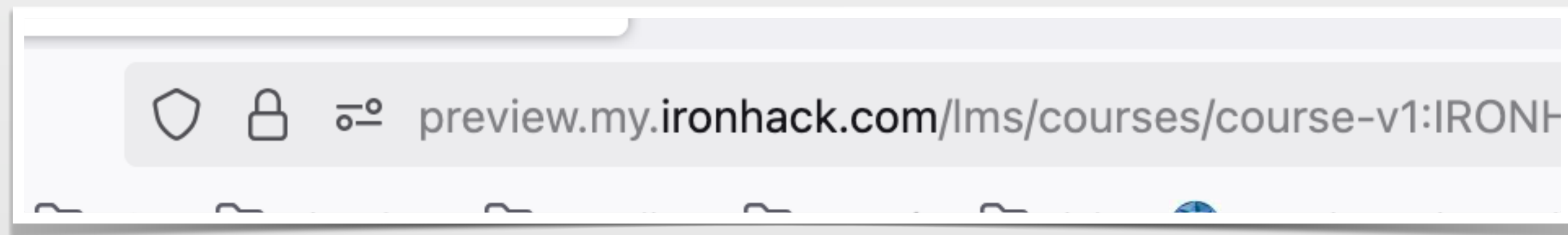# M2W2D3

miguel.garrido@ironhack.com

React Router

React Router is a collection of components.

These components will allow us to handle the navigation.

React Router keeps an eye on the browser URL

preview.my.**ironhack**.com/lms/courses/course-v1:IRONH

React works with the <u>History API</u>.

IRON
HACK

To install React Router

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

IRON
HACK

React Router Components

<BrowserRouter>

This component keeps the User Interface in sync with the URL. It uses the HTML5 History API.

<HashRouter>

Uses the hash portion of the URL. (Only for older browsers that don't support the HTML5 History API)

