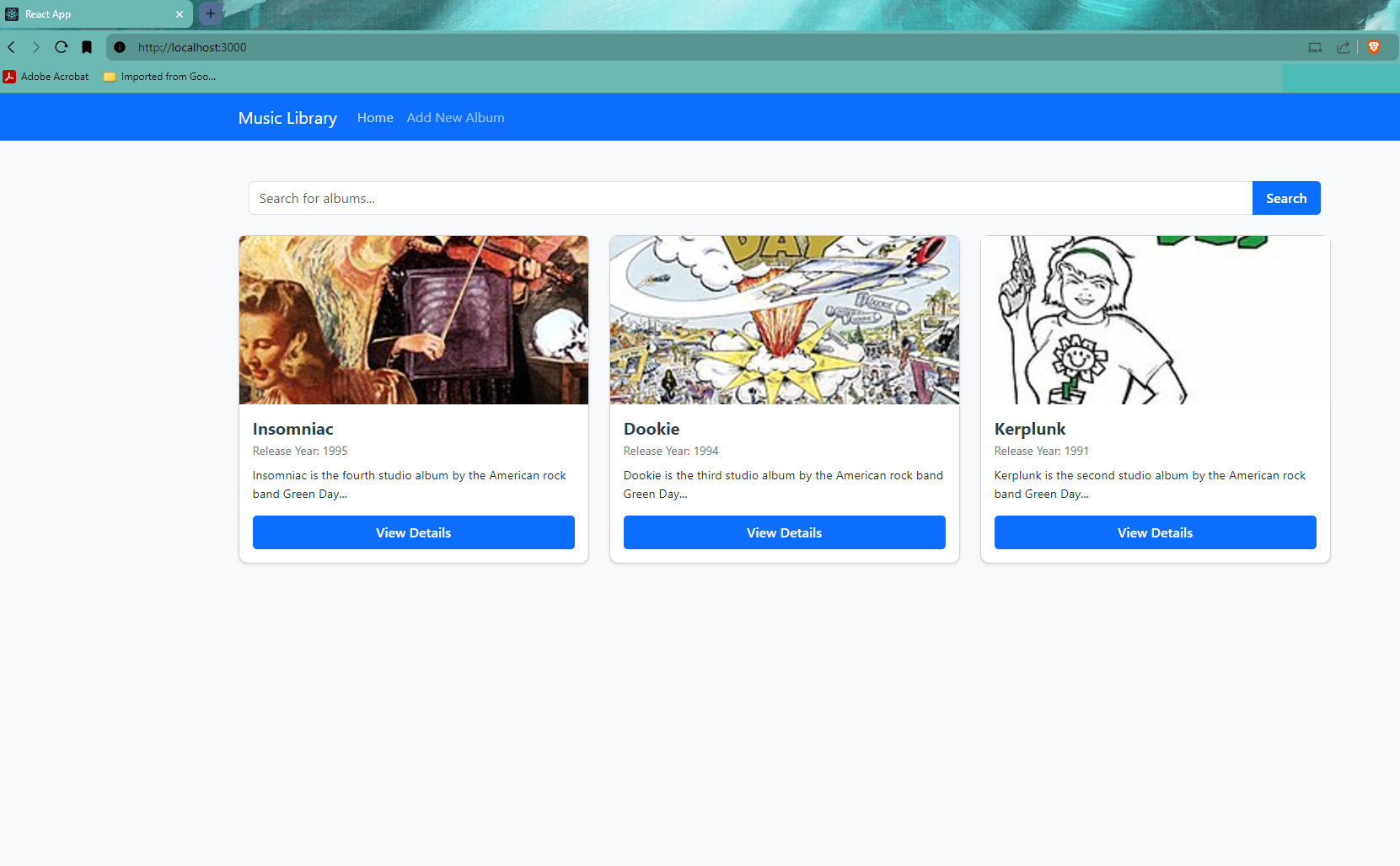
