REACT ROUTER

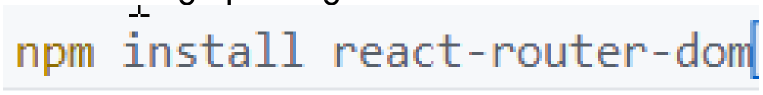
1

- The majority of websites were made up of a number of pages that users may see by requesting and viewing individual files.
- We have current location in the location bar, forward and back buttons to help navigate.
- For multi page server rendering web sites this is fine.
- React however is a single page app(SPA).
- These features will not work without a routing solution.

2

- The React provides code for UI.
- Developers have developed router specific code for React.
  - React Router
  - React Router DOM (Web)
  - React Router Native (Mobile)

- Before setting up navigation routes.



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

3

React Router

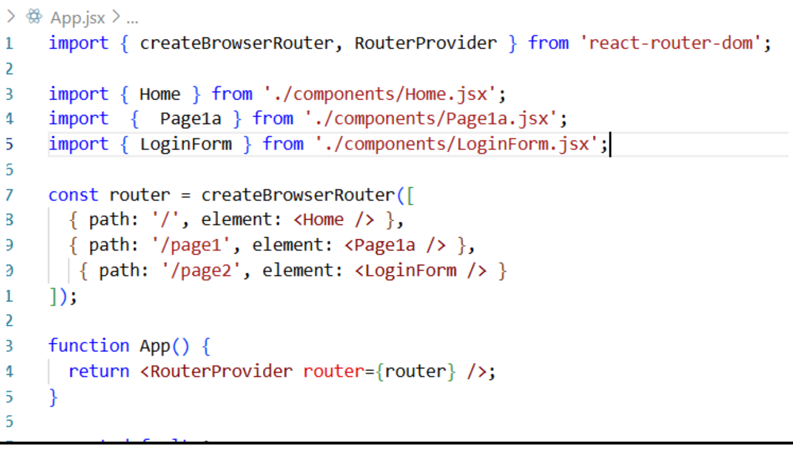
- Complex websites typically consist of multiple pages pages (e.g. an online shop with pages for products, orders etc).
- Without multiple pages
  - state and conditional values to display different content
- React Router package
  - Listen to URL path changes and display different components for different paths.
  - Technically it is a **SPA.**

4

- To create a new React Router framework project.
  - npx create-react-router@latest

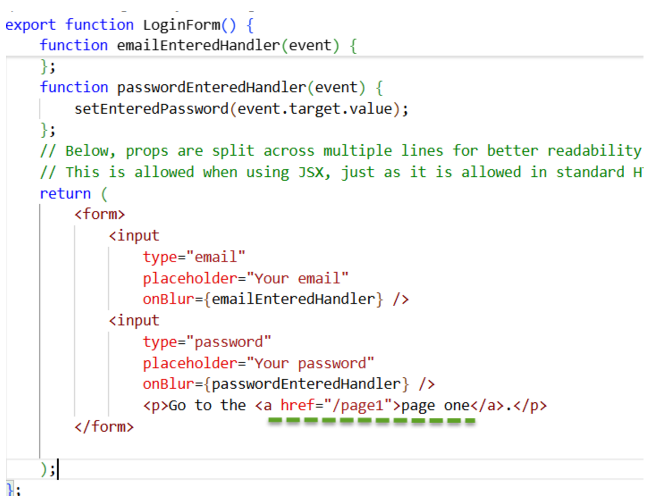
5

1. Define the routes using **createBrowserRouter**
2. Defining the root component



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

import { Home } from './components/Home.jsx';
import  {  Page1a } from './components/Page1a.jsx';
import { LoginForm } from './components/LoginForm.jsx';

const router = createBrowserRouter([
  { path: '/', element: <Home /> },
  { path: '/page1', element: <Page1a /> },
   { path: '/page2', element: <LoginForm /> }
]);