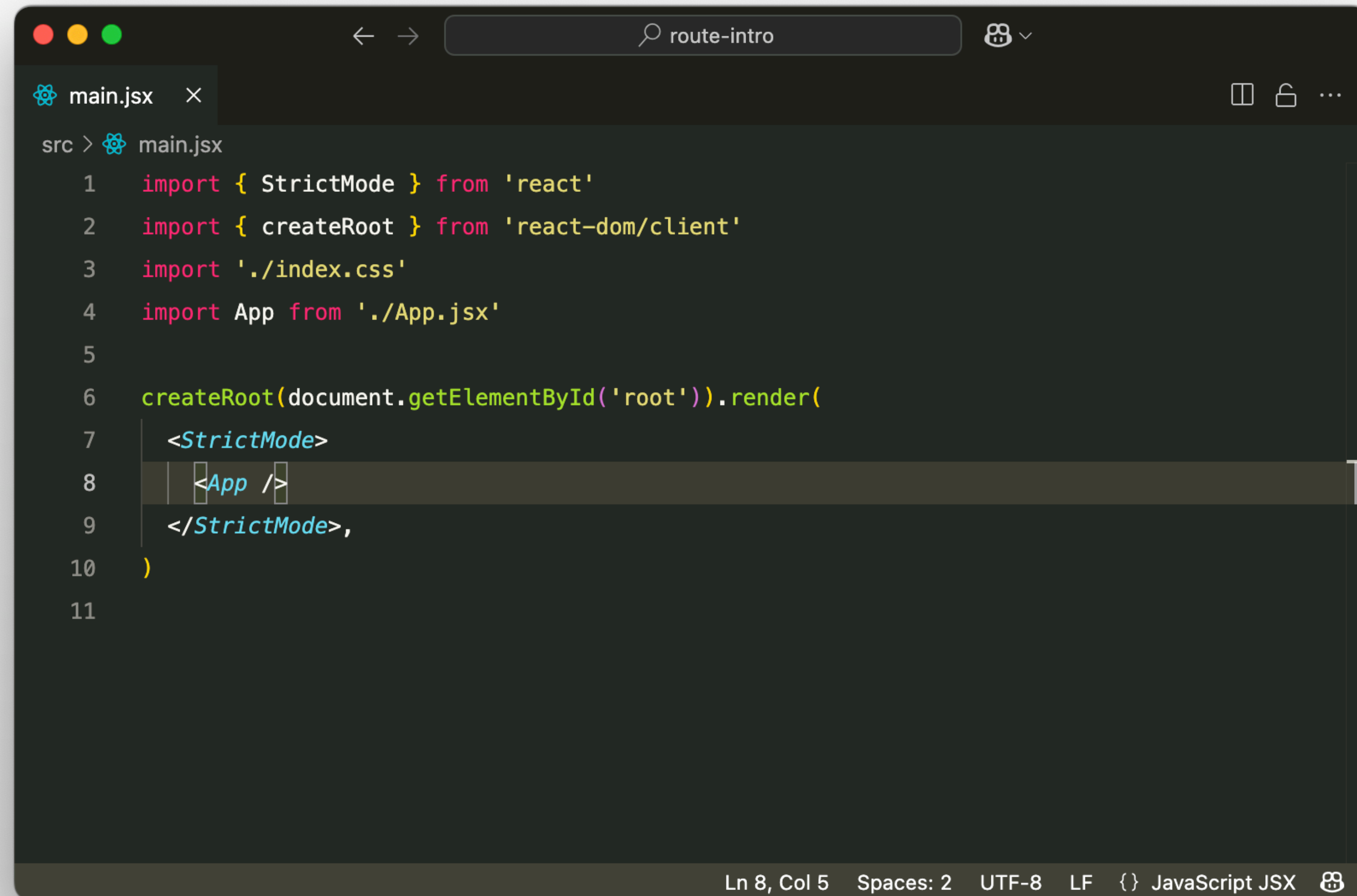**<Routes>** & **<Route>**

Renders a UI component depending on the URL.

<Link>

Renders a navigation link. (basically an <a> tag, but they change the URL without refreshing the page),
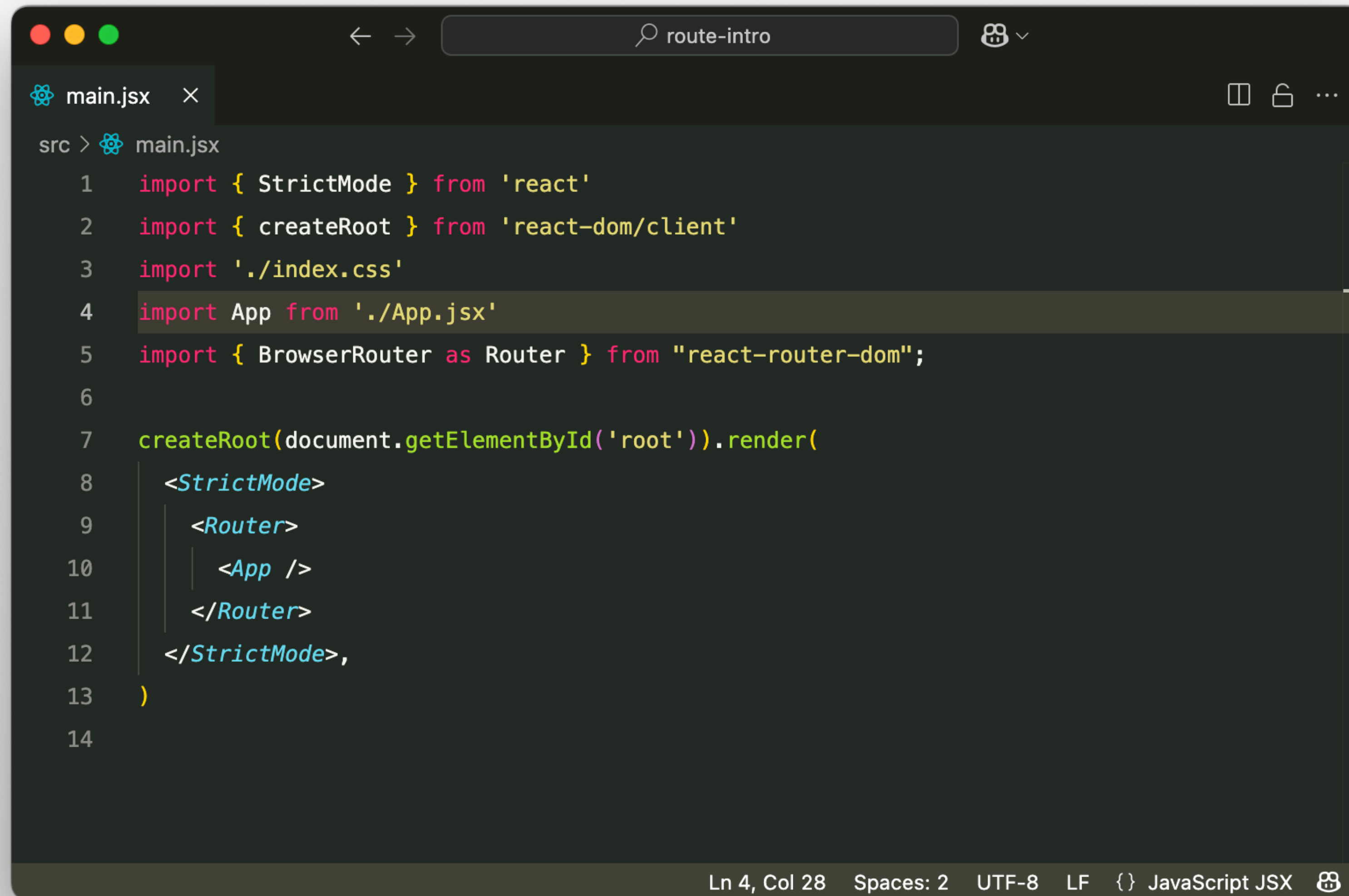
IRON
HACK

