Feature Request: Follow an Artist.

**February 4th 2019**

# **OBJECTIVE**

To give the user the ability to follow artists or user.

# **BACKGROUND**

Jamming supports the ability to generate a manual playlist, but users are not notified as new tracks and albums are added to an artist’s profile.

* This feature will allow users to follow an artist or user so they are notified about new music that becomes available in the Spotify application.
* They will be notified in the Spotify application or if their account is configured for email notifications they will receive notifications via email.
* Once notified about new music, users can use the existing functionality of the Jamming application to generate a new playlist to include the new tracks.
* Users will also have the ability to unfollow artists and users.

# **TECHNICAL DESIGN**

Retrieve and Display Followed Artists and Users:

A new component, **Following** should be created. This component will render and retrieve a list of Artists or Users currently followed by a user.

The **Following** component will need to be initialized with an empty array of Artists and Users.

The base URL for the follow API is <https://api.spotify.com/v1>

To retrieve the list of followed artists and users we will create a new method called **Spotify.getFollowed()**, that hits the <https://api.spotify.com/v1/me/following>? Endpoint. There is one required parameter *limit* that identifies the number of items to return. In this application we will implement the default of 20 items to return.

We will utilize existing functionality in this request to check if a **userID** is already set and return a promise required by the Spotify API. We will additionally need to create a new component, **Artists** that will be used to render the followed artists and users in the *Following* component.

The required query parameter is the artist or users Spotify id. The Spotify ID is a base-62 identifier that is found at the end of the Spotify URI.

In *Spotify.getFollowed()*, once the user ID has been retrieved, we will make a call to the /v1/me/following/contains endpoint to see if the user is following artists or users. If not we will return a message that there are no artists or users followed.

If yes, then we will implement a call to the /v1/me/following?type=artist endpoint to retrieve the list of artists followed. We will populate **Artists** with the returned values.

Similar to the playlist, we will utilize the SearchBar and TrackList components and modify them to include a button to follow an artist or user. The new functionality will use a *Following* component render a *Playlist* component with the followed artists and users.

We will create a new method called *addArtist* for moving artists and users to the *Following* component and adding them to the *Artists* array. Once the *Artists* array is populated with selected artists or users, the array will be utilized to upload the list to Spotify.

To put the followed artists or users in the Spotify app we will implement a new method called **Spotify.putFollowed()**, that hits the <https://api.spotify.com/v1/me/following> endpoint to follow an artist or user.

In *Spotify.putFollowed()*, once the user ID has been retrieved, we will make the API call upload the list of artists or users selected to be followed.

In order to remove followed artists or users we will utilize the *removeTrack* method for removing artists or users.

To delete the followed artists or users in the Spotify app we will implement a final new method called **Spotify.deleteFollowed()**, that hits the <https://api.spotify.com/v1/me/following> endpoint to remove a followed artist or user.

We will create a new method called *removeArtist* for removing artists and users from the *Following* component and removing them from the *Artists* array. We will also implement a new array called *Unfollow* that will be utilized to upload the list to Spotify.

In *Spotify.deleteFollowed()*, once the user ID has been retrieved, we will make the API call unfollow an artist or user.

# **CAVEATS**

With the new functionality we are utilizing similar add and remove methods used to create the *Playlist*. It is possible to modify those methods to add and remove both *Tracks* and *Artists*.

Additionally we could re-use the *Track* render code to also render the *Artists* for the *Follow* component.

The modification of existing methods would simplify the implementation.