

**Project Name:** Rhythmic

Submitted By: Yashbeer Singh (1910990251), Anuttar Jain (1910990244),

Saumya Singh Bisht (1910990234)

Submitted To: Mr. Amit Sharma

# INTRODUCTION

Music is a vital part of daily living, as Albert Einstein stated, “**life without playing music would be inconceivable**”. There is no one living in this earth who does not listen to any kind of music.

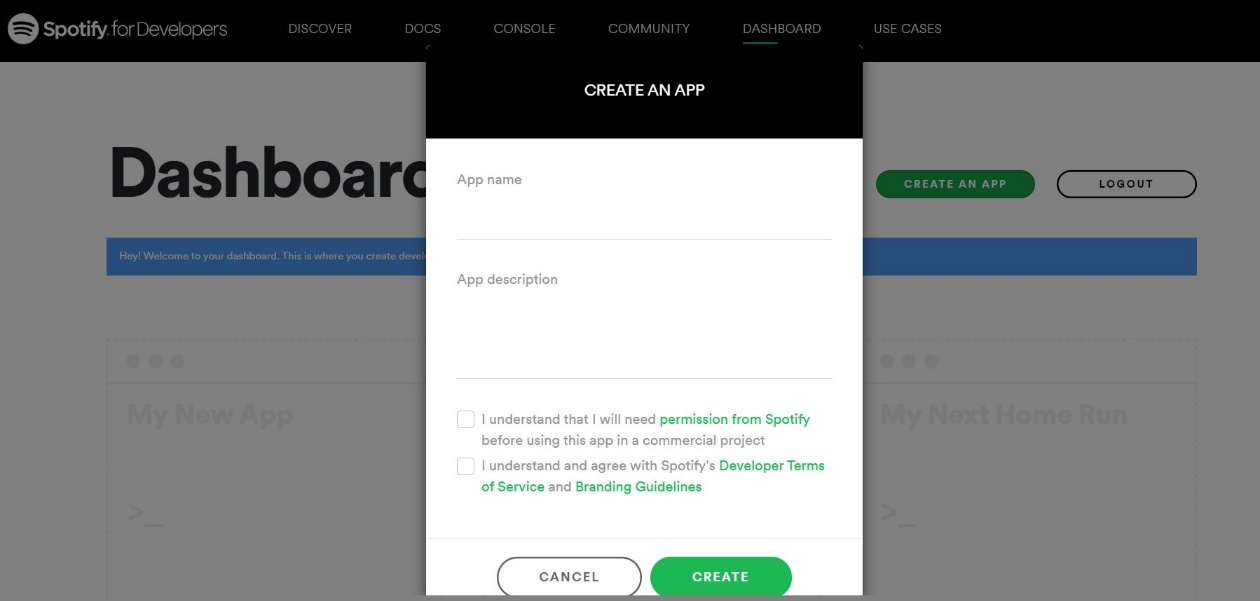
Music may be played or sung and heard live at a rock concert or orchestra performance, as part of a dramatic work such as an opera, or it may be recorded and listened to on a radio, MP3 player, CD player, smartphone or as a film score or TV show. The **present generation** **is witnessing millions of new songs and videos** coming out each day. **People who are very fond of music have to download them from the internet, which ultimately waste their device memory**. This **ultimately inspired us** to do something to solve the issue.

**In order to solve the problem of large required memory** of mobile phone or music player on the current market, **a new music player of simple, convenient, less required memory as well as user-friendly is developed**. Based on the **Web-technology**, using **React Js, SpotifyAPI,**  **and other programming libraries** lead to design and coding of this **music player,** which we prefer calling it as **Rythmic**.

