

W7D1

miguel.garrido@ironhack.com

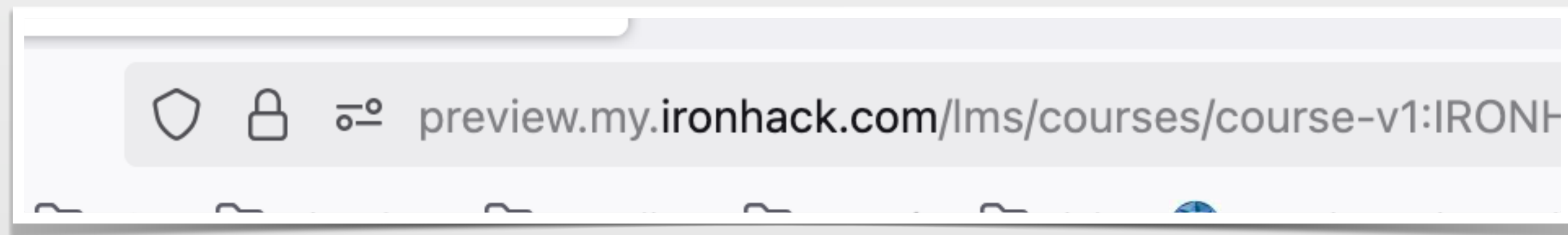


React Router

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



React works with the History API.



To install React Router

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



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>

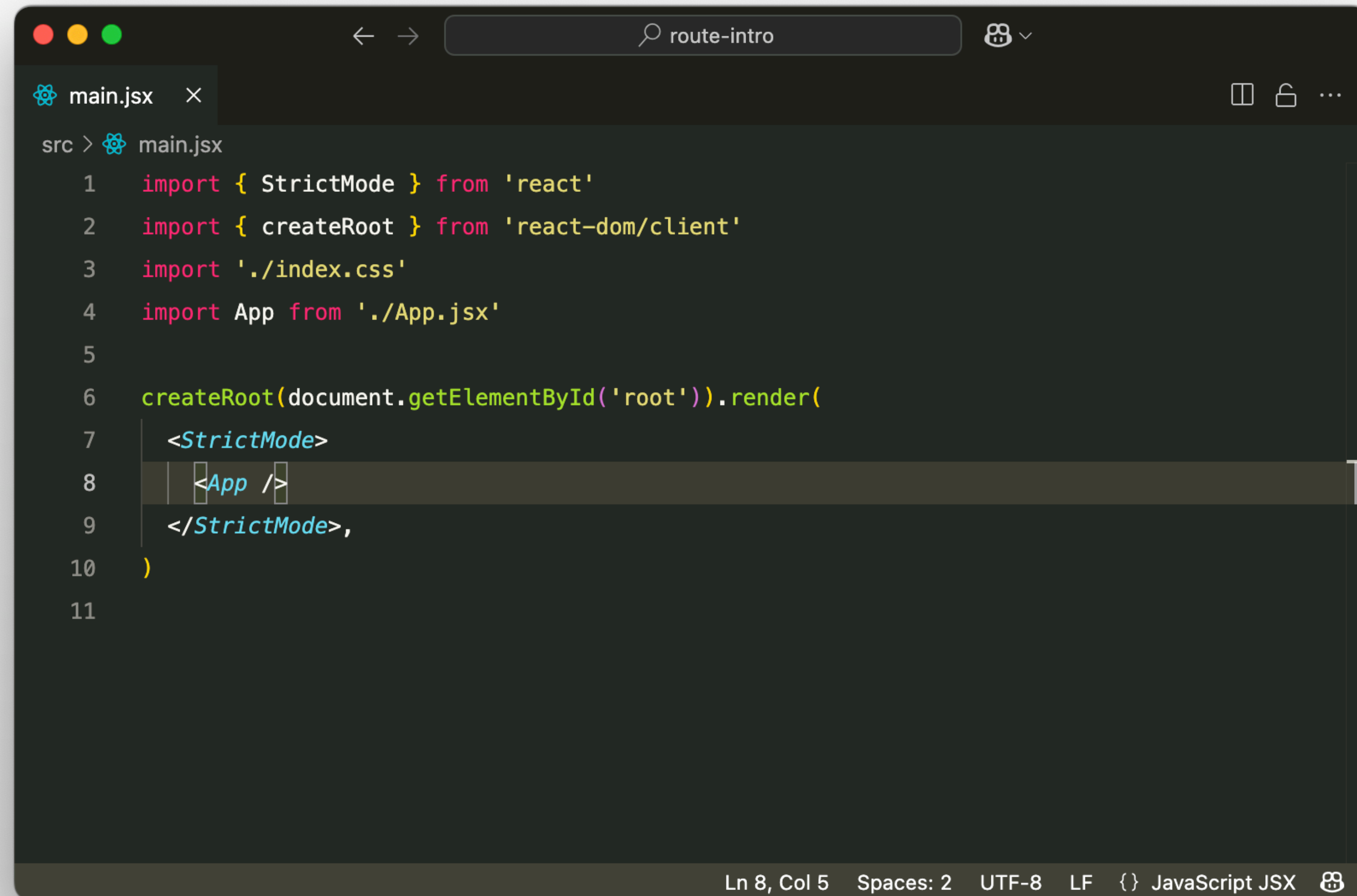
Renders a UI component depending on the URL.