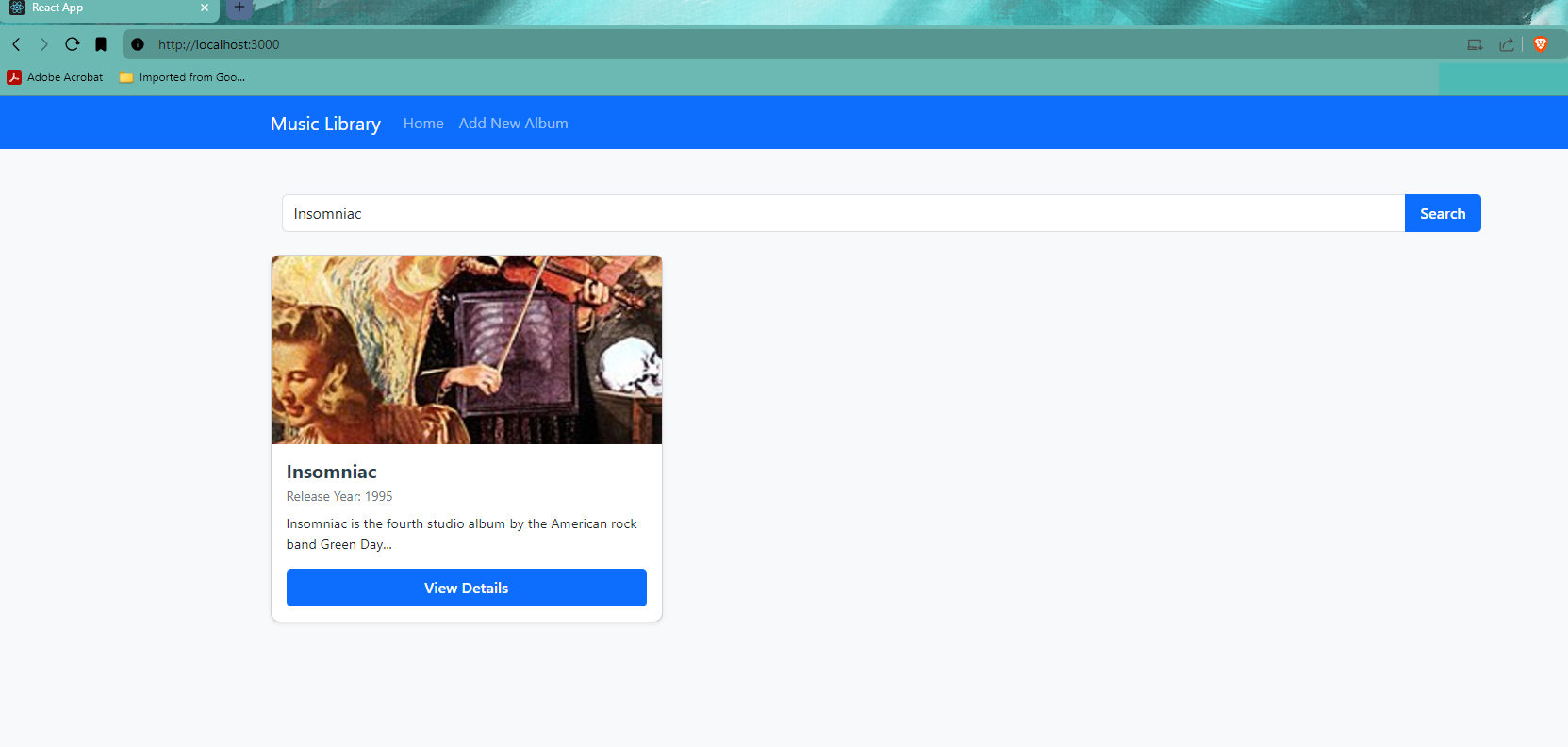
