CS375 - Final Project Report

Introduction:

Team name: Song Masheen

Team members:

1. Olayinka Badmus
2. Cole Hoener
3. John Kenney

Code Repository: https://github.com/John-Kenney115/CS375.git

Project Description: For this project, our group designed a playlist-making website that incorporates the Spotify API. A user can search for tracks that are within Spotify's music library, and automatically create a 7 song playlist based on the song they pick. Playlists can then be saved to their local user or to a Spotify account.

Architecture:

Technologies Used:

1. Node JS, and the following modules:
    1.1. Express
    1.2. Axios
    1.3. CORS
    1.4. Pg, or Postgres
    1.5. Cookie-parser
    1.6. Bcrypt
    1.7. Spotify-web-api-node
2. Postgres as the local database system
3. Spotify Web API

New Technology:

1. React:
    1.1. The react component takes in 4 parameters and displays a static html component. These 4 parameters are the artist, song name, album name, and album cover. These parameters are then displayed according to the css style sheet
    1.2. Obstacles while working with React came in the form of having to embed React into our existing project rather than starting fresh with a React project. In the future, using the React project start up command would work better since it includes JSX.

Overview of Architecture:

1. Login/Create User:
    1.1. Upon navigating to the URL “localhost:3000”, a script at the top of the index.html file checks the stored cookies for one named “logged_in” and if it is not set to true it will redirect the client to login.html.
    1.2. The login.html page has inputs for username and password, as well as a link to the create.html page, where new users can set up an account. Login.html takes the input strings, checks to see if username exists in the database and then performs a bcrypt comparison on the stored hashed password and the input password. If these passwords match then two cookies are set: the username of the client that just logged in, and “logged_in” is set to true.
    1.3. The create.html page is essentially the same, although no comparison of passwords is made. Instead, the server performs a few checks on the string length of both the username and password, as well as to make sure that username does not already exist within the database.
2. Search songs and Create Playlist:
    2.1. To search for a song, the user clicks the search button which sends a fetch request to the server with the user’s input in the query. The server

returns with ten track objects provided from the Spotify API that have a similar artist name, album name or track name to the user's input.

2.2. When the user adds a song to the playlist, the server responds with six songs that are related and these are added to the playlist.

3. Save playlist locally:
   3.1. When a song is added to a playlist, its Spotify API response object's href and uri are both saved to a database table "Songs". This table has columns for username, playlist name, track href, and track uri.
   3.2. The server checks to make sure that identical hrefs/uris are not saved for that user and that specific playlist before adding. This all occurs in the function addSong() in local_save.js.
4. Retrieving stored information from DB:
   4.1. The functions getPlaylists() and getSongs() both access the database and respond with query results that match the "username" cookie.
   4.2. getPlaylists() will search the user_playlists table for anything that matches the "username" cookie, and will update the DOM with the results (specifically the dropdown that allows the user to select which playlist they are viewing/updating).
   4.3. getSongs() will search the songs table for anything that matches the "username" cookie, as well as the currently selected playlist in the dropdown menu, and will update the DOM with the results (specifically the right side, or playlist section, of index.html).

Reflection:

Planned Features:

- Search for a song
- Choose the correct song from returned choices
- Generate a 7 song playlist from chosen song
- Option to save to your Spotify library

Implemented Features:

- Search for a song
- Choose the correct song from returned choices
- Generate a 7 song playlist from chosen song
- Save playlist to your local account

Obstacles:

- Working with Promises
- Spotify API

Features that did not make it:

- Saving to Spotify - We were not able to implement this feature because it required several requests to the Spotify API, which was giving us strange errors that we could not decipher. Next time, we would have split up complex features instead of just one person working on it.