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

```

6

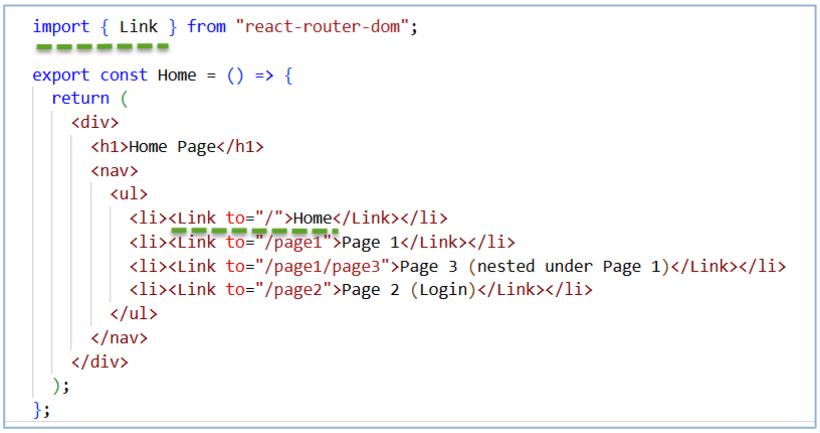


```
export function LoginForm() {
    function emailEnteredHandler(event) {
    };
    function passwordEnteredHandler(event) {
        setEnteredPassword(event.target.value);
    };
    // Below, props are split across multiple lines for better readability
    // This is allowed when using JSX, just as it is allowed in standard H
    return (
        <form>
            <input
                type="email"
                placeholder="Your email"
                onBlur={emailEnteredHandler} />
            <input
                type="password"
                placeholder="Your password"
                onBlur={passwordEnteredHandler} />
                <p>Go to the <a href="/page1">page one</a>.</p>
        </form>

    );
};
```

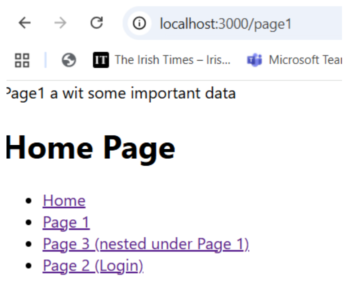
- HTTP request is sent to the server whenever a link is clicked.
- Issues
  - shared state.
  - the browser to download all website assets (e.g., script files) again

7

- Access internal pages by clicking on a link.
- The `to` prop, specifies the link to navigate.



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

export const Home = () => {
  return (
    <div>
      <h1>Home Page</h1>
      <nav>
        <ul>
          <li><Link to="/">Home</Link></li>
          <li><Link to="/page1">Page 1</Link></li>
          <li><Link to="/page1/page3">Page 3 (nested under Page 1)</Link></li>
          <li><Link to="/page2">Page 2 (Login)</Link></li>
        </ul>
      </nav>
    </div>
  );
};
```



age1 a wit some important data

Home Page

- Home
- Page 1
- Page 3 (nested under Page 1)
- Page 2 (Login)

8

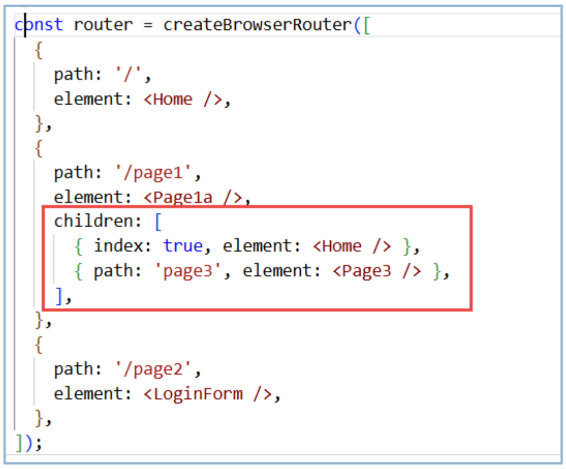
NavLink

- **NavLink** can detect when its target route is active.
- You can style the active link dynamically (`className` or `style`).
- end on the `/page1` link ensures it's only active on that exact path (not on subroutes like `/page1/page3`

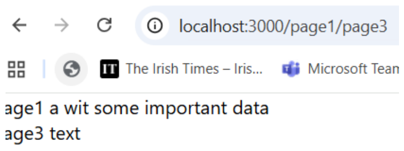
9

Wrapping routes

- Wrap other routes with a special **children** property



```
const router = createBrowserRouter([
  {
    path: '/',
    element: <Home />,
  },
  {
    path: '/page1',
    element: <Page1a />,
    children: [
      { index: true, element: <Home /> },
      { path: 'page3', element: <Page3 /> },
    ],
  },
  {
    path: '/page2',
    element: <LoginForm />,
  },
]);
```