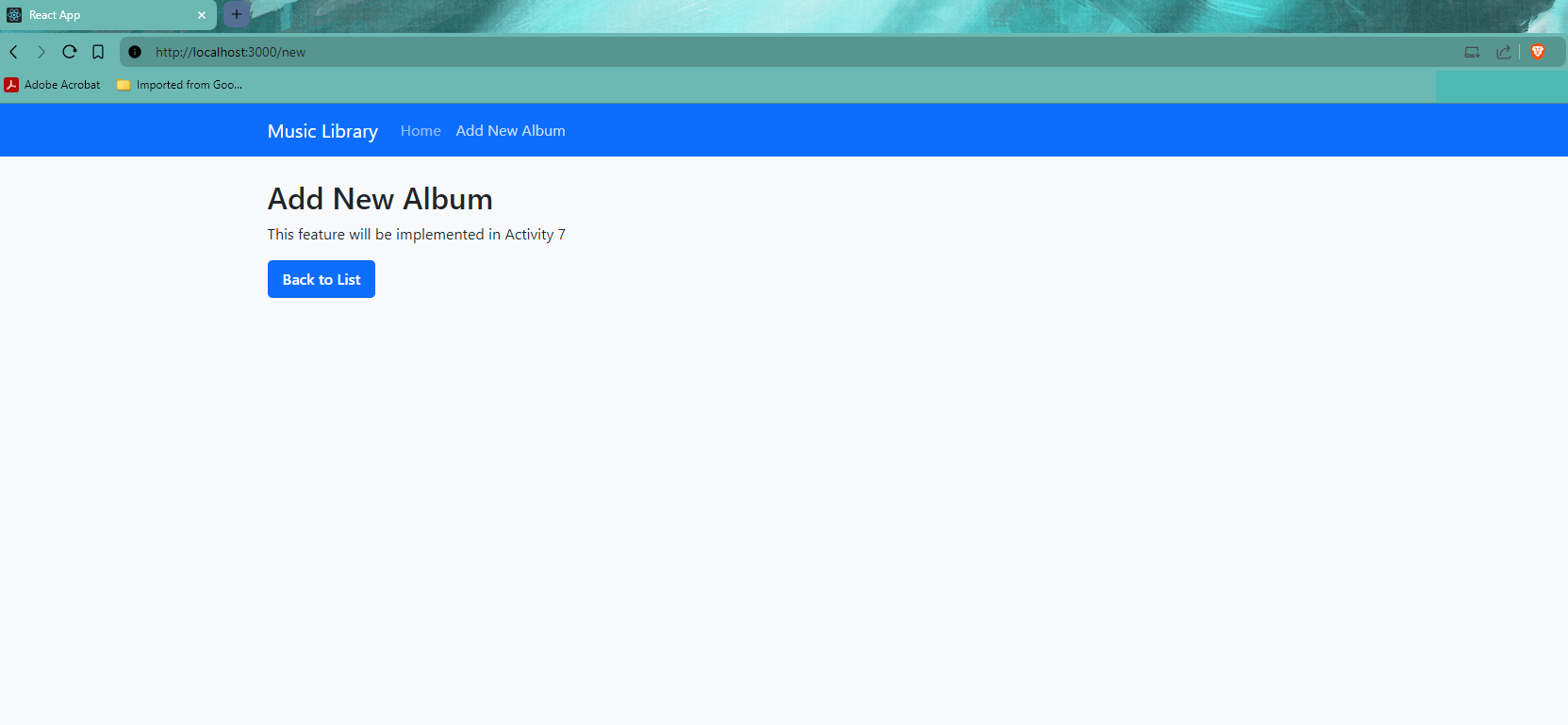
*Owen Lindsey*

