React Router Step-by-Step Guide

React Router is a powerful tool for creating navigable, multi-page applications in React. Below is a step-by-step guide for setting up and using React Router in a React app.

Step-by-Step Guide to React Router

Step 1: Install React Router

First, you need to install `react-router-dom`:

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

Step 2: Set Up Basic App Structure

Let?s create a simple React app structure with some components that will act as different pages.

```
// App.js
import React from 'react';
import { BrowserRouter as Router, Route, Routes, Link } from 'react-router-dom';
import Home from './pages/Home';
import About from './pages/About';
import Contact from './pages/Contact';

function App() {
  return (
    <Router>
      <nav>
```

```
        <ul>
          <li><Link to="/">Home</Link></li>
          <li><Link to="/about">About</Link></li>
          <li><Link to="/contact">Contact</Link></li>
        </ul>
      </nav>

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

export default App;
```

Explanation:

- Router: Wraps the entire application in routing logic.
- Link: Used for navigating between pages without reloading the page (single-page application behavior).
- Routes: Replaces the older Switch and contains all the different routes.
- Route: Defines each route with a specific path and the corresponding element to render.

Step 3: Create Component Pages

Now create the individual components that represent the pages.

```
// pages/Home.js
import React from 'react';

function Home() {
  return (
    <div>
      <h1>Home Page</h1>
      <p>Welcome to the homepage!</p>
    </div>
  );
}

export default Home;

// pages/About.js
import React from 'react';

function About() {
  return (
    <div>
      <h1>About Page</h1>
      <p>Learn more about us here.</p>
    </div>
  );
}
```

```
export default About;


// pages/Contact.js
import React from 'react';


function Contact() {
  return (
    <div>
      <h1>Contact Page</h1>
      <p>Get in touch with us.</p>
    </div>
  );
}


export default Contact;
```

Step 4: Test Navigation

Start your React app and check if navigation works by clicking on the links.

Step 5: Adding Nested Routes

You can also add nested routes, where a page might have sub-pages.

```
// pages/About.js
import React from 'react';
import { Link, Outlet } from 'react-router-dom';
```

```
function About() {
  return (
    <div>
      <h1>About Page</h1>
      <nav>
        <ul>
          <li><Link to="team">Our Team</Link></li>
          <li><Link to="mission">Our Mission</Link></li>
        </ul>
      </nav>
      <Outlet />
    </div>
  );
}

export default About;

// App.js (updated with nested routes)
import React from 'react';
import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';
import Home from './pages/Home';
import About from './pages/About';
import Contact from './pages/Contact';
import Team from './pages/Team';
import Mission from './pages/Mission';

function App() {
  return (
    <Router>
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/about" element={<About />}>
          <Route path="team" element={<Team />} />
          <Route path="mission" element={<Mission />} />
        </Route>
        <Route path="/contact" element={<Contact />} />
      </Routes>
    </Router>
  );
}

export default App;

// pages/Team.js
import React from 'react';

function Team() {
  return (
    <div>
      <h2>Our Team</h2>
      <p>Meet the team that makes everything possible.</p>
    </div>
  );
}
```

```
export default Team;


// pages/Mission.js
import React from 'react';


function Mission() {
  return (
    <div>
      <h2>Our Mission</h2>
      <p>Learn about our goals and values.</p>
    </div>
  );
}


export default Mission;
```

Step 6: Handling 404 Pages

If the user navigates to a page that doesn?t exist, you can show a custom 404 page.

```
// App.js (adding a 404 route)
import React from 'react';
import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';
import Home from './pages/Home';
import About from './pages/About';
import Contact from './pages/Contact';
import NotFound from './pages/NotFound';
```

```
function App() {
  return (
    <Router>
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/about" element={<About />} />
        <Route path="/contact" element={<Contact />} />
        <Route path="*" element={<NotFound />} />
      </Routes>
    </Router>
  );
}

export default App;

// pages/NotFound.js
import React from 'react';

function NotFound() {
  return (
    <div>
      <h1>404 - Page Not Found</h1>
      <p>Sorry, the page you are looking for does not exist.</p>
    </div>
  );
}
```

```
export default NotFound;
```

Step 7: Programmatic Navigation

Sometimes, you might want to navigate to a route programmatically.

```
// pages/Contact.js (example with useNavigate)
import React from 'react';
import { useNavigate } from 'react-router-dom';

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

  const handleSubmit = (e) => {
    e.preventDefault();
    navigate('/thank-you');
  };

  return (
    <div>
      <h1>Contact Page</h1>
      <form onSubmit={handleSubmit}>
        <input type="text" placeholder="Your Name" />
        <button type="submit">Submit</button>
      </form>
    </div>
  );
}


export default Contact;
```

Key Concepts:

- Routes and Nested Routes: Define different pages and sub-pages.
- Link: Replaces anchor tags to navigate without reloading the page.
- useNavigate: Allows programmatic navigation.
- Outlet: Renders nested routes in the correct place.
- 404 Handling: Use a catch-all Route with "*" path to handle non-existent routes.

This should give you a comprehensive start on how to implement and work with React Router!