main.jxs

```jsx
import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import './index.css'
import App from './App.jsx'

createRoot(document.getElementById('root')).render(
  <StrictMode>
    <App />
  </StrictMode>,
)
```
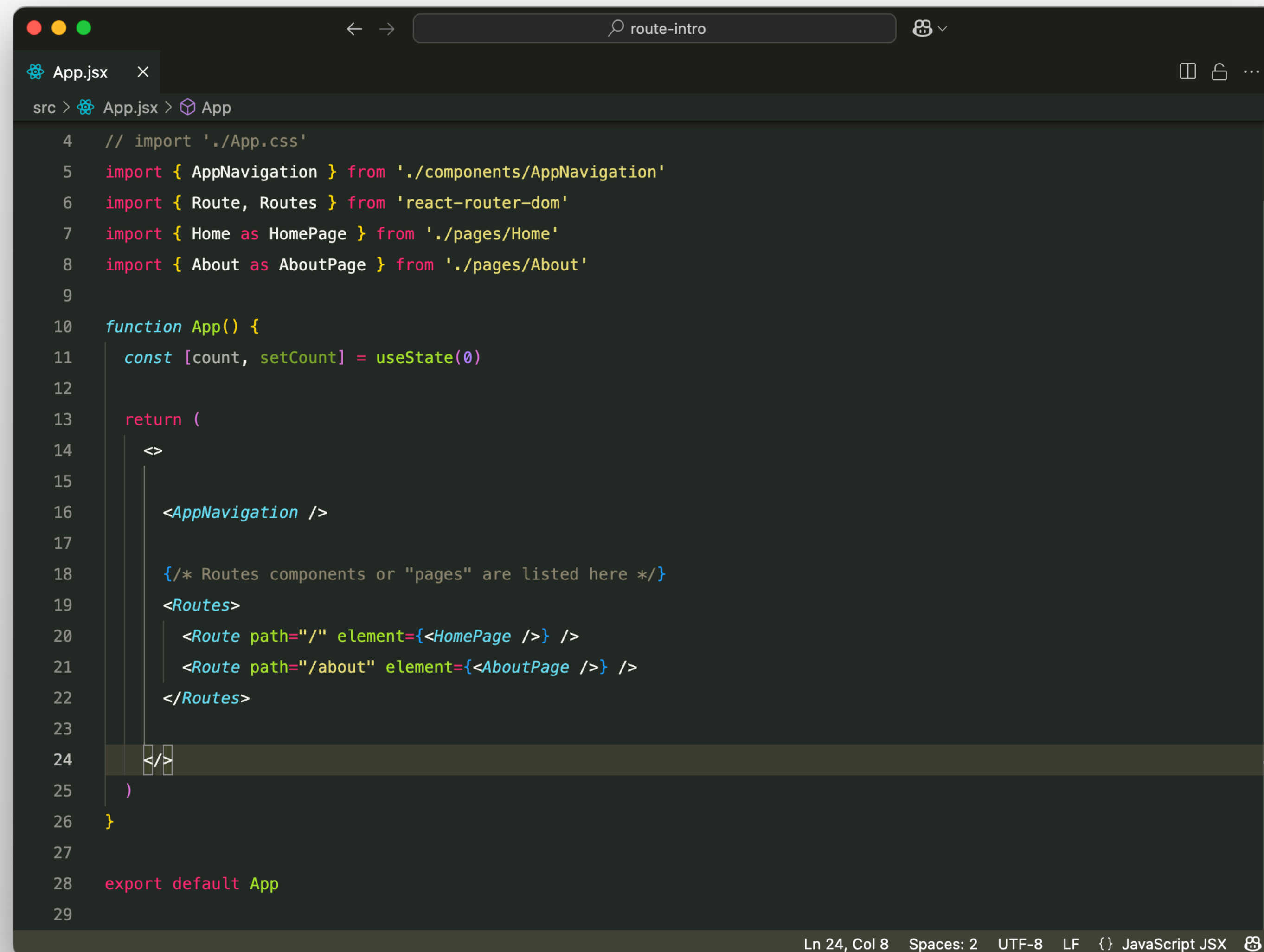
main.jsx with <Router> component

```jsx
import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import './index.css'
import App from './App.jsx'
import { BrowserRouter as Router } from "react-router-dom";


createRoot(document.getElementById('root')).render(
  <StrictMode>
    <Router>
      <App />
    </Router>
  </StrictMode>,
)
```