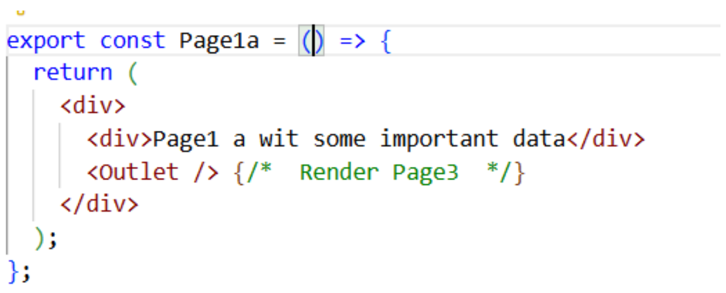
```
localhost:3000/page1/page3
```

age1 a wit some important data
age3 text

10

Outlet

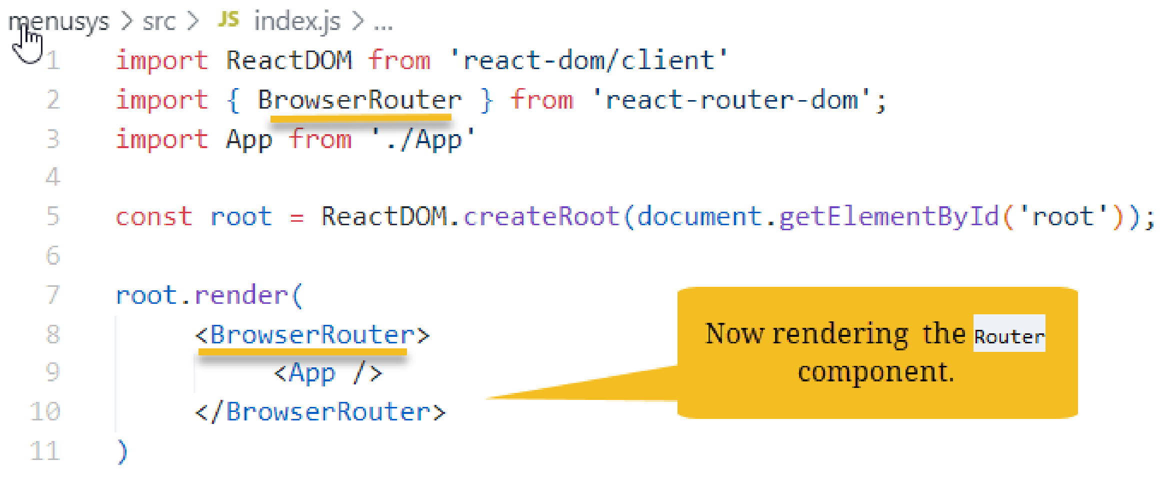
- Placeholder for rendering **nested routes** inside a parent route.
- `<Outlet />` injects child components in the parent's layout
- Without using `<Outlet />` the component has no where to render.



```
export const Page1a = () => {
  return (
    <div>
      <div>Page1 a wit some important data</div>
      <Outlet /> {/*  Render Page3  */}
    </div>
  );
};
```

11

index.js



menusys > src > JS index.js > ...

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

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(
      <BrowserRouter>
            <App />
      </BrowserRouter>
)
```

12

- route configuration places in the `App.js`
- Wrapper component for routes we want to render is called Routes.
- Inside of Routes, there is a Route component for each page that has to be rendered.

13

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

….
return (
    <>
    <Routes>
        <Route path="/">
          <Route index element={<Home2/>} />
          <Route path="/Page1" element={<Page1 />} />
          <Route path="/Page2" element={<Page2 />} />
          <Route path="*" element={<Page404 />} />
        </Route>
      </Routes>
    </>
  );
```

Route map the app's location to different React components

Route component for each page to be rendered

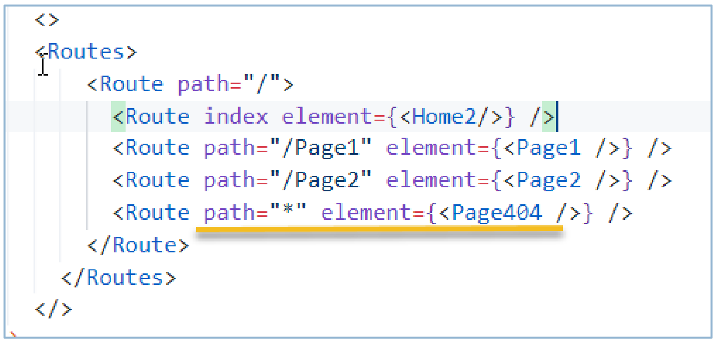
14

- Each `Route` component has `path` and `element` properties.
- When the browser's location matches the path, the element will be displayed.
- When the location is `/`, the `router` will render the `Home` component.

