**Activity 6**

Noah Funderburgh

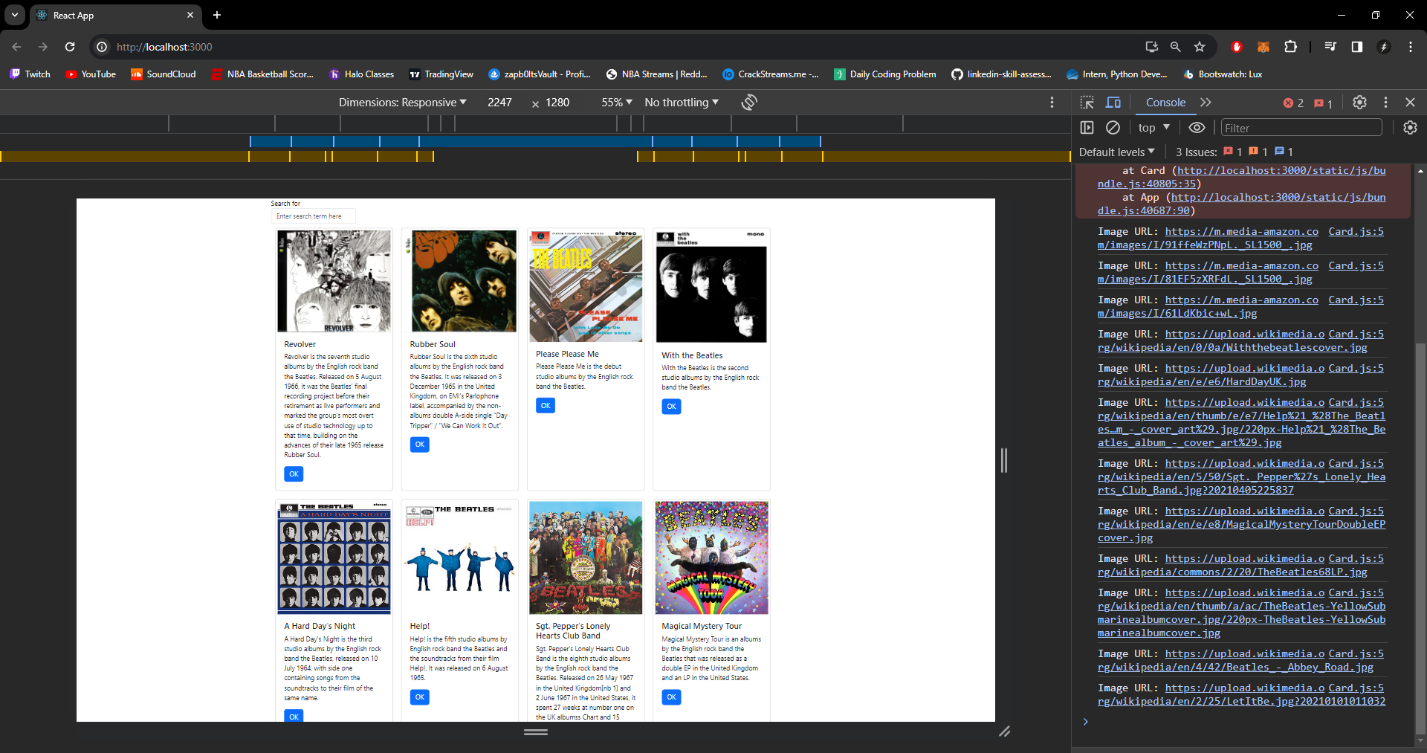
Grand Canyon University

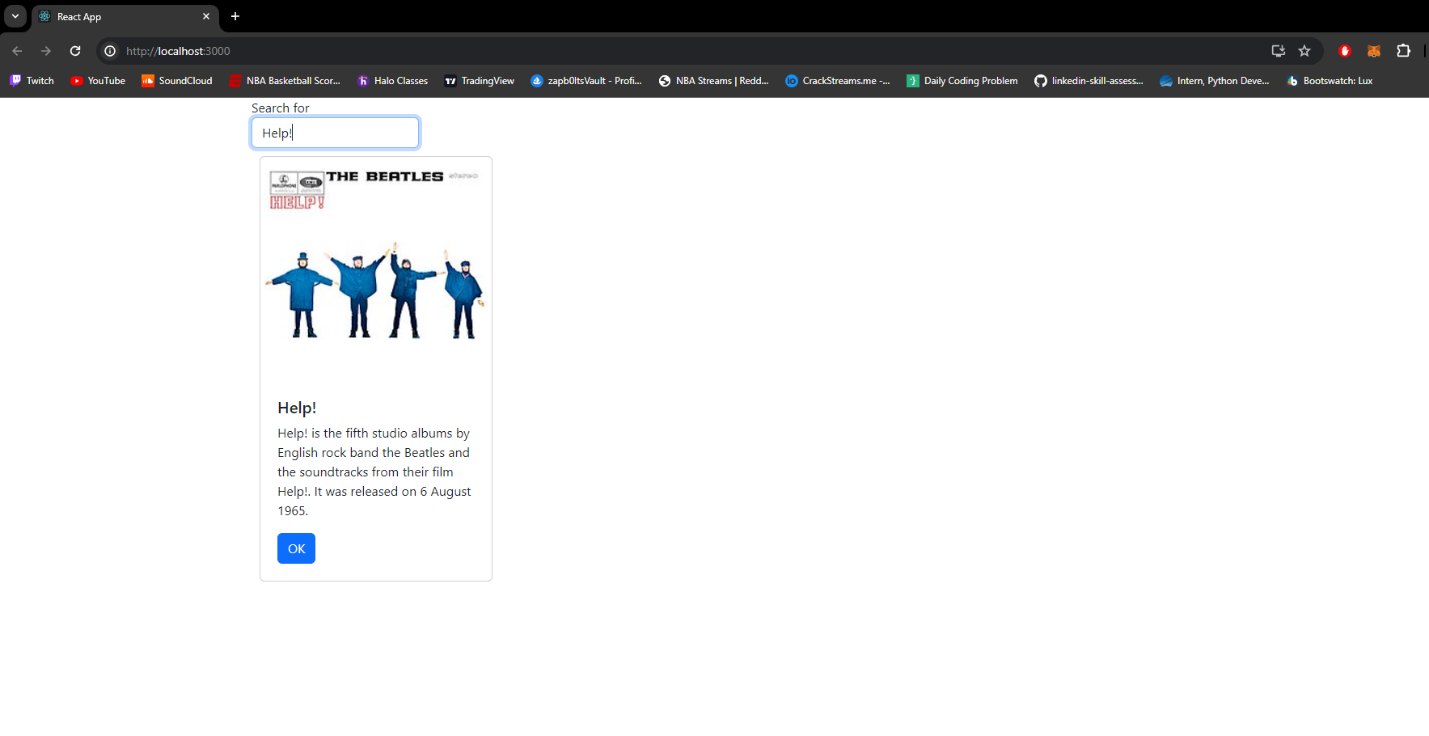
Course Number: CST-391-O500

Jason Jazzar

12/17/2023

**Stopping Point #3 – External Data Sources**

**Music App connected to MusicAPI and displaying all albums**

**Searching for “Help!” Album**

A screenshot of a computer

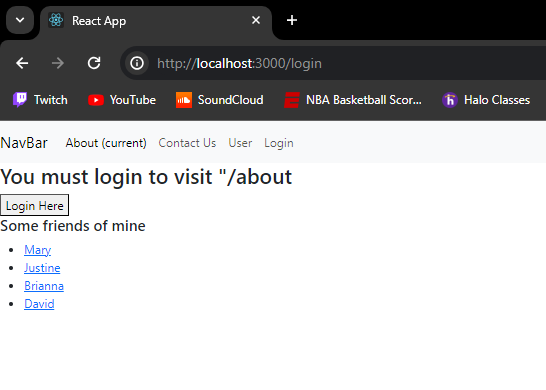
Description automatically generated**Showing console when searching for “Help!” Album**

