JS Jammming

**May 28th 2018**

**OBJECTIVE**

A simpler User Interface when pressing enter triggers a search and loading screen during the search. A description of the artist and biography with a play button on each track to play a sample of the song.

**BACKGROUND**

Currently, Jammming apps allows user to search for an artist and track without any further info. Then, with the search result allow user to add each track to playlist and add it to their Spotify. Jamming is good app but missing some mechanic to make it user friendly. It also lacks artist information such as bio and album images. General flaw is user blindly searching for a track without knowing who the band is or what they sound like.

This Feature accomplices the following:  
-  Pressing enter and click search activate button to make simple for user  
-  Loading animation during the search to let user know that its searching track from Spotify  
- Artist biography or Track information when click on track name and artist on track search result  
- Image of album or the artist should be shown when clicking on the track search result.

**TECHNICAL DESIGN**

**Press Enter to Retrieve and Display Playlist**We need to make simple for user to use the Jamming app.The search bar only works by clicking on search button. To make it user friendly. This search button should also function by pressing enter. We need to add a method to Searchbar.js Create handleKeyPress() method in SearchBar classwith event. It will have an if/else statement when event press a key equal to enter it will run this.search() method in SearchBar. The handleKeyPress Method will be bind to SearchBar constructor. This will be render in the input of div SearchBar. This will on key up. In input,  we will have key of onKeyPress with value of this.handleKeyPress.

**Adding Loading Animation in CSS**First, we need to create a loading animation using css. This css class will create a spinning animation. First, createclass loading to be styled in app.css. This class will have an endless looping animation. This will include a webkit animation  called sk scaleout 1.0s and infinite ease-in-out. Then using keyframes, applied to sk-scaleout with transition from 0% to 100%. At 0% keyframes, the –webkit-transform scale at 0 and transform scale is 0. Next at 100% keyframes, the –webkit-transform scale at 1 and transform scale is 1. Here is the example of css style animation.

With the loading css animation, We will need to apply this to the component in app.js. First, we will have a div with a id of spinner and inside the spinner div we will create another div with a class of loading. Now, we need to render the function. In render, we create a constant variable called spinner. It will equal to document and get Element by id of Spinner. Next, We will add an if statement. If spinner is true and  attribute is not set to hidden, we will set hidden attribute to spinner id to true.

Running app.js should mount correctly. The spinner should be hidden first, applying the hidden attribute to it. It should add the functions condition which prevents to add a hidden attribute to spinner. It should be added when the app loads.

**Retrieving and Displaying the Artist and Track Information**A new component will be added called ArtistInfo. This component, on render will grab the information of artist on current search result of the track. We need to update the state of the searchResults and playlistTracks. We can create a mock sample to test app. For example, each track will grab artist information.

searchResults: [  
     {name: 'Common', artist: 'Common', album: 'Be', id: 1 bio: ‘Common is an American rapper, actor, poet, and film producer. He debuted in 1992 with the album Can I Borrow a Dollar.’ } ]

In order to retrieve the Artist information, We need to update the search method in Spotify.js. We will update the search method jsonresponse when map the track return. After track.uri, We will add comma and add key of artist bio. Next, add colon with track.href. This is artist information from Spotify request format to grab artist information. In addition, We will add link to artist website by artist external link. After track.href we will add a artist site colon and track.external.urls. This will grab you the artist own website from Spotify. Here is documentation use for response format from Spotify.<https://developer.spotify.com/documentation/web-api/reference/tracks/get-track/>.

if(jsonResponse.tracks) {  
    return jsonResponse.tracks.items.map(track => ({  
       id: track.id,  
       name: track.name,  
     artist: track.artists[0].name,  
             album: track.album.name,  
       uri: track.uri,  
       bio: track.href,  
       artistsite: track.external\_url  
     }))  
  
With this Spotify.js updated, We can now retrieve artist information data as well as the external link of the artist website. We need to update the track.js. When tracks is being map, we add two keys: bio and artistsite during the Tracklist render(). This help post the information and links available when the track are map from search results. Same can be applied to album images. In Spotify response format, We can add new key that grab album cover images.

     <div className="TrackList">  
     {this.props.tracks.map(track => {  
         return <Track track={track}  
                      key={track.id}  
                             name={track.name}  
                 artist={track.artist}  
                           album={track.album}  
                           bio={track.href}  
                           artistsite={track.external\_url}  
                    onAdd={this.props.onAdd}  
                             onRemove={this.props.onRemove}  
                           isRemoval={this.props.isRemoval} />  
          })}  
        </div>

**CAVEATS**

**-webkit- animation limits with andriods**There’s some issues with loading animations. There’s some adroid app that does not support the webkit animation. Alternatively, if there is issue webkit animations in css, We can use images or icon to use for loading screen instead of using a loading animation. In addition, there react supported animation to use commercially.

**Lack of Information for new artist**There are common issues with newer artist such lack of information from Spotify. Sometimes, artist doesn’t have external link. We can use their social media link for info such as  twitter, facebook, and youtube. We can use Wikipedia api if user need more in depth information for artist and discography.

https://docs.google.com/document/d/13U5EAhzlqEWFjSMRVfiYEX-KtXCDTB28CvNmAk6bguc/edit?usp=sharing