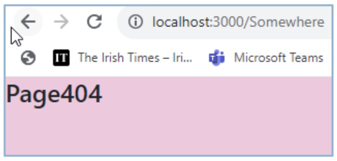
15

404 page

- Visiting pages that do not exist.
- The * path value and the component as the element are used to help unresolved paths

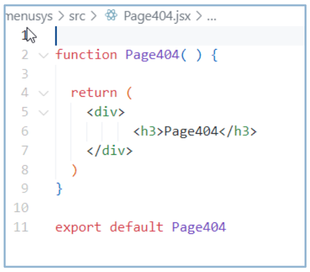


```
<>
<Routes>
    <Route path="/">
      <Route index element={<Home2/>} />
      <Route path="/Page1" element={<Page1 />} />
      <Route path="/Page2" element={<Page2 />} />
      <Route path="*" element={<Page404 />} />
    </Route>
  </Routes>
</>
```



localhost:3000/Somewhere

Page404



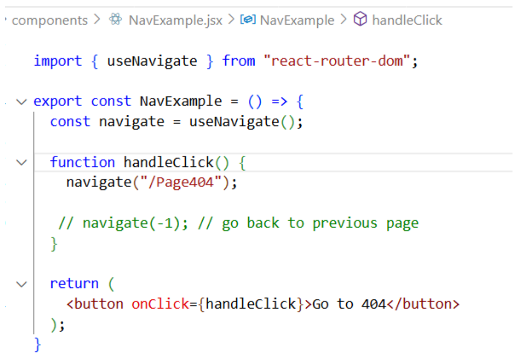
nenusys > src > Page404.jsx > ...

```
function Page404( ) {

  return (
    <div>
        <h3>Page404</h3>
    </div>
  )
}

export default Page404
```

16

Redirecting

- When using React Router, the default behavior is to forward users.
- `useNavigate` lets you change routes programmatically

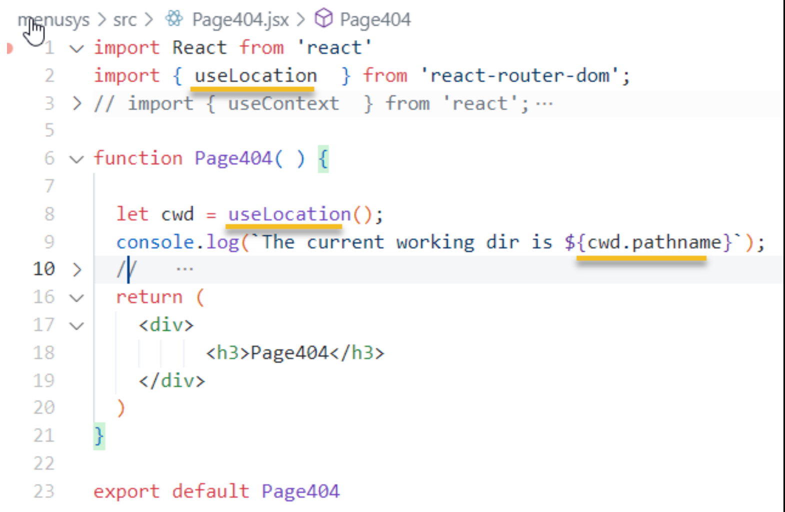


```
import { useNavigate } from "react-router-dom";

export const NavExample = () => {
  const navigate = useNavigate();

  function handleClick() {
    navigate("/Page404");

   // navigate(-1); // go back to previous page
  }

  return (
    <button onClick={handleClick}>Go to 404</button>
  );
}
```