<Link>

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



main.jsx

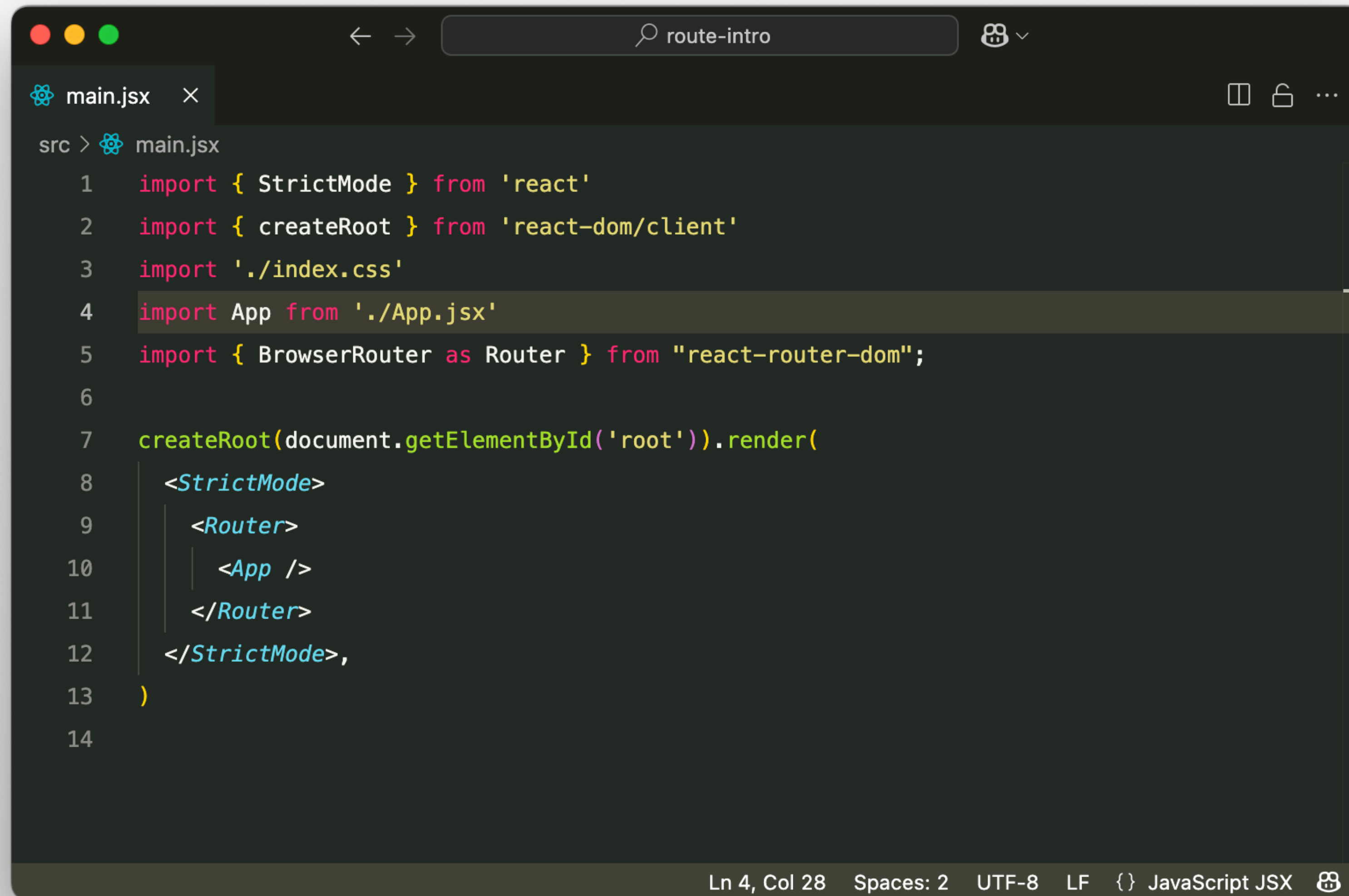


The image shows a code editor window with a dark theme. The title bar at the top includes window control buttons (red, yellow, green), navigation arrows, a search bar containing 'route-intro', and a GitHub icon. The editor has a tab labeled 'main.jsx' with a React icon and a close button. The file path 'src > main.jsx' is shown in the left margin. The code is as follows:

```
1  import { StrictMode } from 'react'
2  import { createRoot } from 'react-dom/client'
3  import './index.css'
4  import App from './App.jsx'
5
6  createRoot(document.getElementById('root')).render(
