Feature Request: Capturing the Search Term **January 12th 2019**

OBJECTIVE

To improve user ease-of use and satisfaction by eliminating repetitive entry of the search term when the app is required to ask Spotify for an access token.

BACKGROUND

The Spotify API requires an active user access token in order for our client, Jammming, to conduct business with the API on the user's behalf. To achieve this, the app redirects to the Spotify API to request the token. The Spotify API then redirects back to our web app.

Currently, one of the resulting effects of this round trip is we lose any user input (in this case, the search term) from prior to leaving our app. Therefore the user is required to remember the search term they entered previously and re-enter it.

As this feature request falls under the category of ease-of-use or customer satisfaction, it should be strongly considered.

This feature accomplishes the following:

- If the app determines it needs to leave the site to request an access token from Spotify, it stores the user's input prior to exiting the app.
- Automatically re-enters the user's input after return to the app.

TECHNICAL DESIGN

Saving the Search Term

Upon clicking the search button and prior to potentially exiting our app, we will need to persist the user's search term.

In *Spotify.js*, we will modify the **.search()** method to store the user's search term using the **.setItem()** method of localStorage to add a key/value pair for the search term. We will do this immediately within the .search() method as this needs to occur prior to any call to **.getAccessToken()** within the .search() method.

Recalling the Search Term

When our web app loads, we need to check if we are actually returning to the app from the Spotify API and have a search term in storage which we need to retrieve and display.

In *SearchBar.js*, we will add a **.componentWillMount()** lifecycle method to the SearchBar component which will check localStorage for our 'key'. If found, we will set the state of the component's 'term' key (using setState()) to reflect the 'value'.

In the SearchBar component's **.render()** method, we will add a 'value' attribute to the input element which will display the retrieved search term (this.state.term), if any. Otherwise, the placeholder attribute's value will display.

Clearing the Search Term

Since we are storing the search term everytime we click search, we need to garbage collect and clear the search term after each search completes.

In *Spotify.js*, we will modify the .getAccessToken() method in two places:

1. In the if condition where we already have an access token;
2. In the if condition where the access token is in the URL.

In both locations, we will use the **.removeItem()** method of localStorage to delete our key/value pair.

CAVEATS

Data Storage Techniques

Based on limited research, there are several storage options available for capturing and persisting user input:

- Databases - viable but overkill for the change we are making.
- Cookies - requires server side
- Session storage - does not persist beyond exiting the current domain.
- Local storage - our choice as this persists until cleared and is accessible to the originating domain only.

Falling Short on User Satisfaction

As currently envisioned, this feature falls short of the stated objective of improving user satisfaction. The user action that triggers the trip to the Spotify page and back is the clicking of the search button. If we intend to completely alleviate repetitive actions (re-entry of user data **and** re-clicking of the search button), both actions should be addressed and corrected in the same feature request.