App.jsx with basic Routes

```jsx
4    // import './App.css'
5    import { AppNavigation } from './components/AppNavigation'
6    import { Route, Routes } from 'react-router-dom'
7    import { Home as HomePage } from './pages/Home'
8    import { About as AboutPage } from './pages/About'
9
10   function App() {
11     const [count, setCount] = useState(0)
12
13     return (
14       <>
15
16         <AppNavigation />
17
18         {/* Routes components or "pages" are listed here */}
19         <Routes>
20           <Route path="/" element={<HomePage />} />
21           <Route path="/about" element={<AboutPage />} />
22         </Routes>
23
24       </>
25     )
26   }
27
28   export default App
29
```
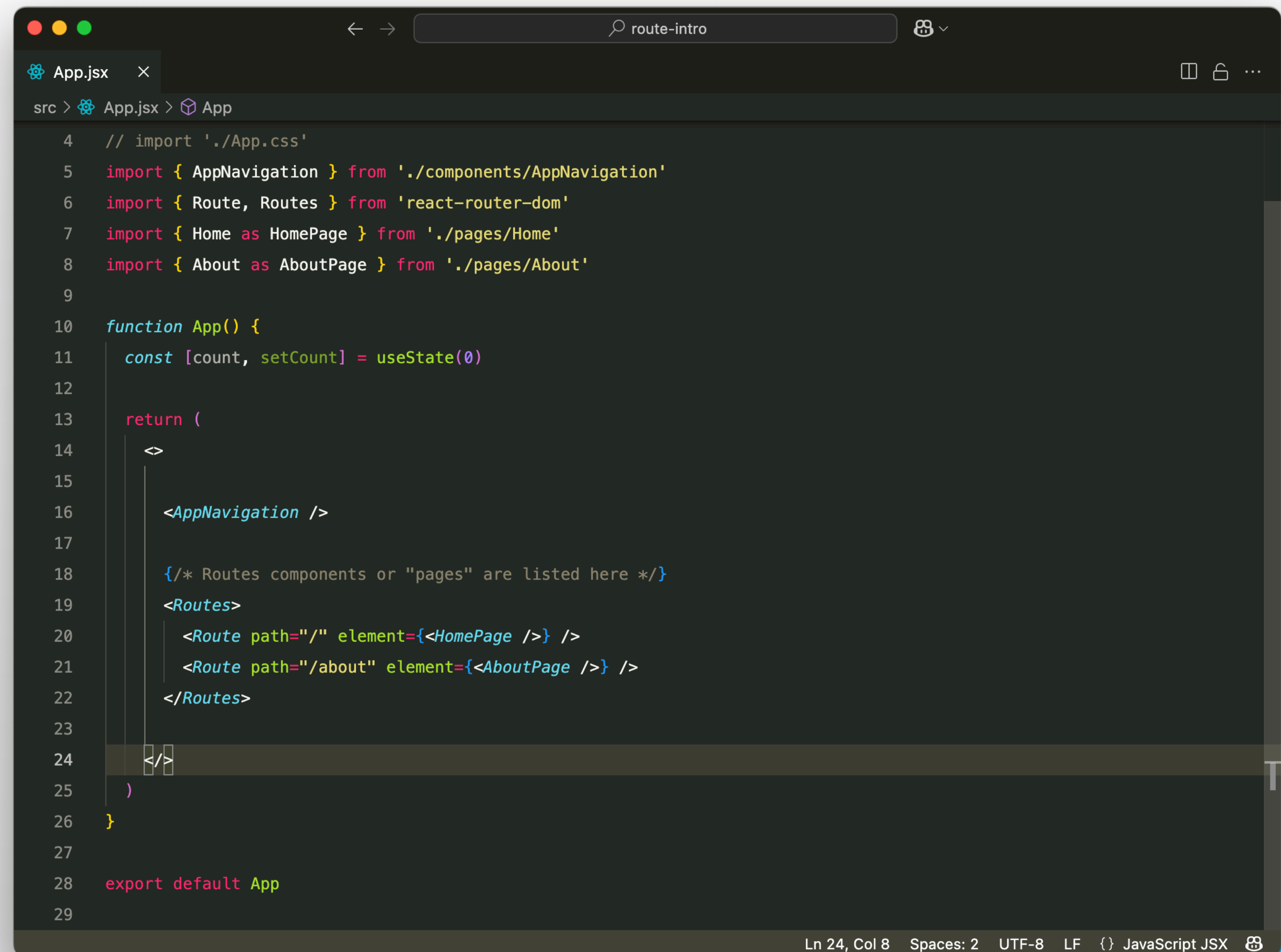
Pages are components