7    <StrictMode>
8      <App />
9    </StrictMode>,
10 )
11
```

The status bar at the bottom indicates the current cursor position is 'Ln 8, Col 5', with settings for 'Spaces: 2', 'UTF-8', 'LF', and the file type 'JavaScript JSX'.

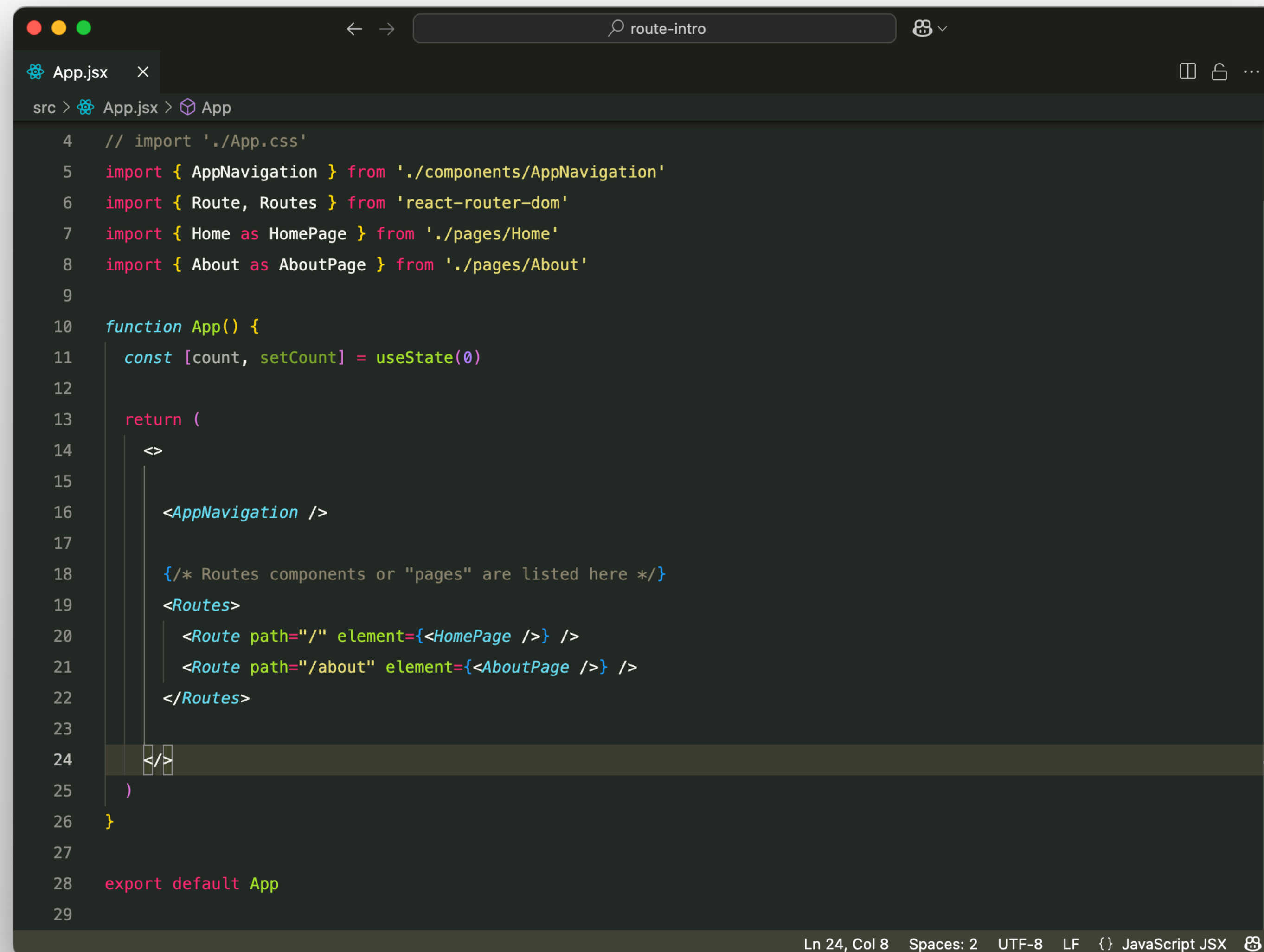
main.jsx with <Router> component



```
src > main.jsx
1  import { StrictMode } from 'react'
2  import { createRoot } from 'react-dom/client'
3  import './index.css'
4  import App from './App.jsx'
5  import { BrowserRouter as Router } from "react-router-dom";
6
7  createRoot(document.getElementById('root')).render(
8    <StrictMode>
9      <Router>
10        <App />
11      </Router>
12    </StrictMode>,
13  )
14
```