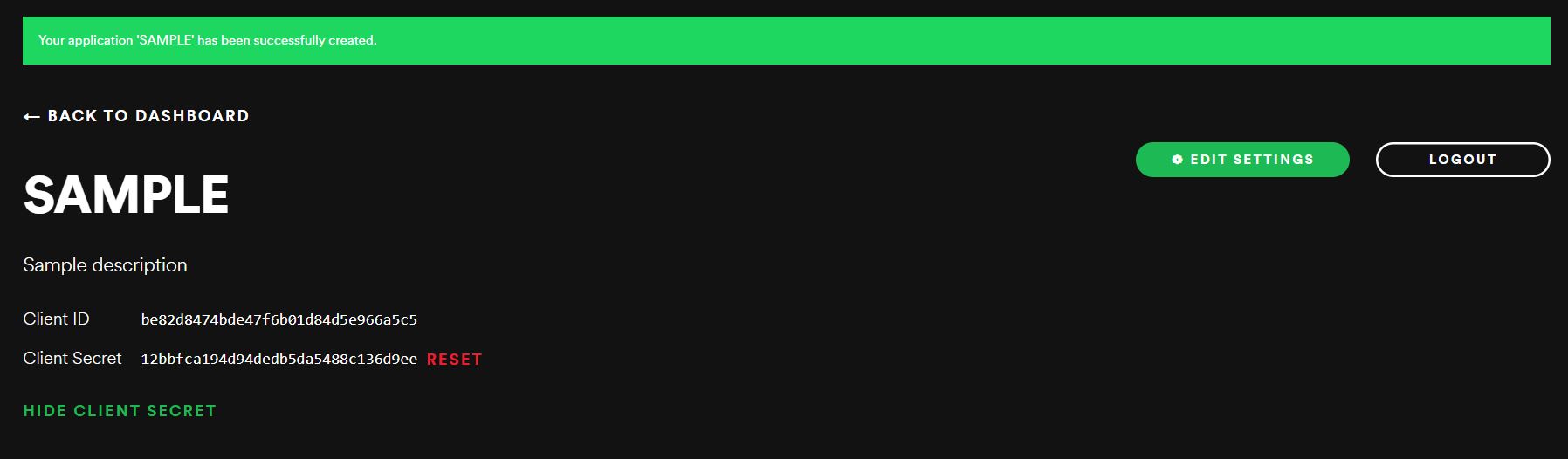
The new design mainly realizes **three core functions** including **main play interface, user playlists and dynamic song search.**

## Design ProCESS

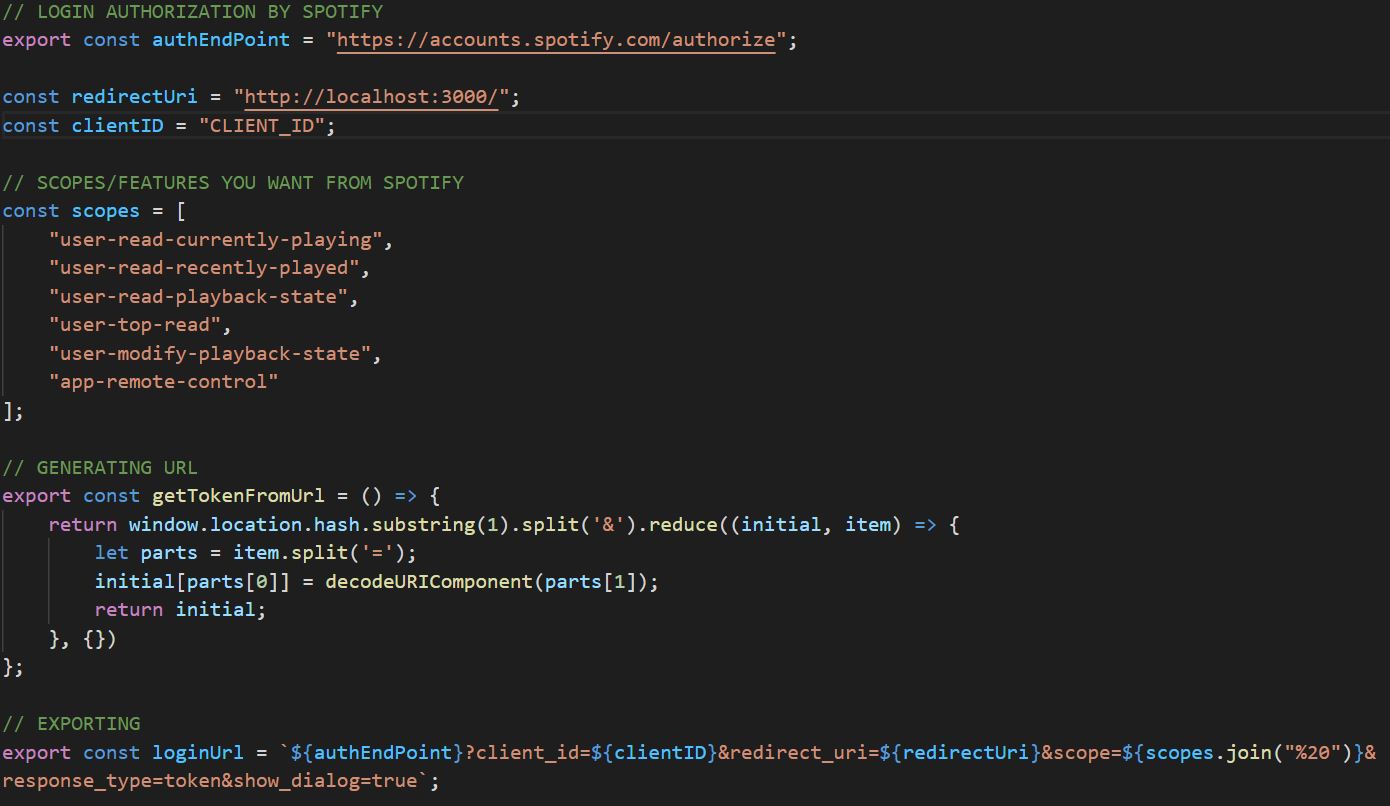
1. **Generating ClientID using the Spotify API:** We have used Spotify API as the main tool of our project. Follow the steps below.
   * Click on <https://developer.spotify.com/dashboard/> and login with your credentials.
   * Click on “CREATE AN APP” floating on the right and fill the details as demanded. Look into the screenshot for better understanding.