Pages are components too, passed by the **element** prop inside **<Route>**

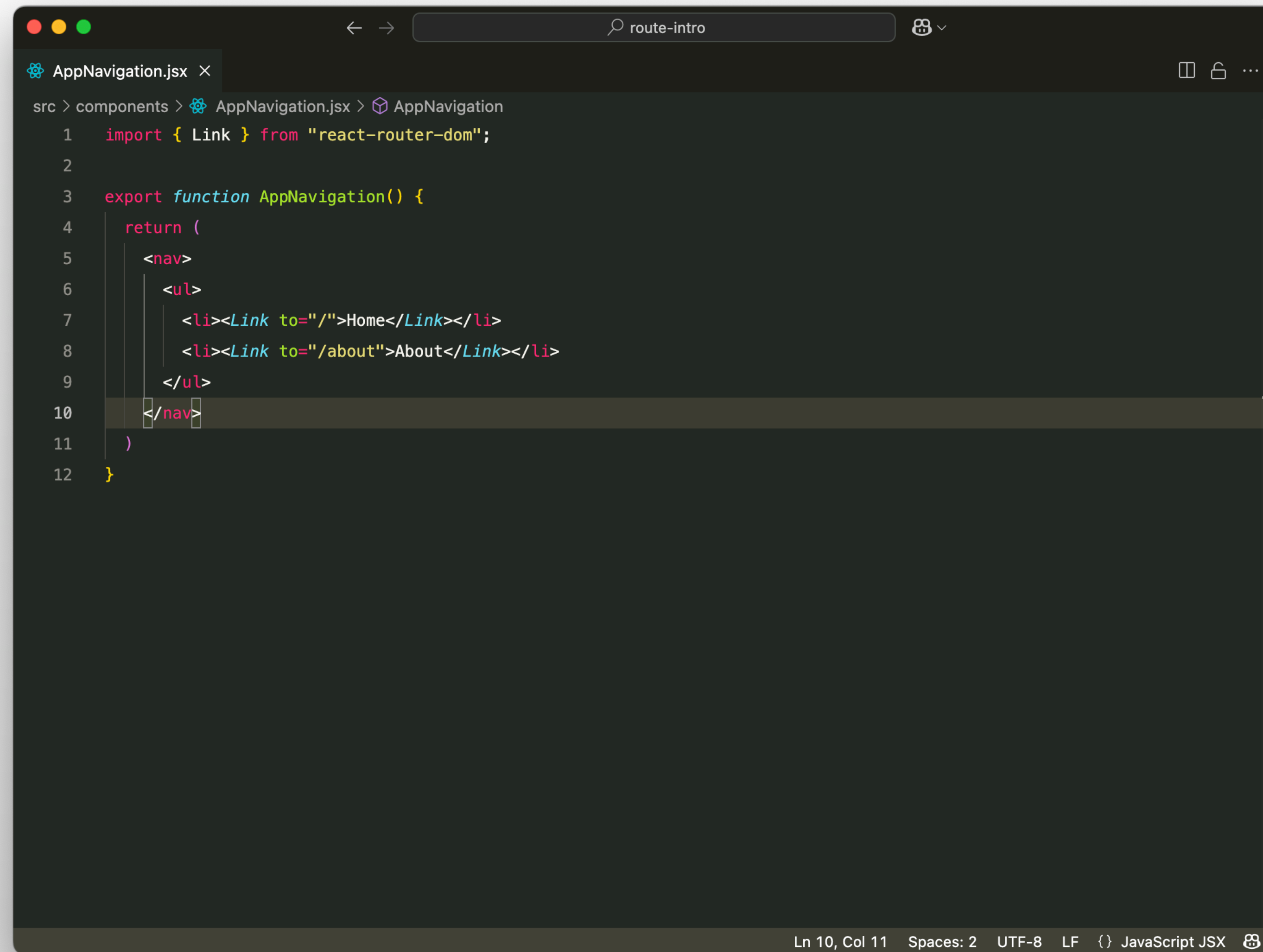We can store **Pages** on a different folder for better organisation.

```jsx
// import './App.css'
import { AppNavigation } from './components/AppNavigation'
import { Route, Routes } from 'react-router-dom'
import { Home as HomePage } from './pages/Home'
import { About as AboutPage } from './pages/About'

function App() {
  const [count, setCount] = useState(0)

  return (
    <>

      <AppNavigation />

      {/* Routes components or "pages" are listed here */}
      <Routes>
        <Route path="/" element={<HomePage />} />
        <Route path="/about" element={<AboutPage />} />
      </Routes>

    </>
  )
}

export default App
```

Pages can take props

```
<Route
  path="/about"
  element={<AboutPage content={aboutPageData} />}
  />
```