Ln 4, Col 28 Spaces: 2 UTF-8 LF {} JavaScript JSX

App.jsx with basic Routes



```
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       <AppNavigation />
16
17       {/* Routes components or "pages" are listed here */}
18       <Routes>
19         <Route path="/" element={<HomePage />} />
20         <Route path="/about" element={<AboutPage />} />
21       </Routes>
22     </>
23   )
24 }
25
26 export default App
```

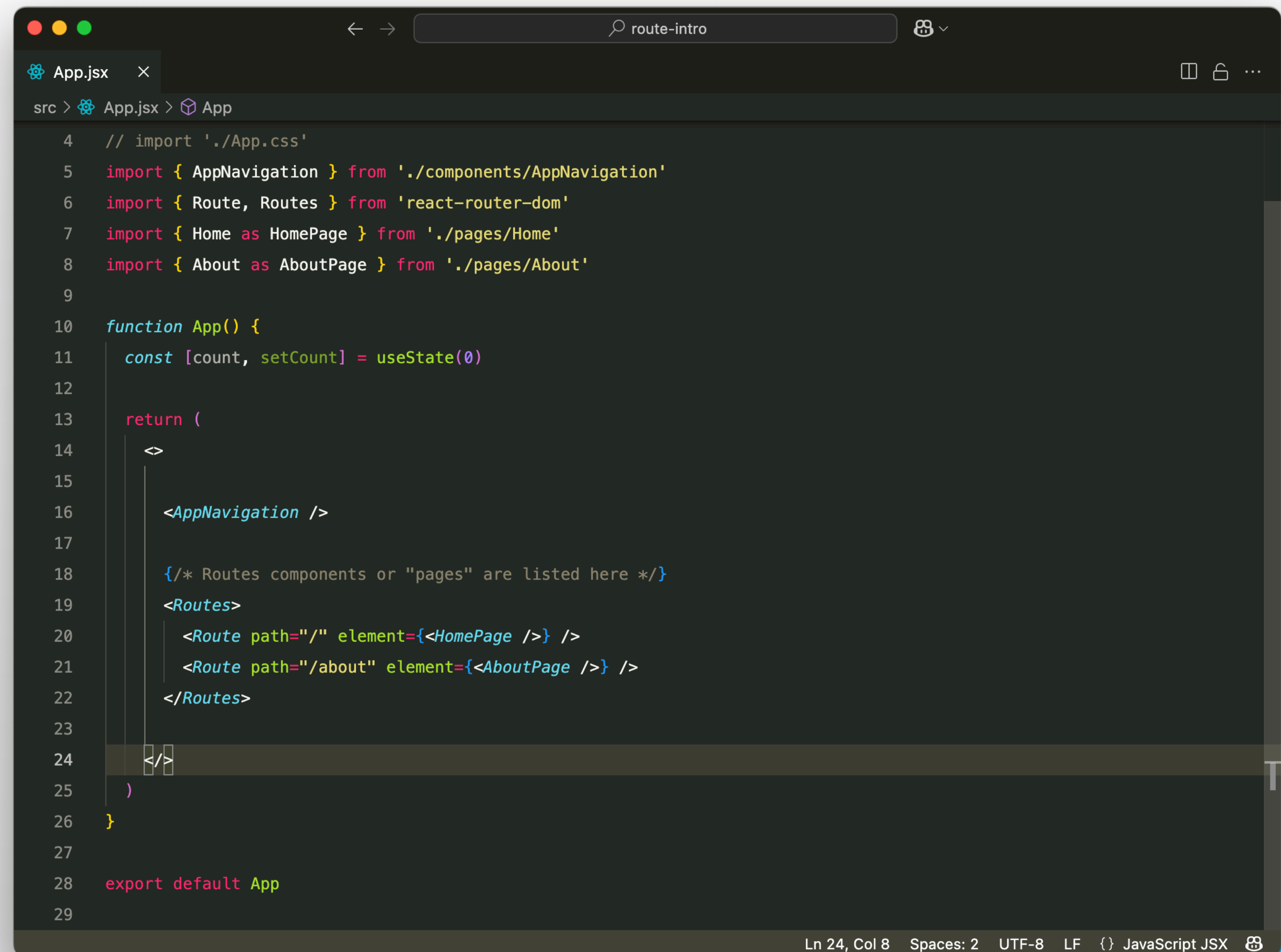
The screenshot shows a code editor window with a dark theme. The file name is 'App.jsx'. The code defines a function 'App()' which uses 'useState' to manage a 'count' state. It returns a JSX element containing an 'AppNavigation' component and a 'Routes' component. The 'Routes' component contains two 'Route' elements: one for the root path '/' pointing to 'HomePage' and another for '/about' pointing to 'AboutPage'. The editor interface includes a breadcrumb 'src > App.jsx > App', a search bar with 'route-intro', and a status bar at the bottom indicating 'Ln 24, Col 8', 'Spaces: 2', 'UTF-8', 'LF', and 'JavaScript JSX'.



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.



```
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       <AppNavigation />
16
17       {/* Routes components or "pages" are listed here */}
18       <Routes>
19         <Route path="/" element={<HomePage />} />
20         <Route path="/about" element={<AboutPage />} />
21       </Routes>
22     </>
23   )
24 }
25
26 export default App
```