* + You will get your ClientID, save this for future use.

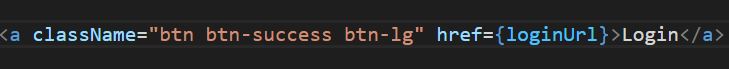


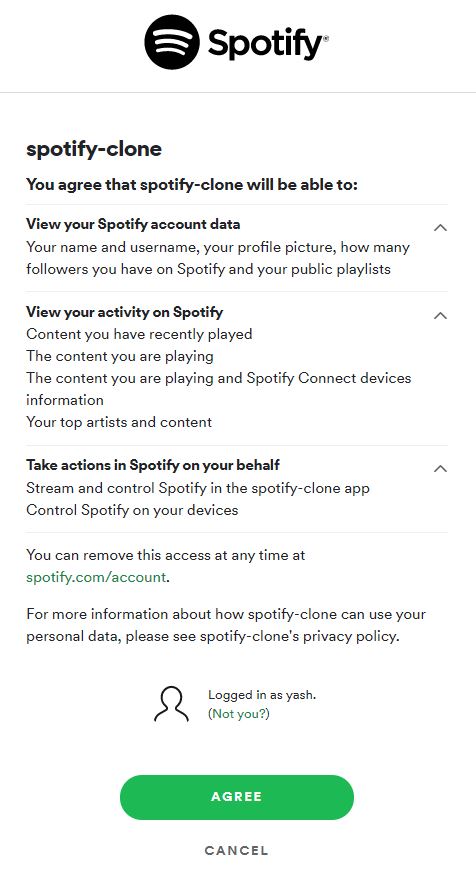
* + The following snippet will help in connecting to the spotifyAPI. You can use **loginUrl** to get through the authorization page of spotify.