*Professor Sparks, James*

*CST-391 Activity 6*

*11/24/2024*

**Part 3 External Data Source:**   
*Screenshots*  
  
*Screenshot of the home page for music app:*  
  
  
*Screenshot of the search bar functioning:*  


*Screenshot of the Add new Album page:*  


**Part 3 External Data Source:**

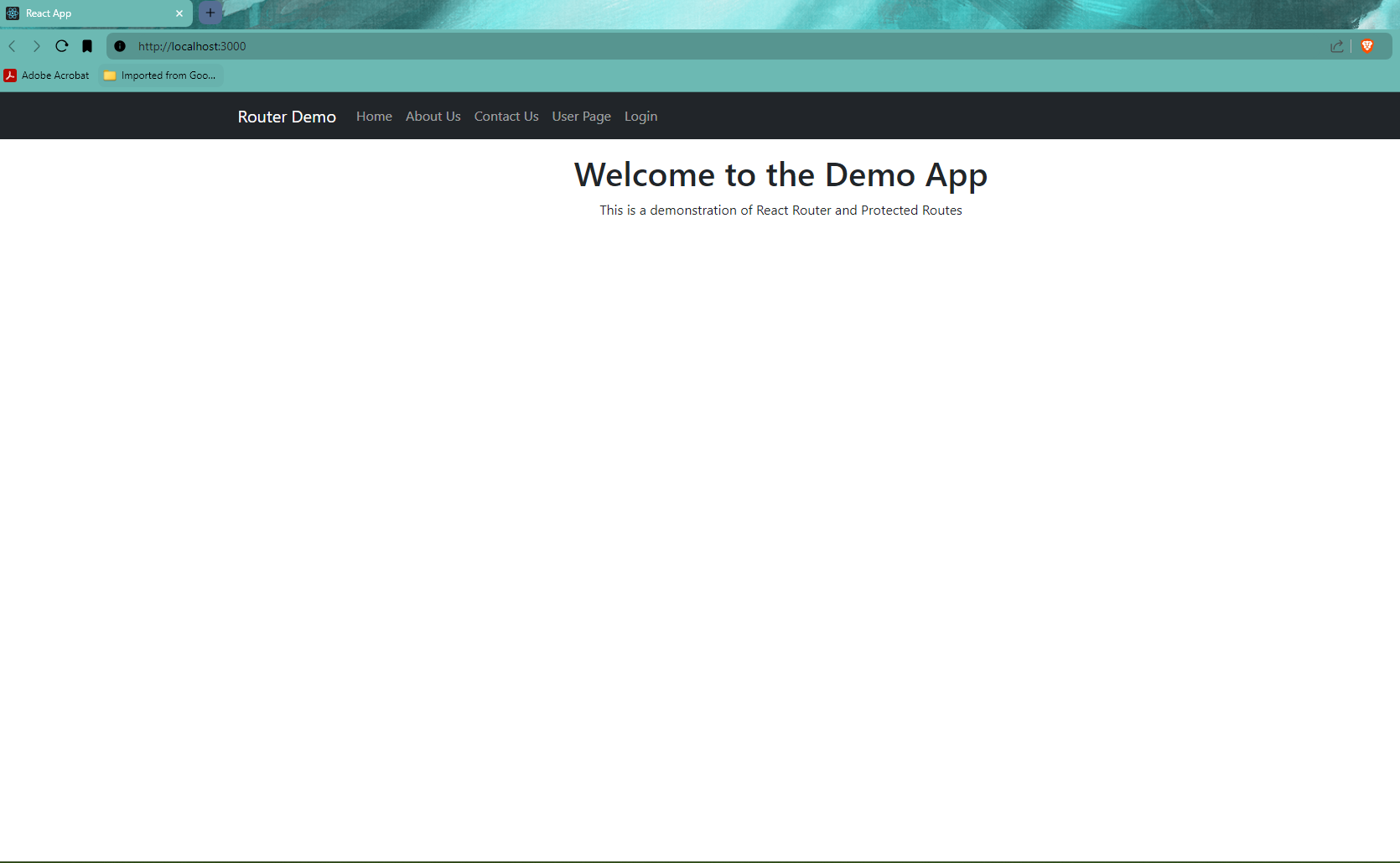
*Summary*

In this React version of the Music App, we implemented several significant features and modern React patterns. We migrated from static data to a dynamic Express backend API, utilizing **Axios** for HTTP requests to fetch album data. The application now includes search functionality that filters albums in real-time using both frontend and API-based searching. We implemented React Router (v6) for client-side navigation, allowing users to move between different views including a home page displaying all albums, individual album detail pages, and a placeholder for adding new albums. The app's architecture was improved by breaking down the UI into reusable components (**Card**, **SearchForm**, **NavBar**) and implementing proper state management using React hooks like **useState** and **useEffect.**