Pages can take props

<Route

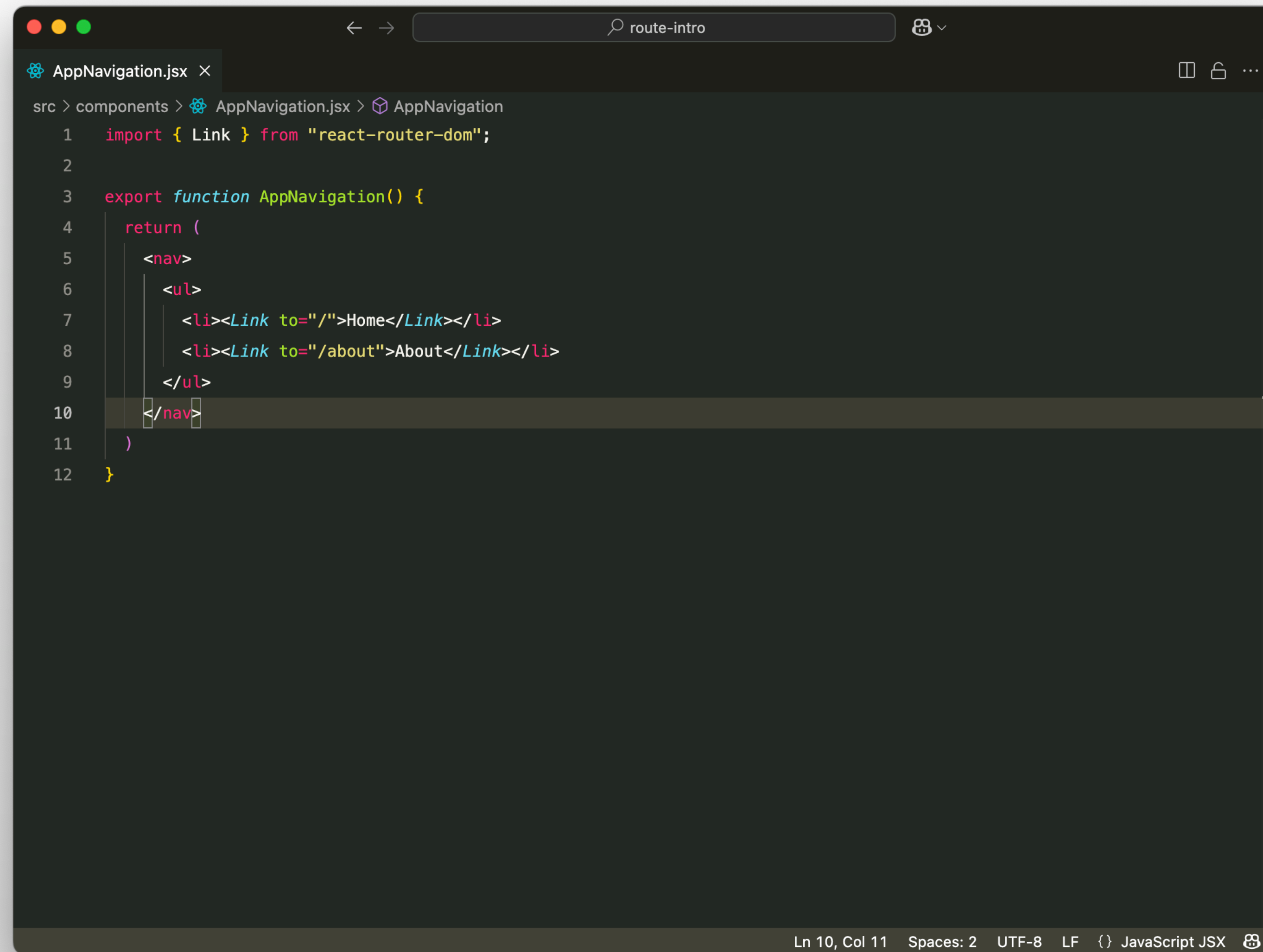
path="/about"