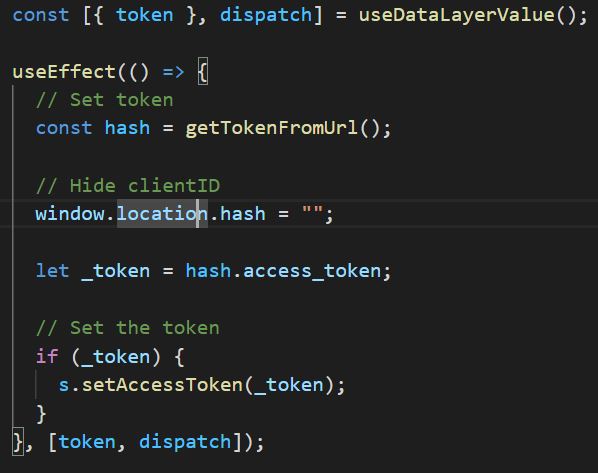


1. **Implementing LOGIN page:**

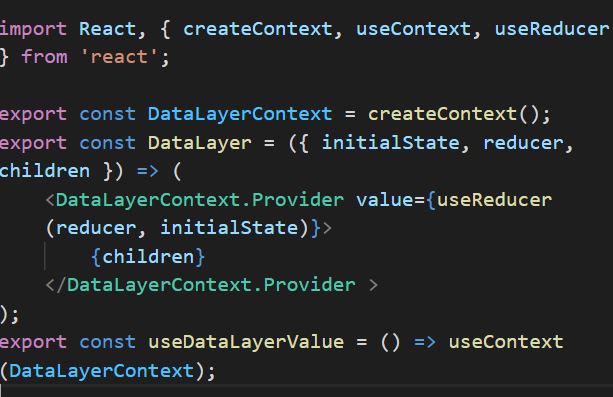
The login Button is just an anchor tag styled like a button which on clicking **gets through the authorization process** of the spotifyAPI. **If authorized** successfully, it **generates a token based on the scopes mentioned in the ‘scopes’ variable and redirects back to the localhost**. But, instead of displaying the LOGIN page again**, it shows MAIN page which is nothing but a hidden component of LOGIN page**. This **simulation is because of the token** which is generated in the process. When the user is **not authorized, token is null**, so, **component LOGIN is displayed**. Likewise, **if a user is authorized, token is generated so component MAIN is displayed**. This functionality is possible through **useEffect** hook. **Using the same token, we also fetched the user’s playlists and related list of artists that user sees right after logging in**. Below are the attached snippets for better clarity.





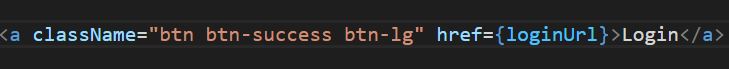


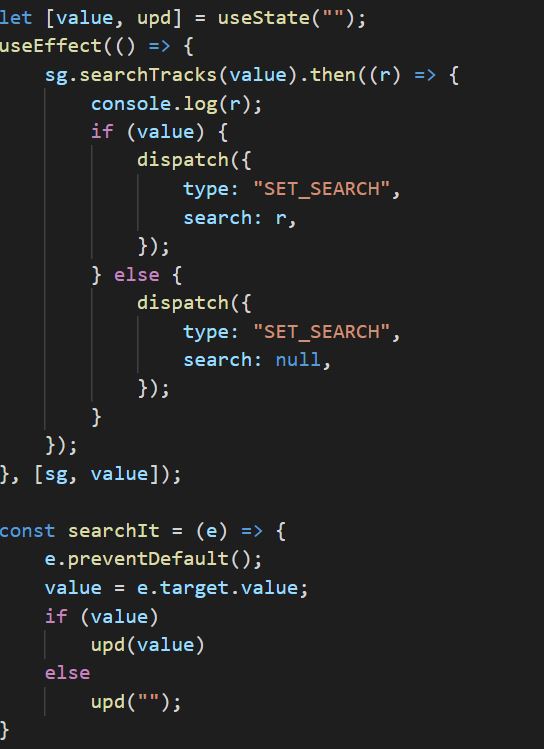
1. **Implementing the MAIN Page:** Instead of passing the data from props, **we defined our own DataLayer (just calling it that way, it is actually a global storage area) using the functionalities of createContext, useReducer and useContext**. Whatever information was needed to passed to another component, just dispatch it into the DataLayer using **useReducer** and grab it when required.

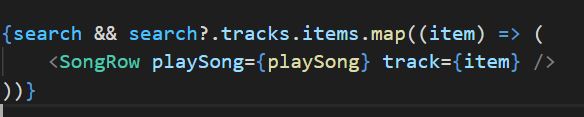
****

## 

1. **Implementing Search-bar:** The **search actually fetches the results from the spotify’s list of pre-recorded songs and displays on the screen**. As soon as the user starts to type on the search box, the input is validated and passed to the **searchTracks()** function available with the spotifyAPI. This function **returns a promise** which is **dispatched into the DataLayer**. Now, this **object is just displayed using map function**. The **data gets refreshed** each time user type something, this is achieved using **useEffect and useState** hooks respectively. Following are the screenshots.

****

****

****

1. **Fetching User Playlists & Switching between them:** Whenever token is not null, **user’s account details, playlists and artists are fetched from the spotifyAPI** using the **getMe(), getUserPlaylists() and getMyTopArtists()** functions respectively. These functions return promise which are finally dispatched into the DataLayer and used where required. Refer to the following screenshotsfor better clarity.

****

1. **Playing Songs and handling the main play Interface:** This part is the most essential component of Rythmic. Using **play(), pause(), skipToNext(), skipToPrevious(), getMyCurrentPlayingTrack() and setVolume()** functionsof the **spotifyAPI,** the promise involving the **currentState** is returned by these functions are similarly pushed into the DataLayer from better user experience. Below the screenshots for referral.









