every link = network request = slow 
 - UI reloads = poor UX
-


2. Client-side Routing (CSR) – Modern React Way

 Concept:

Page is **not refreshed**. React controls the route change using JS.

```
// Modern React route
<Link to="/about">About</Link> // No refresh 
```

 Benefits:





- ⚡ Fast transitions
 -  Better UX
 - ✨ No full-page reload
-

3. SPA vs MPA

Feature	SPA (Single Page App)	MPA (Multi Page App)
---------	-----------------------	----------------------

Feature	SPA (Single Page App)	MPA (Multi Page App)
Pages	One HTML, dynamic updates	Each page = new HTML
Reload	✗ No reload	☑ Reload on each page
Speed	⚡ Super Fast	🐢 Slower
SEO	✗ Not ideal	☑ Great
Routing	JS handles it (CSR)	Server handles it (SSR)

Real-Life SPA Scenarios

App	Why SPA Works?
 Swiggy/Zomato	Fast nav, no reloads
 WhatsApp Web	Chat stays loaded
 Netflix	Seamless video nav
 Amazon Cart Page	Smooth UX without reloads

React Routing – All Types

1. `BrowserRouter` + `<Routes>`

 Best for **simple-to-moderate** apps

```
import { BrowserRouter, Routes, Route } from 'react-router-dom';
import Home from './Home';
import About from './About';

function App() {
  return (
    <BrowserRouter>
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/about" element={<About />} />
      </Routes>
    </BrowserRouter>
  );
}
```

2. `createBrowserRouter` + `RouterProvider`

 **Modern**, declarative, more flexible  Used in **data routers**, better for nested routes

```
import { createBrowserRouter, RouterProvider } from 'react-router-dom';

const router = createBrowserRouter([
  { path: "/", element: <Home /> },
  { path: "/about", element: <About /> },
]);

function App() {
  return <RouterProvider router={router} />;
}
```

3. HashRouter (For old browsers or GitHub Pages)

```
import { HashRouter, Route, Routes } from 'react-router-dom';

function App() {
  return (
    <HashRouter>
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/about" element={<About />} />
      </Routes>
    </HashRouter>
  );
}
```

☒ No need for server config ☒ Ugly URLs (e.g., #/about)

Nested Routes Example

```
const router = createBrowserRouter([
  {
    path: "/",
    element: <Layout />,
    children: [
      { path: "about", element: <About /> },
      { path: "contact", element: <Contact /> },
    ],
  },
]);

function Layout() {
  return (
    <div>
      <Navbar />
      <Outlet /> { /* renders nested routes */ }
    </div>
  );
}
```

```
    </div>
  );
}
```

Components Overview

Component	Purpose
<code>BrowserRouter</code> / <code>HashRouter</code>	Base routing context
<code>Routes</code> & <code>Route</code>	Routing structure (path + element)
<code>Link to="/path"</code>	Navigation link
<code>Outlet</code>	Where nested routes render
<code>createBrowserRouter</code>	Data-router config
<code>RouterProvider</code>	Required with <code>createBrowserRouter</code>

☒ What Should I Use?

Use Case	Prefer
Simple app	<code>BrowserRouter</code>
Nested layout/route setup	<code>createBrowserRouter</code> <input checked="" type="checkbox"/>
Static Hosting (e.g. GitHub Pages)	<code>HashRouter</code> <input checked="" type="checkbox"/>
SEO-heavy site	Consider SSR frameworks like Next.js

Quick Setup (Modern SPA Setup)

1. Install:

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

2. Create Route Components:

```
// pages/About.js
export default function About() {
  return <h1>About Page</h1>;
}
```

3. Create Router:

```
// App.js
import { createBrowserRouter, RouterProvider } from "react-router-dom";
import About from "./pages/About";
import Home from "./pages/Home";

const appRouter = createBrowserRouter([
  { path: "/", element: <Home /> },
  { path: "/about", element: <About /> },
]);

export default function App() {
  return <RouterProvider router={appRouter} />;
}
```

💡 CDN for Images – Real Use

CDNs like Cloudinary, AWS, etc., are used to serve images **super fast** in apps like Swiggy, Flipkart.

☑ Advantages:

- 🌐 Global caching
- 📺 Optimized images
- ⚡ Lightning fast load
- 📈 Increases performance

```

```

📋 Final Comparison Summary

Feature	SSR	CSR (React SPA)
Page Reloads	Yes	No
Navigation	Slow	Fast
SEO	Good	Needs setup
Dev Control	Less	High
Routing Type	Server-handled	Client-handled
Real Use	Blogs, News	Dashboards, Apps

📄 Bonus Tips

- 🔍 Use **React Router v6+** (has many improvements)
- ⚙️ Always wrap your app with a Router (**BrowserRouter** or **RouterProvider**)

- 🧠 Use `useNavigate` instead of `history.push`
 - 🛠 For tests use `MemoryRouter`
-

📦 Packages to Explore

- `react-router-dom`
 - `formik` (for forms)
 - `yup` (for validation schema)
 - `react-hook-form` (faster alternative to Formik)
-