Feature Request: Search by Album, Artist or Song

**May 8th, 2025**

**OBJECTIVE**

Give the user the ability to search for songs by album, artist or song.

**BACKGROUND**

Currently, Jammming supports the ability to search for songs by the track name, add them to a playlist, login to spotify and save that playlist to their account. Spotify itself has the feature to search by artist, albums and songs and I find myself searching by artist very often as well.

This feature will be able to:

* Search by artist, song or album when the user selects the parameter from the dropdown list and enters information in the search field
* The results field will population with relevant artists, albums or songs
* If the user is searching for artists or albums, the user can click the album or artist and it will take the user to their respective tracks so the user can add the invidual tracks to a playlist
* If the user changes the input in the search field it will redirect the user back to the main results

**TECHNICAL DESIGN**

There will be three new components. One will be **Albums**, this will display the albums tracks when a given album is selected. Two will be **Artists**, this will display the artists tracks when a given artist is selected. Three will be **Songs**, this will display the songs when a given artist and will replace the **SearchResults** component.

A dropdown list next to the search bar will show the options of song, album and artist. This will be stored in a state variable named **searchParam** which will be triggered in an onClick event. The searchParam variable will have a default value of "track". When one is selected it will populate our endpoint URL with said selection in a params string.

To retrieve albums, we will modify our existing **getSearchData()** function to input the selected search parameter in our current endpoint with &type={param}. This endpoint will require the access token. This function will also need to set albumId and artistId to an empty string.

We already have an access token available for re-use stored in the state variable **accessToken**. We will use this token and not loginAccessToken so we can still search without having to log the user in.

A new state variable will be named **results** and the setter to **setResults** and eliminate the **handleData()** function and the **songs** state variable. We will then pass the results variable to our components and handle the data directly with dot notation on the results objects. We will also need a **getAlbum()** and **getArtist()** function that will be triggered in the onClick event in our **Album** and **Artist** component.

We will also need two other sate variables named **albumId** and **artistId**. We will pass their setters to their respective components as props and change our + button to an > instead on the Albums and Artist components. When the user clicks the > for a given artist or album the setter function will be called and will pass that artist or album Id back to our main App component.

The **getAlbum()** function will take the albumId variable and pass it into the endpoint [https://api.spotify.com/v1/albums/{albumId}](https://api.spotify.com/v1/albums/%7BalbumId%7D). It will then set the results state variable to the retrieved data.

The Album and Artist component will have a ternirary operator that will render that data differently if albumId or artistId return a truthy value.

**CAVEATS**

This section is used to lay out alternative solutions and their respective drawbacks, as well as potential drawbacks to the proposed solution above. This is used to make it clear why the technical implementation detailed previously was chosen instead of alternatives. It additionally allows stakeholders or other developers to consider those drawbacks and choose one of the alternate solutions if they prefer it. This may occur if they feel the benefits or drawbacks of that solution are more desirable than the current solution, or if they can identify other benefits and drawbacks not currently listed.