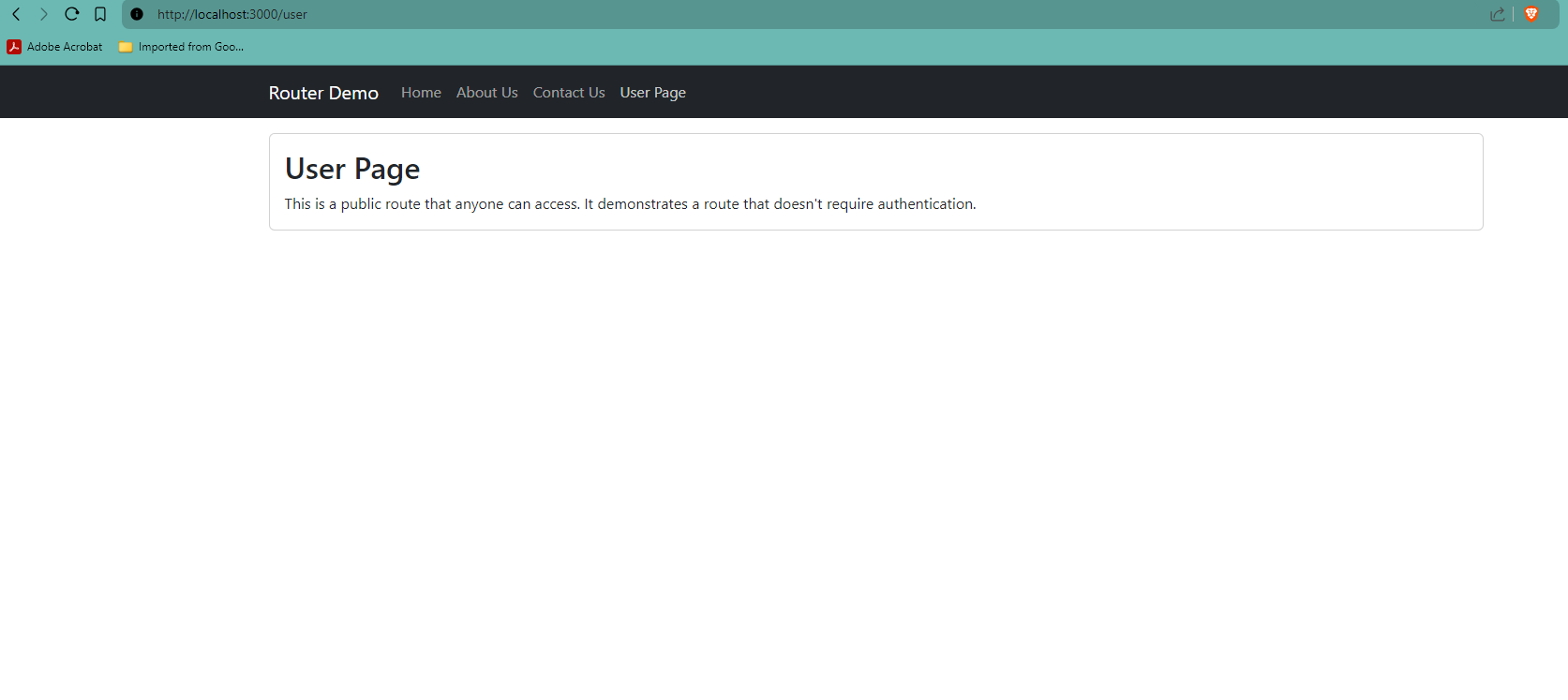
**Key terminology** introduced includes 'routing' (managing different views/pages in a single-page application), 'client-side navigation' (changing views without full page reloads), 'API integration' (connecting frontend to backend services), and 'component lifecycle management' (using **useEffect** for side effects like data fetching).

**Part 4: Navigation Routing**  
*Screenshots*

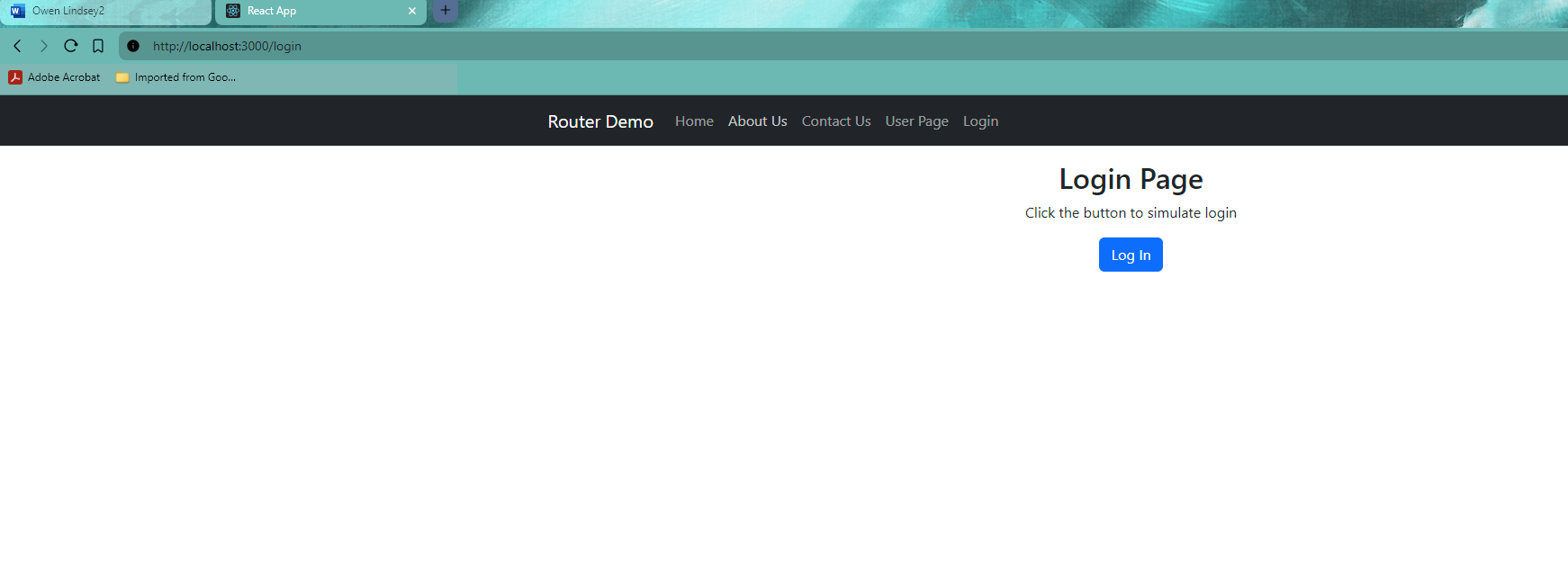
Home page of our demo application.