## TEAM CONTRIBUTION:

The **LOGIN** page right from its designing till the implementation was completely handled by **Saumya Singh Bisht.** The **MAIN** pagewas equally handled by **Yashbeer Singh and Anuttar Jain.**

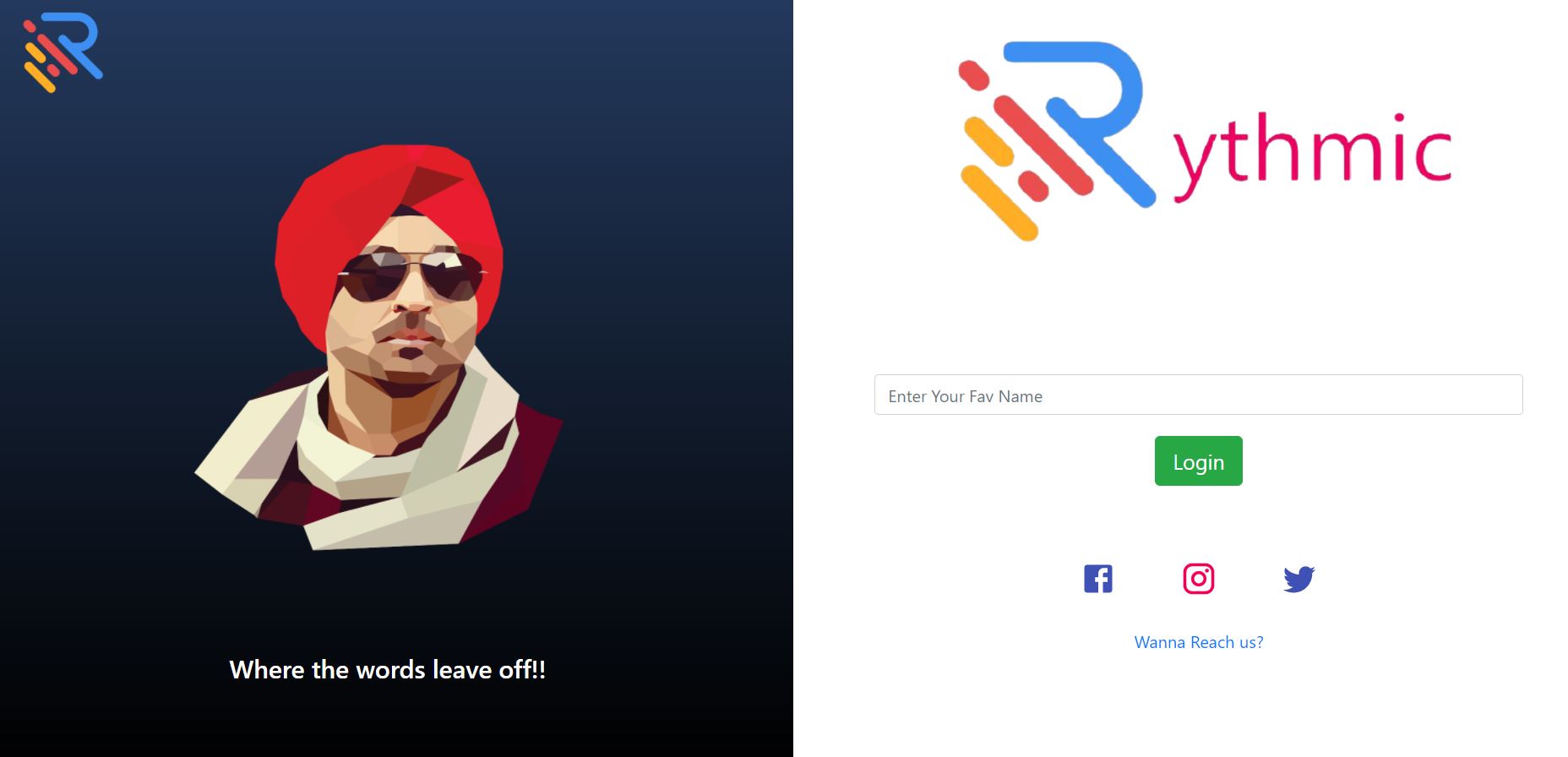
## INDIVIDUAL CONTRIBUTIONS:

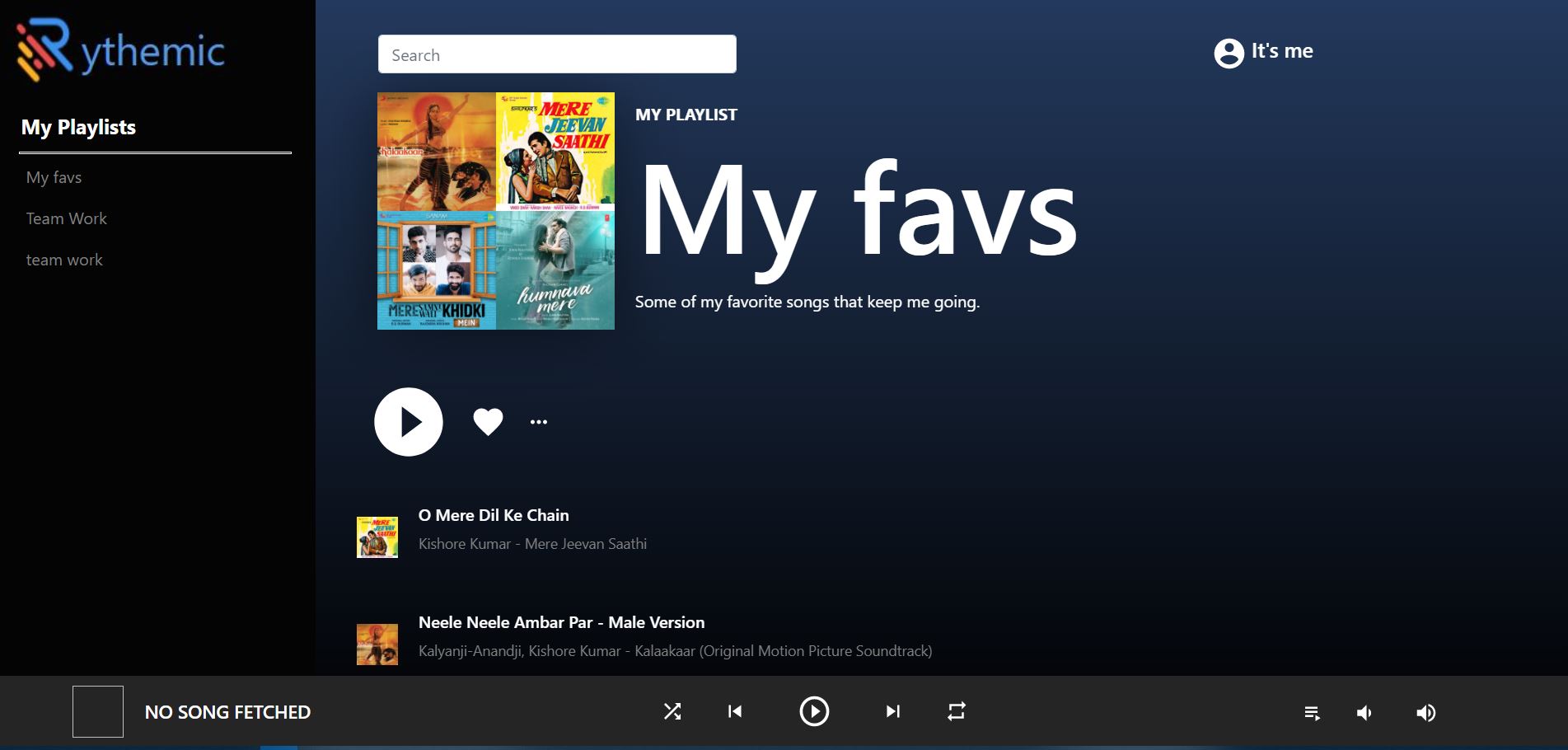
**Saumya Singh Bisht:** Designing and editing project logo and complete implementation of **LOGIN** page using Bootstrap 4.5.3 and React JS (**Components:** Login.js, Login.css).

**Anuttar Jain:** Designing and CSS implementation of MAIN page, handling of playlist Toggle button and playlist switching functionalities in the Sidebar of **MAIN** page along with its implementation (**Main** **Components:** App.js, Body.js, Sidebar.js, SidebarOption.js) using React JS and CSS.

**Yashbeer Singh:** Implementation and handling of searchbar, song toggling functionalities in the Footer of **MAIN** page (**Main Components:** App.js, Header.js, Footer.js, reducer.js) using React JS.

## FINAL RESULTS:



****

## SUMMARY:

This project helped us in learning new advanced topics of ReactJs, we improved our knowledge about fetching data from APIs and gained a lot of coding experience.