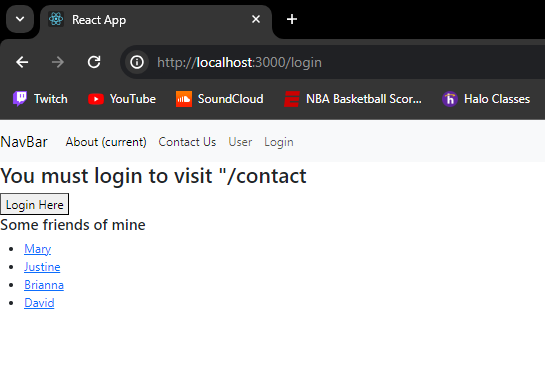
**Summary**

In this part of the activity, we managed our album data using our MusicAPI instead of using albums JSON. I built up and updated our initial album list using a hook to accomplish this. Next, to update and present the list, we utilized the useEffect function to set a new albumList state. We also included a brand-new search feature. This enables users to look up a term in our albums. To accomplish this, we had to develop a search Form component. The OnChange and OnSubmit handlers were used to show the component's current state. To transfer data from the SearchForm to the handler, we used callbacks. Lastly, we switched from our album.json to a fetch from our API using the Axios framework.

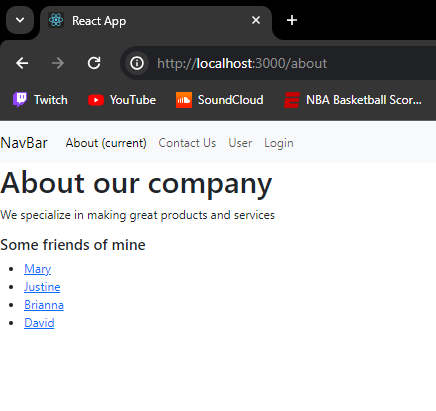
**Mini App #2 = Routing Application Demo**