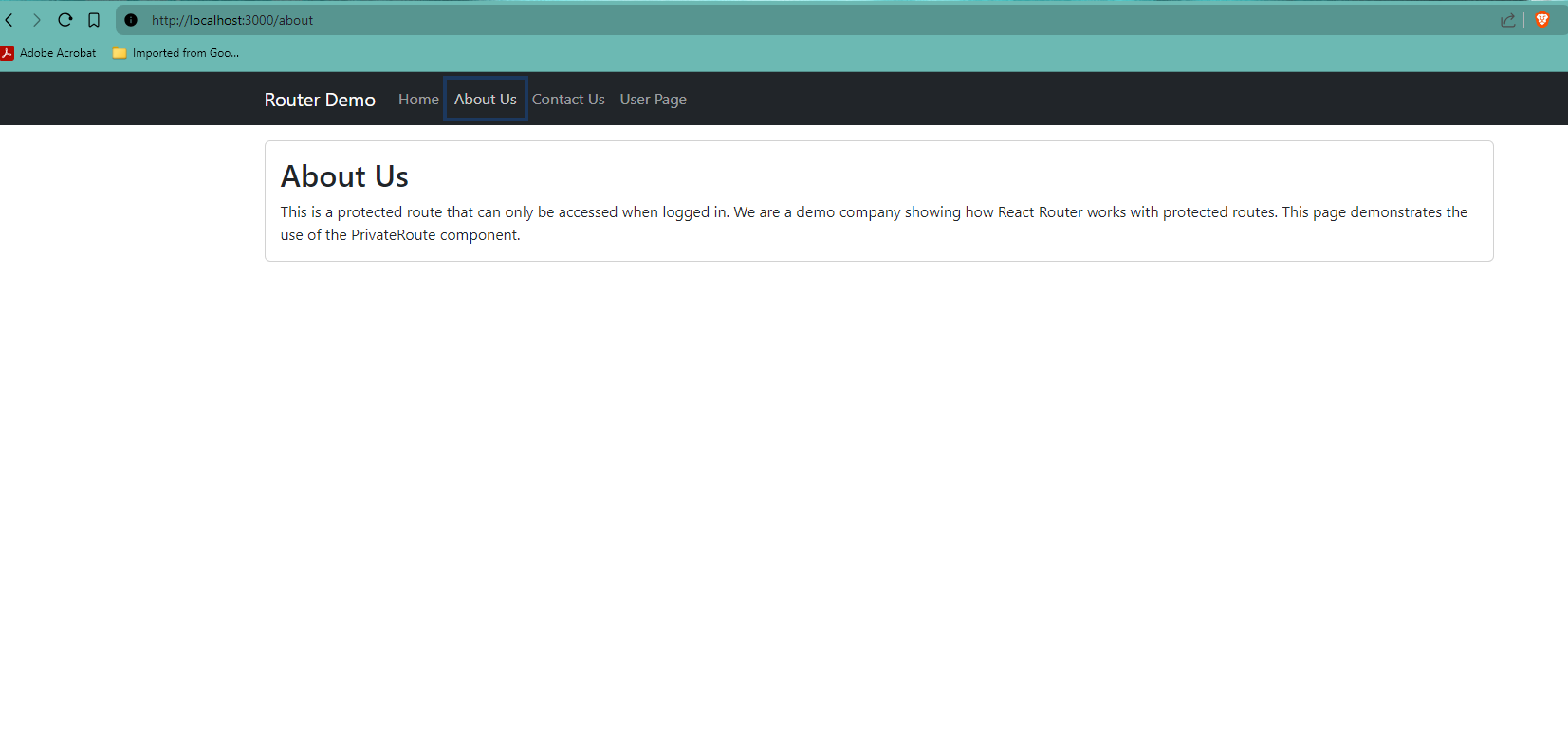
*User page is public, and anyone can access this page.*