17

useLocation

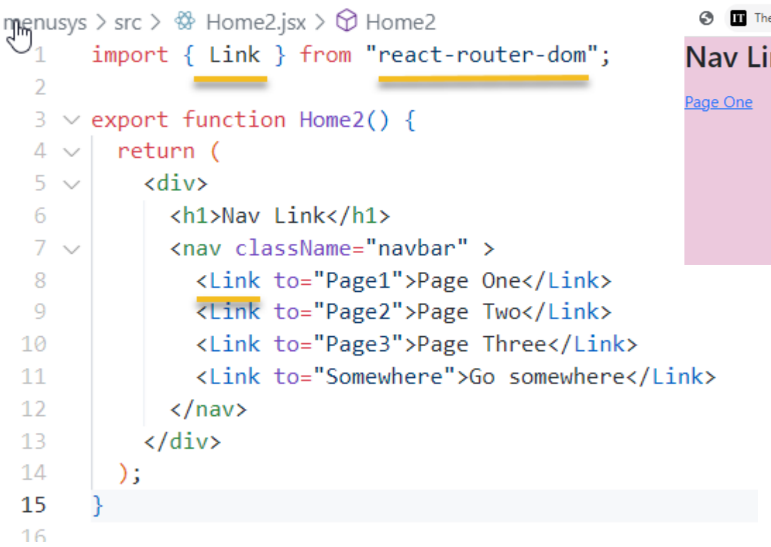
- **useLocation** lets you access information about the current URL



```
import React from 'react'
import { useLocation  } from 'react-router-dom';
// import { useContext  } from 'react';…

function Page404( ) {

  let cwd = useLocation();
  console.log(`The current working dir is ${cwd.pathname}`);
  //  …
  return (
    <div>
        <h3>Page404</h3>
    </div>
  )
}

export default Page404
```

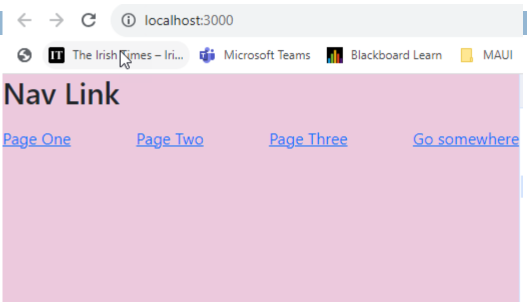
18

Link component



menusys > src > Home2.jsx > Home2

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

export function Home2() {
  return (
    <div>
      <h1>Nav Link</h1>
      <nav className="navbar" >
        <Link to="Page1">Page One</Link>
        <Link to="Page2">Page Two</Link>
        <Link to="Page3">Page Three</Link>
        <Link to="Somewhere">Go somewhere</Link>
      </nav>
    </div>
  );
}
```



localhost:3000

The Irish Times – Iri... Microsoft Teams Blackboard Learn MAUI

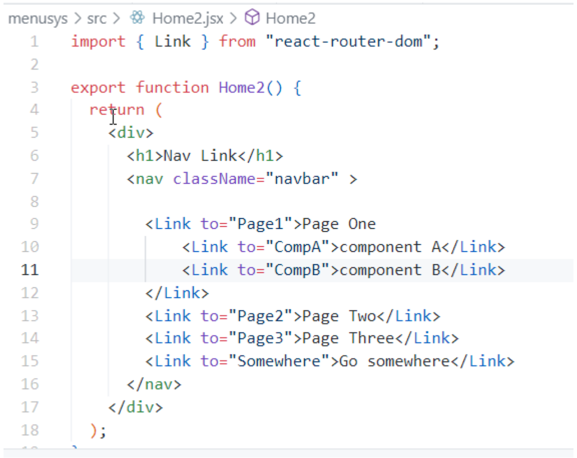
Nav Link

Page One Page Two Page Three Go somewhere

19

Nesting Routes

- Organise web apps into hierarchies

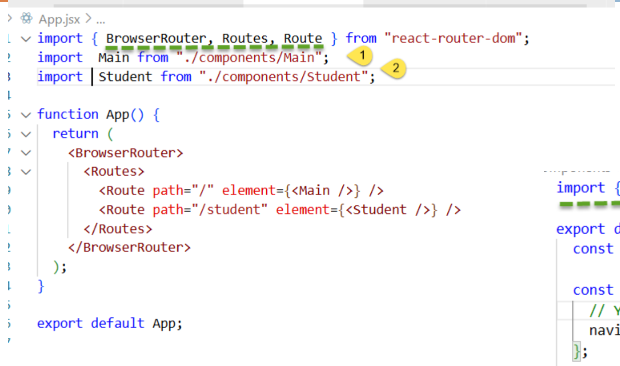


menusys > src > Home2.jsx > Home2

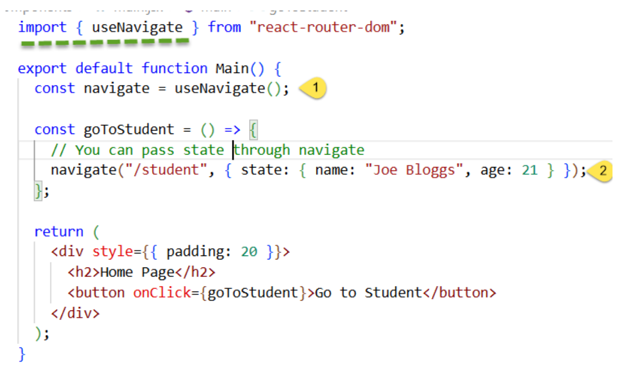
```
import { Link } from "react-router-dom";

export function Home2() {
  return (
    <div>
      <h1>Nav Link</h1>
      <nav className="navbar" >