IRON
HACK

AppNavigation.jsx with <Link> component

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

export function AppNavigation() {
  return (
    <nav>
      <ul>
        <li><Link to="/">Home</Link></li>
        <li><Link to="/about">About</Link></li>
      </ul>
    </nav>
  )
}
```
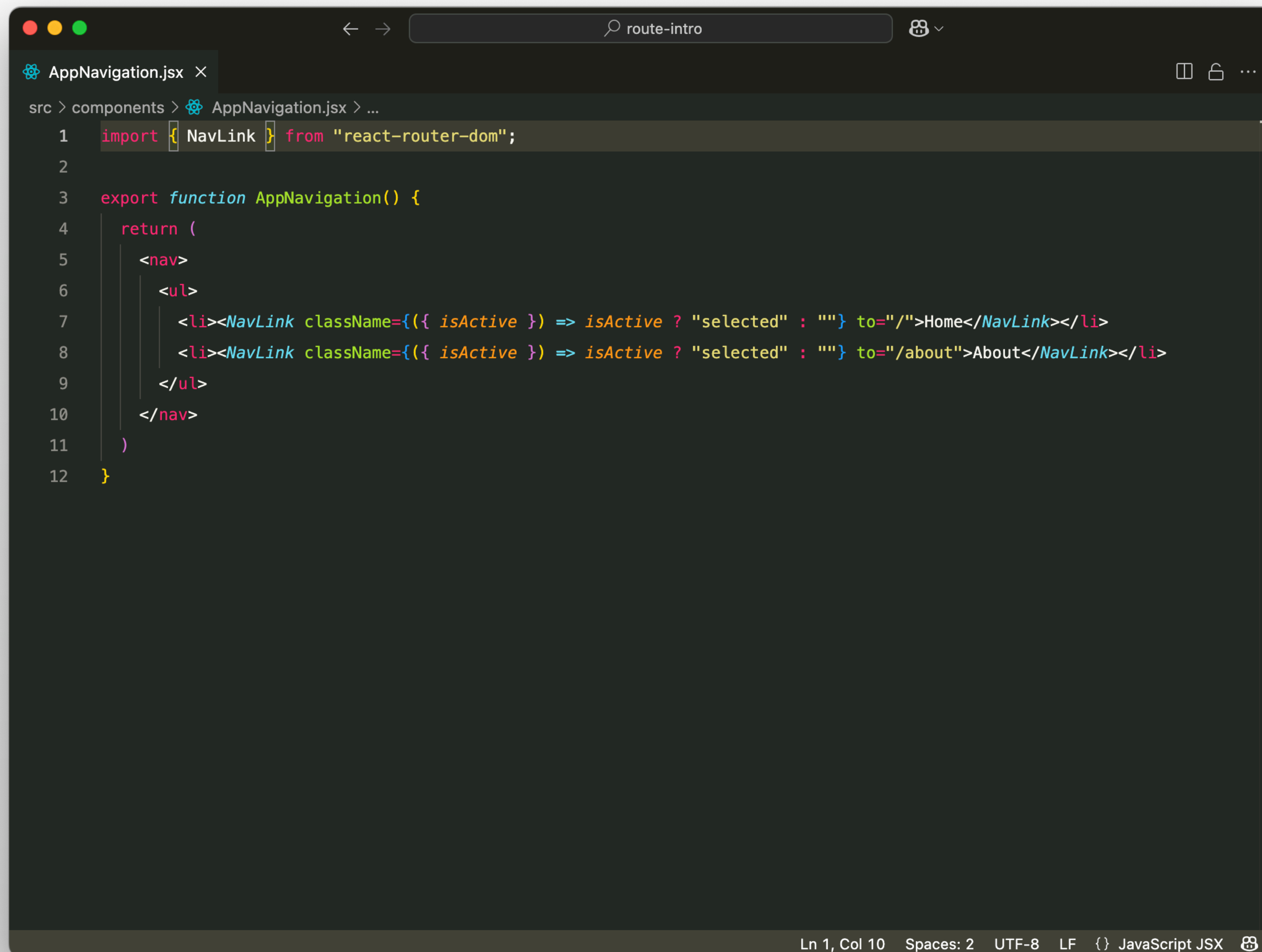
AppNavigation.jsx with <Link> component

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