element={<AboutPage content={aboutPageData} />}

/>



AppNavigation.jsx with `<Link>` component

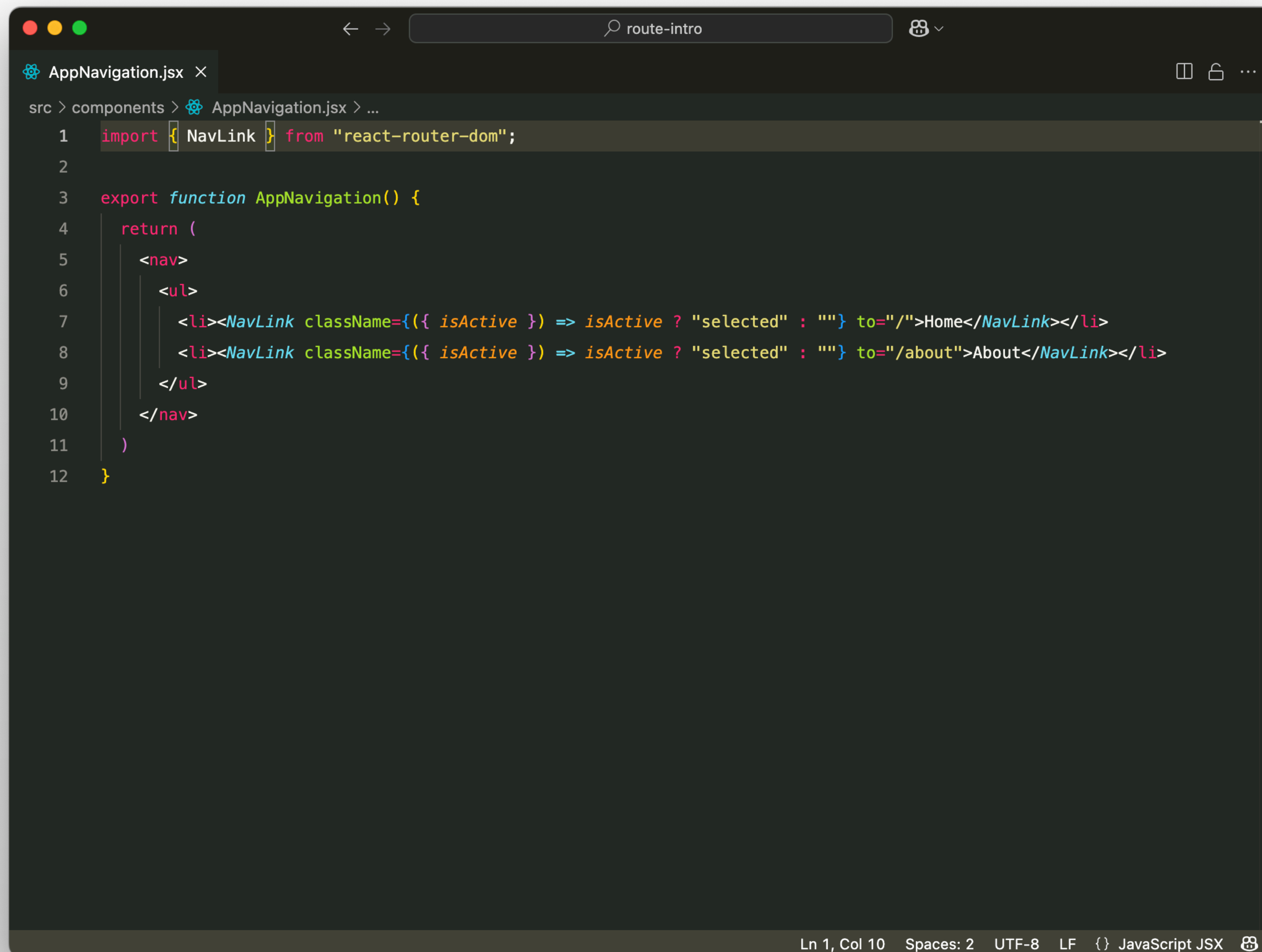


```
1 import { Link } from "react-router-dom";
2
3 export function AppNavigation() {
4   return (
5     <nav>
6       <ul>
7         <li><Link to="/">Home</Link></li>
8         <li><Link to="/about">About</Link></li>
9       </ul>
10    </nav>
11  )
12 }
```