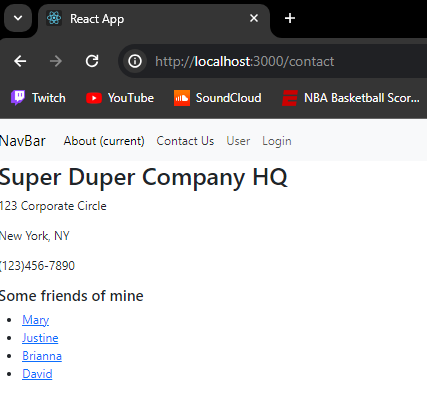
**Router App without logging in**

****

****

**Router App after login**

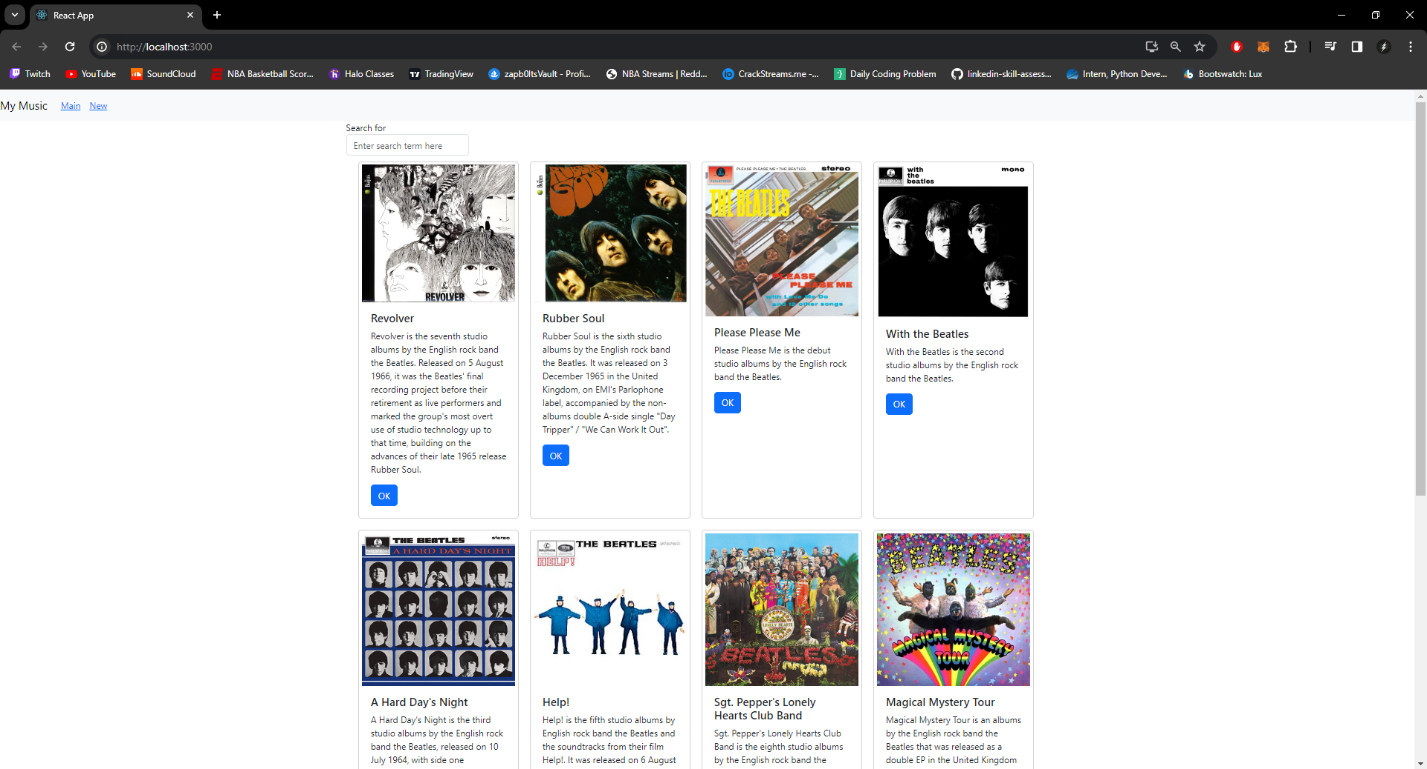
****

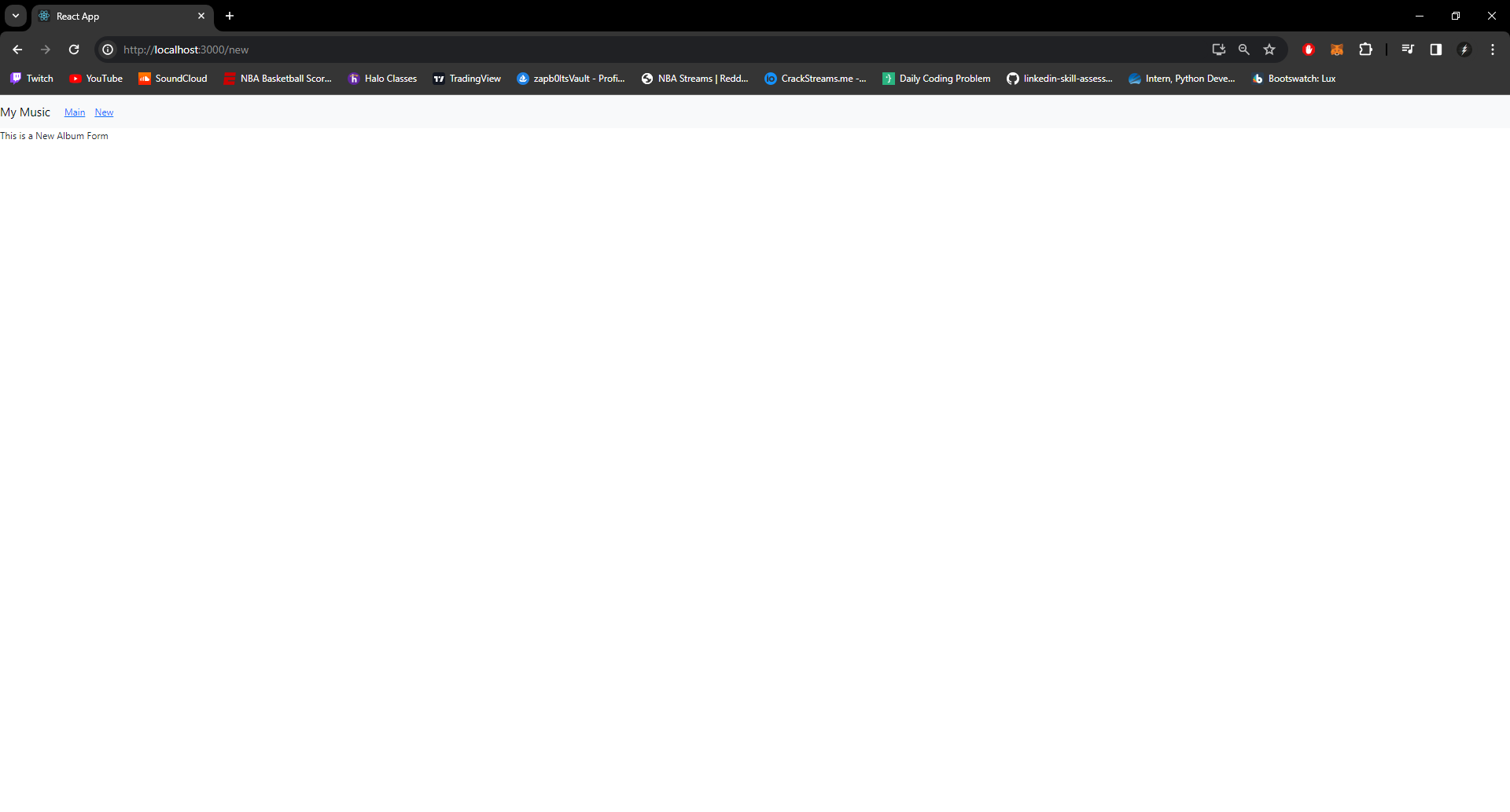
****

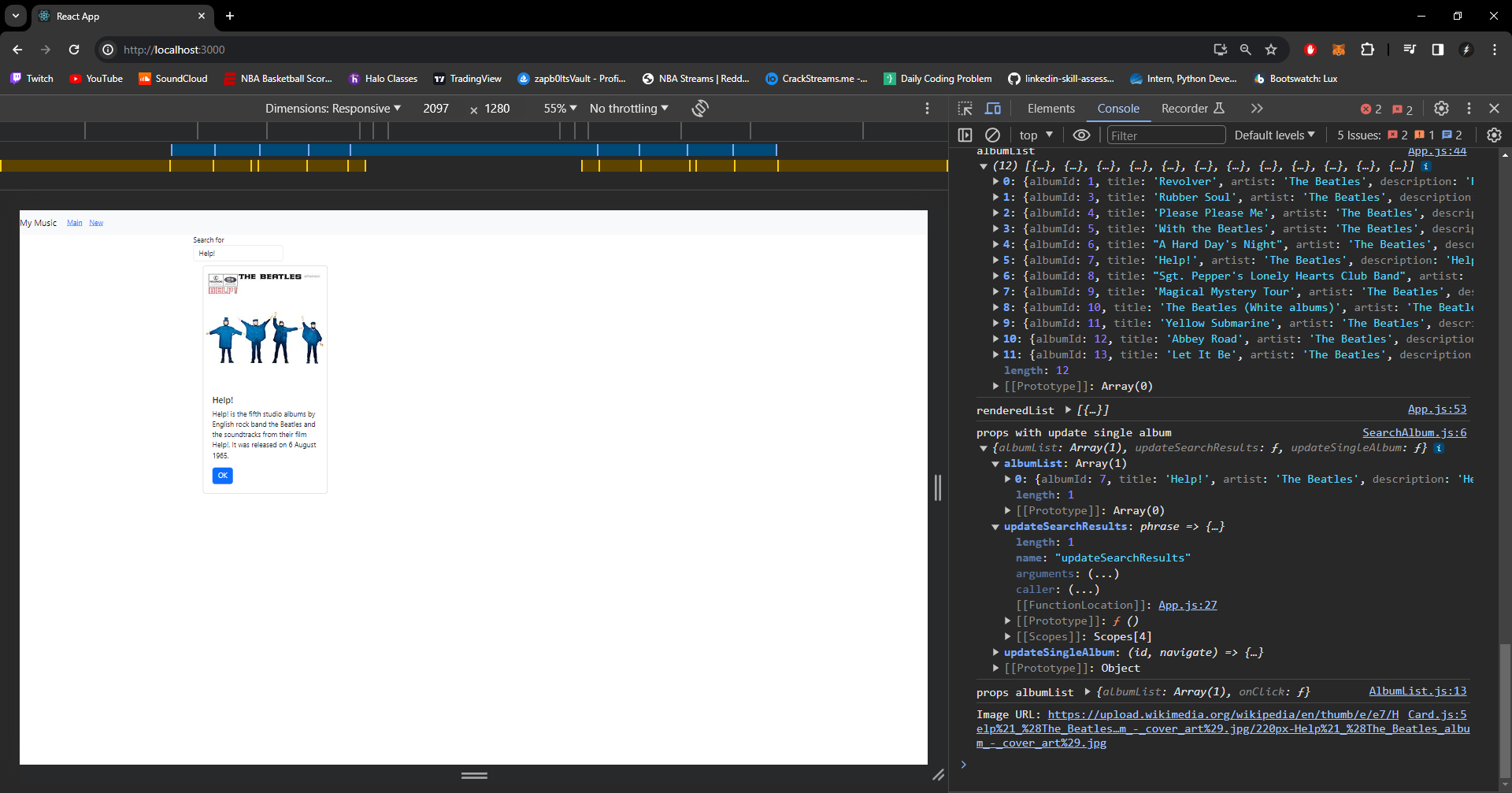
**Summary**

In this demo, we used React's routing framework to define our application's behaviors based on the URL entered in the browser. To begin with, a route is a link that exists between the URL of the browser and the web page content. To keep your UI in sync with the URL, BrowserRouter makes use of the HTML5 history API. All the other components are stored in this parent component. Initially, we developed an application consisting of four parts: About, Contact, Login, and PrivateRoute. We utilized the navigate method in our login component, which gave the code permission to use routing to go to the next application page and location to get the current location. Routes are safeguarded by the PrivateRoute to prevent unauthorized users are prevented from reaching protected routes. The components protected by PrivateRoute are AboutThisSite and ContactU

**Stopping Point #4 – Navigation Routing**

**Music App main page**

**Music App new page**

**Music App search filter**

**Summary**

We refactored our music app into a set of components in this activity. A new component, AlbumList, has been created to handle a list of Card elements. The SearchForm and AlbumList components are also part of a new component we made called SearchAlbum. Lastly, we changed App.js to incorporate SearchAlbum. The stateful component is still App.js, but most of the information is now contained in child components.