export function AppNavigation() {
  return (
    <nav>
      <ul>
        <li><NavLink className={({ isActive }) => isActive ? "selected" : ""} to="/">Home</NavLink></li>
        <li><NavLink className={({ isActive }) => isActive ? "selected" : ""} to="/about">About</NavLink></li>
      </ul>
    </nav>
  )
}
```
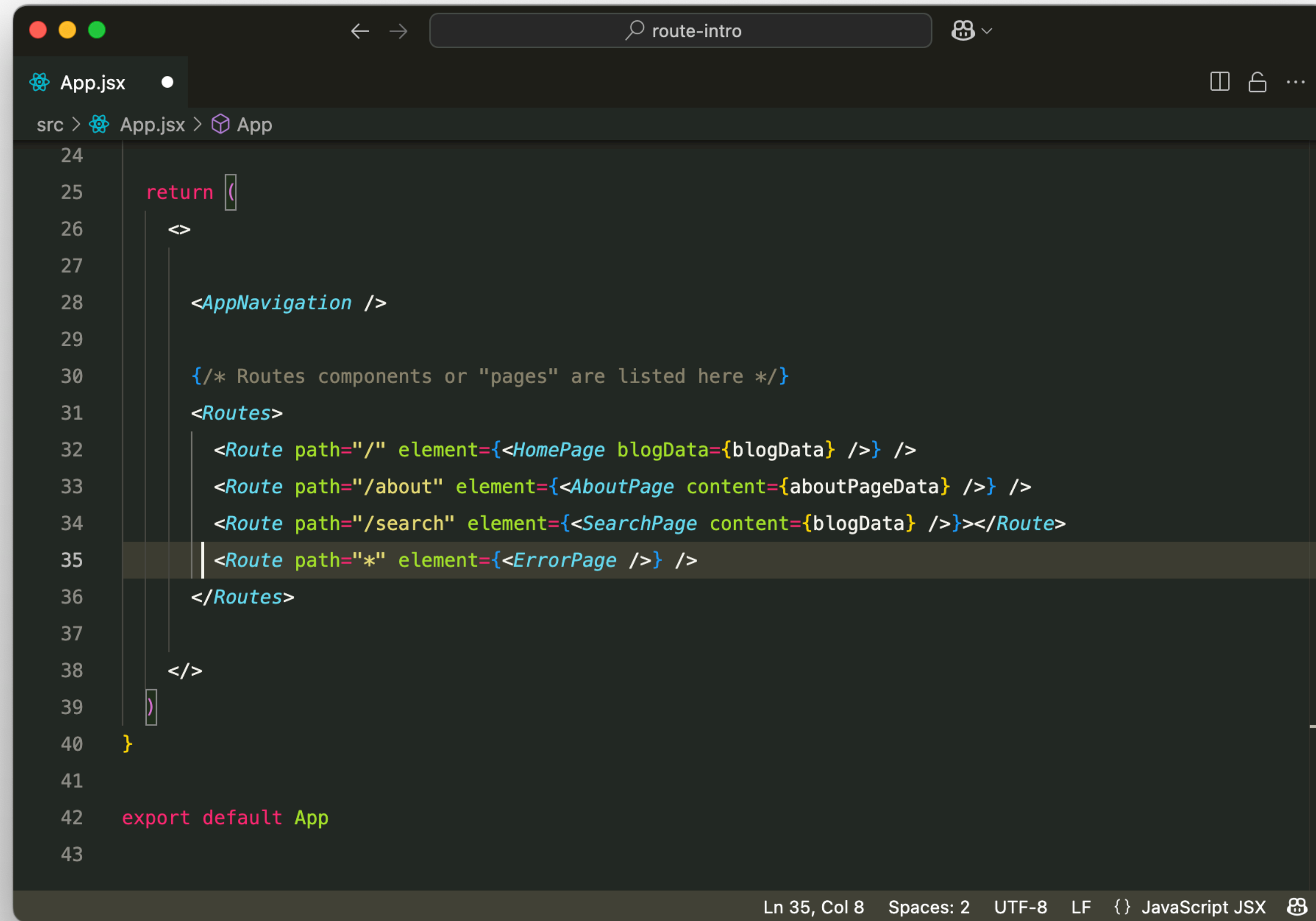
We use **<NavLink>** Components because it allow us to use **className** attr with **isActive** argument