Ln 10, Col 11 Spaces: 2 UTF-8 LF {} JavaScript JSX



AppNavigation.jsx with <Link> component

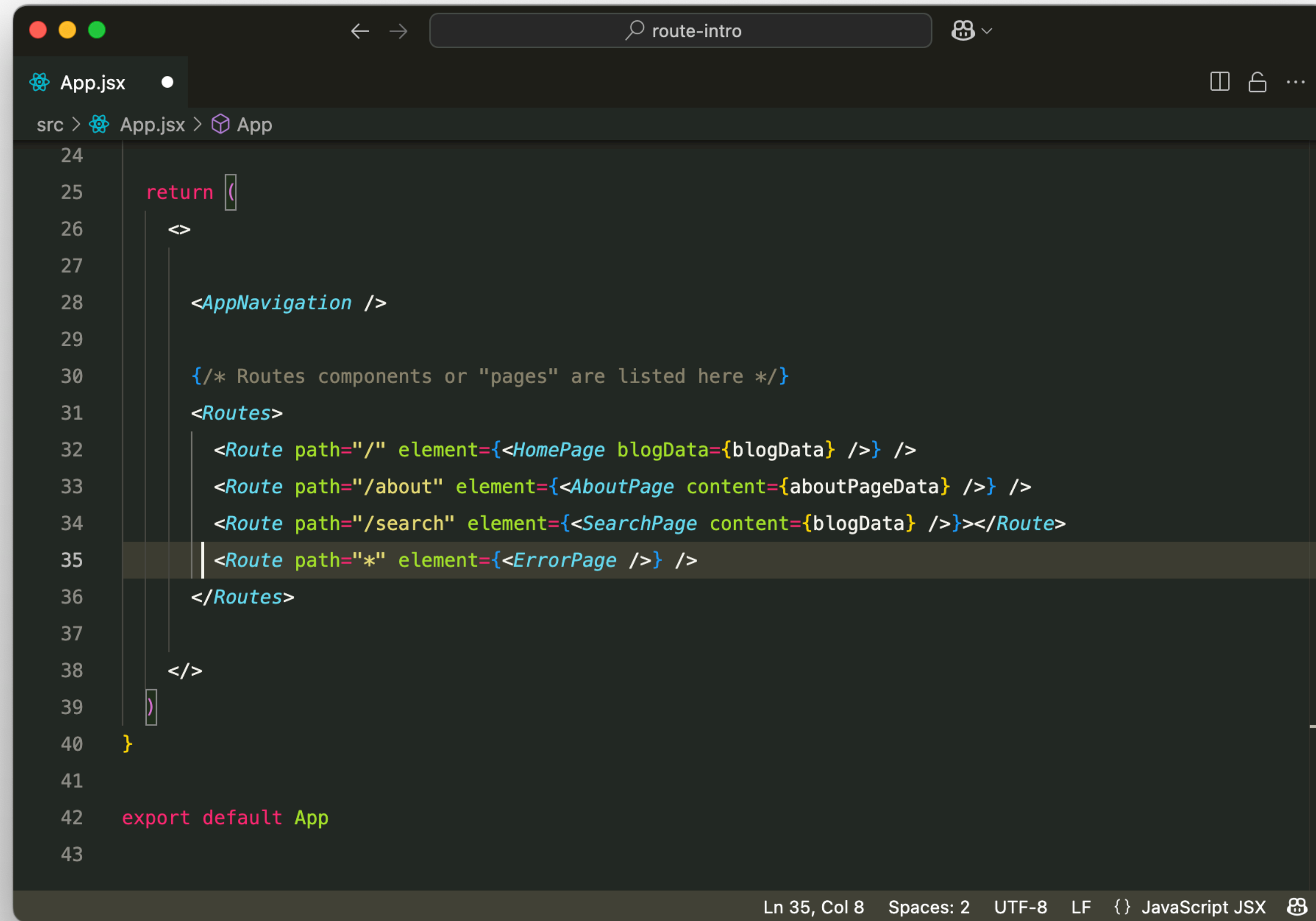


```
1 import { NavLink } from "react-router-dom";
2
3 export function AppNavigation() {
4   return (
5     <nav>
6       <ul>
7         <li><NavLink className={({ isActive }) => isActive ? "selected" : ""} to="/">Home</NavLink></li>
8         <li><NavLink className={({ isActive }) => isActive ? "selected" : ""} to="/about">About</NavLink></li>
9       </ul>
10    </nav>
11  )
12 }
```

We use <NavLink>
Components because it
allow us to use **className**
attr with **isActive** argument



Fallback route path="*"



```
24
25 return (
26   <>
27     <AppNavigation />
28     { /* Routes components or "pages" are listed here */ }
29     <Routes>
30       <Route path="/" element={<HomePage blogData={blogData} />} />
31       <Route path="/about" element={<AboutPage content={aboutPageData} />} />
32       <Route path="/search" element={<SearchPage content={blogData} />} />
33       <Route path="*" element={<ErrorPage />} />
34     </Routes>
35   </>
36 )
37
38
39 }
40
41
42 export default App
43
```

Ln 35, Col 8 Spaces: 2 UTF-8 LF {} JavaScript JSX

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?



Dynamic Routes

Using url paths with id, slugs or arguments.

Eg:

`/blog/:blogId`

`/news/:newsId`

`/categories/:categorySlug`

`/search?keyword=foo`



SingleBlogPage and :blogId

<Route

path="/blog/:blogId"

element={<SingleBlogPage content={blogData} />} />

:blogId is a URL parameter and not a prop.

:blogId can be accessed on
<SingleBlogPage>



SingleBlogPage and :blogId

```
src > pages > SingleBlog.jsx > SingleBlogPage
1  import { Link, useParams } from "react-router-dom"
2
3  export function SingleBlogPage ({ content }) {
4
5    const { blogId } = useParams();
6
7    // Rest of the code
8
```

Ln 7, Col 22 Spaces: 2 UTF-8 LF {} JavaScript JSX

Let's create our first React App with Routes

And remember... `npm install react-router-dom` 😊