        <Link to="Page1">Page One
            <Link to="CompA">component A</Link>
            <Link to="CompB">component B</Link>
        </Link>
        <Link to="Page2">Page Two</Link>
        <Link to="Page3">Page Three</Link>
        <Link to="Somewhere">Go somewhere</Link>
      </nav>
    </div>
  );
```

20

Sample example

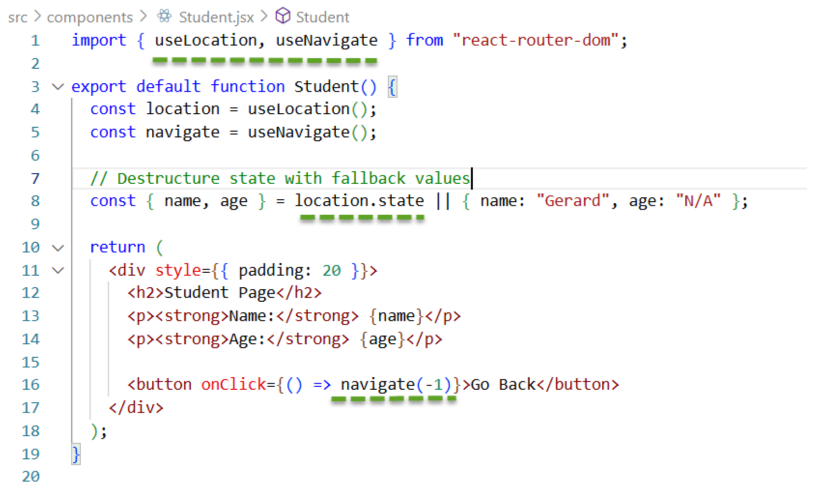


```
App.jsx > ...
import { BrowserRouter, Routes, Route } from "react-router-dom";
import  Main from "./components/Main";
import Student from "./components/Student";

function App() {
  return (
    <BrowserRouter>
      <Routes>
        <Route path="/" element={<Main />} />
        <Route path="/student" element={<Student />} />
      </Routes>
    </BrowserRouter>
  );
}

export default App;
```

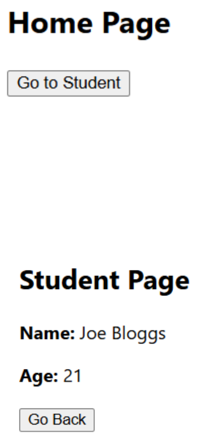


```
import { useNavigate } from "react-router-dom";

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

  const goToStudent = () => {
    // You can pass state through navigate
    navigate("/student", { state: { name: "Joe Bloggs", age: 21 } });
  };

  return (
    <div style={{ padding: 20 }}>
      <h2>Home Page</h2>
      <button onClick={goToStudent}>Go to Student</button>
    </div>
  );
}
```

21



```
src > components > Student.jsx > Student
import { useLocation, useNavigate } from "react-router-dom";

export default function Student() {
  const location = useLocation();
  const navigate = useNavigate();

  // Destructure state with fallback values
  const { name, age } = location.state || { name: "Gerard", age: "N/A" };

  return (
    <div style={{ padding: 20 }}>
      <h2>Student Page</h2>
      <p><strong>Name:</strong> {name}</p>
      <p><strong>Age:</strong> {age}</p>

      <button onClick={() => navigate(-1)}>Go Back</button>
    </div>
  );
}
```



Home Page

Student Page

Name: Joe Bloggs

Age: 21

22

Summary

- React itself doesn't include routing, use `react-router-dom`
- Wrap your app in a router'
  - `BrowserRouter` top-level router for web apps.
  - `Routes` groups routes.
  - `Route` defines the path and the component to render.
- Navigation Components
  - `NavLink` adds an active class automatically
- Use `<Outlet />` with nested routes
- `useNavigate()` hook to navigate

23