IRON
HACK

Fallback route path="*"

```jsx
      return (
        <>

          <AppNavigation />

          {/* Routes components or "pages" are listed here */}
          <Routes>
            <Route path="/" element={<HomePage blogData={blogData} />} />
            <Route path="/about" element={<AboutPage content={aboutPageData} />} />
            <Route path="/search" element={<SearchPage content={blogData} />}></Route>
            <Route path="*" element={<ErrorPage />} />
          </Routes>

        </>
      )
    }

    export default App
```

path="*" will load a page when the URL does not match to anything previously defined

Dynamic Routes

What are dynamic routes?

Why do we need them on my app?

IRON
HACK

Dynamic Routes

Using url paths with id, slugs or arguments.

Eg:

```
/blog/:blogId > /blog/1239-01239-1802

/news/:newsId > /news/latest-news-today

/categories/:categorySlug > /categories/top

/search?keyword=foo
```

IRON
HACK

SingleBlogPage and :blogId

```
<Route
  path="/blog/:blogId"
  element={<SingleBlogPage content={blogData} />} />
```

:blogId is a URL parameter (not a prop)

:blogId can be accessed on
<SingleBlogPage>

IRON
HACK

SingleBlogPage and :blogId


```jsx
import { Link, useParams } from "react-router-dom"


export function SingleBlogPage ({ content }) {


  const { blogId } = useParams();


  // Rest of the code

```

Let's create our first React App with Routes

And remember… `npm install react-router-dom` 😉

IRON
HACK