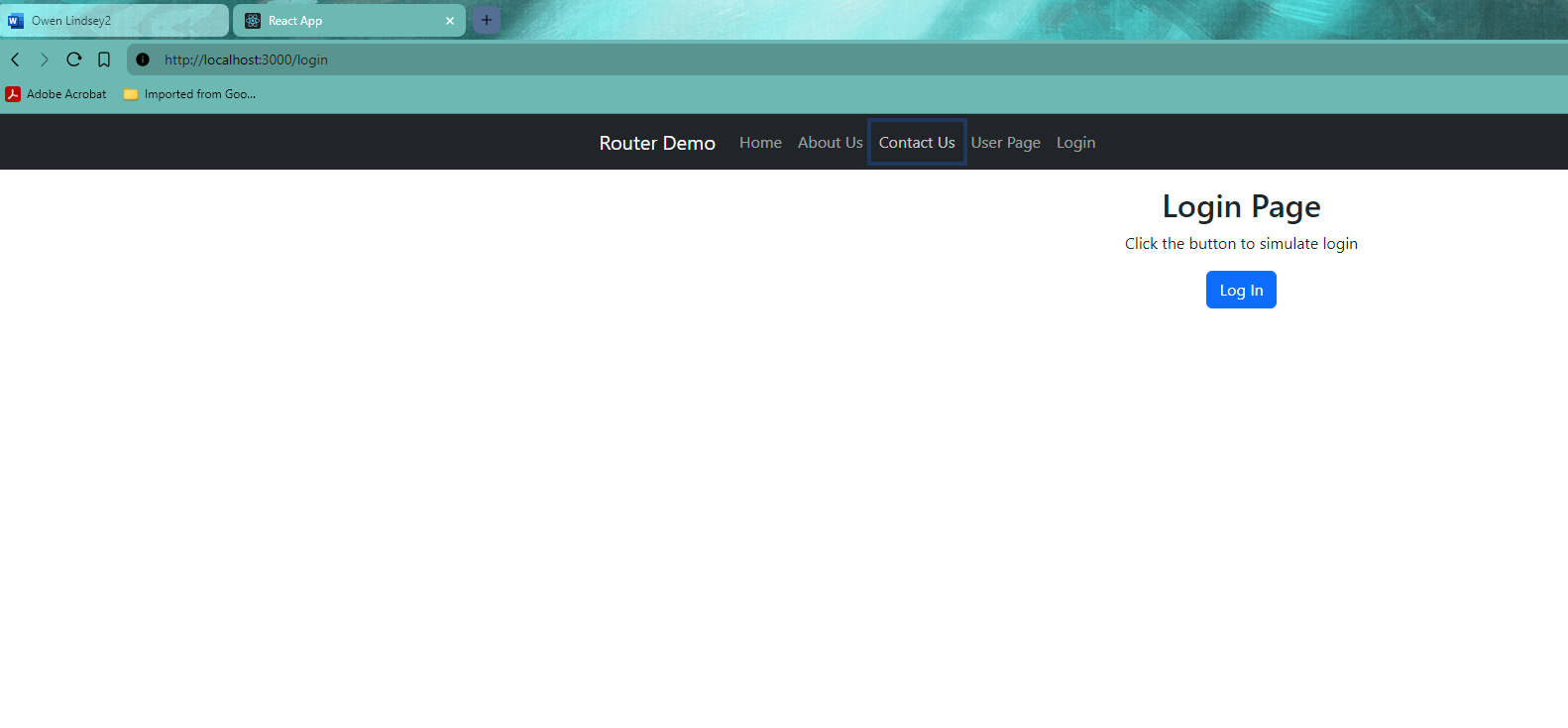


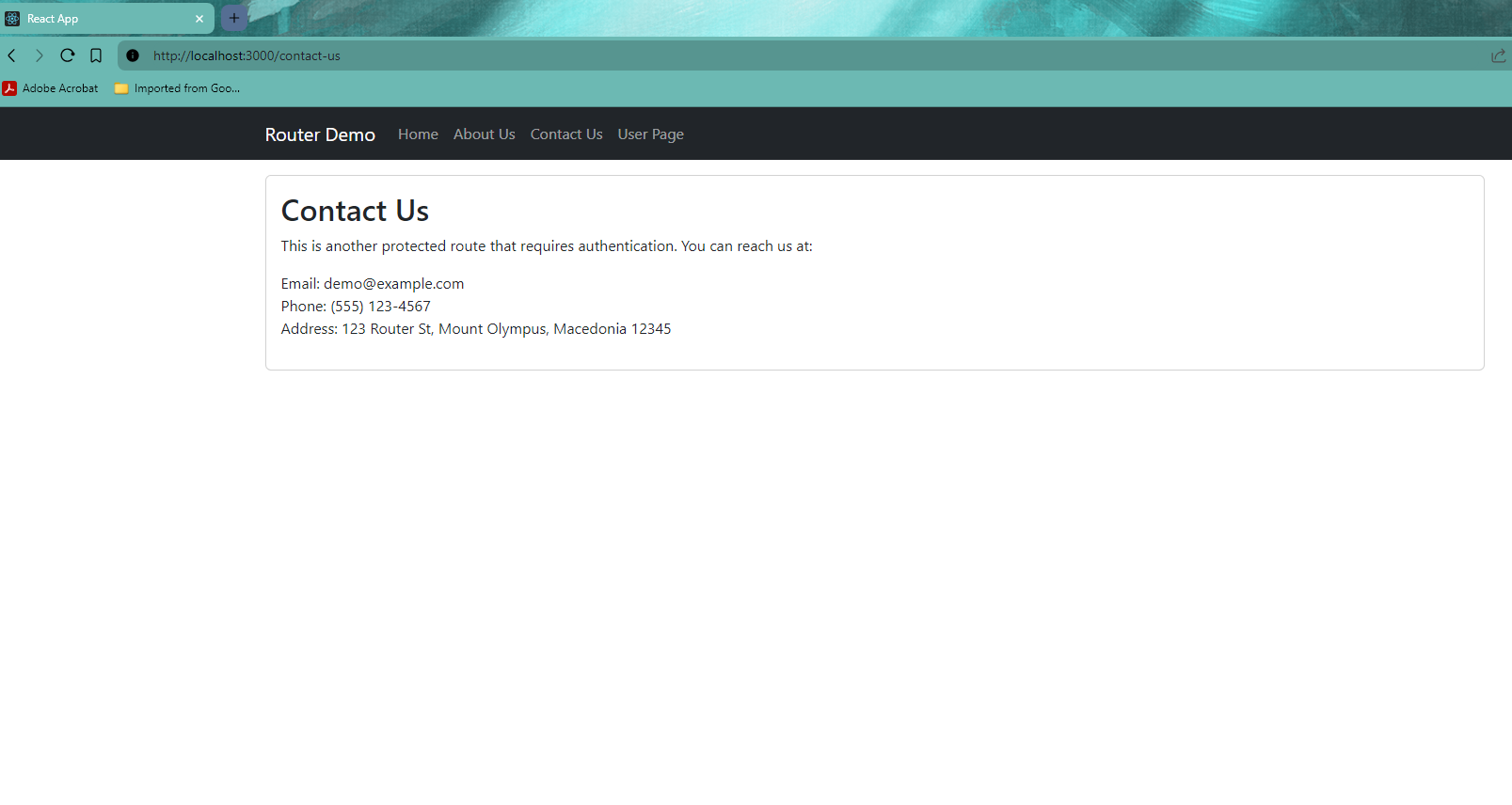
**Part 4: Navigation Routing**  
*Screenshots*

*About us page requires login to view full content (before login).*  


*About us after login.*



Contact us is also a protected page (before login).**Part 4: Navigation Routing**  
*Screenshots*

Contact us (post login).   


**Part 4: Navigation Routing**  
*Summary*

The Router Demo Application uses concepts of React Router and authentication flow in a single-page application. The app implements protected routes using React Router v6, where certain pages (About Us and Contact Us) are only accessible to authenticated users. We utilized several key React Router features including BrowserRouter for routing context, Routes and Route for path matching, Link for navigation, and hooks like **useNavigate** and **useLocation** for programmatic navigation and location awareness. The authentication system demonstrates a simple login flow where unauthorized access attempts to protected routes are redirected to a login page, with the original destination stored and restored after successful login.

**Key terminology** includes 'protected routes' (routes that require authentication), 'route guards' (components that control access to routes), 'navigation state' (storing and retrieving navigation history), and 'conditional rendering' (showing different content based on authentication status). The app also demonstrates proper component composition and state management patterns using React hooks.