Project Title: Spotify Comment Hub

Team members: Sally Xue, Pradyumann Singhal, Lily Zhang, Joseph Schanne

Project Summary

(GOT TA APPROVAL FROM ANAY BHAKAT)

Our platform enhances the Spotify experience by allowing users to rate and comment on songs, a feature currently unavailable on Spotify. By integrating with Spotify's API, users can authenticate with their Spotify accounts and use their Spotify usernames on our site. They can comment and rate songs, and ratings are averaged and visualized for easy viewing.

To provide some initial content to the users, we plan to add songs/albums based on Spotify's popular playlists and retrieve editorial notes for albums from Apple Music to display them with the albums in our database.

Additionally, users can receive personalized playlist recommendations based on their song ratings, comments and listening history, and optional features include analyzing their musical tastes and building a community for users with similar preferences.

Project Description:

(This would be the MVP we are aiming for)

Problem we want to solve:

Currently, Spotify lacks a comment or rating functionality, preventing users from sharing their opinions on songs. This limits user interaction and engagement around the music they listen to. Our website aims to solve this problem by allowing users to rate and leave comments on songs, creating a space for discussion and feedback.
Based on Spotify API : https://developer.spotify.com/documentation/web-api

Detailed Design and Functionalities:

- **User Authentication and Navigation**

- ➢ **User Sign-In**:
  - Users authenticate through Spotify using SPotify credentials
  - Upon successful registration, users use their Spotify username as their username on our website.
- ➢ **Navigating the Start Page**:
  - After signing in, users are directed to the start page where they can search for songs or albums.

❖ ***Searching and Commenting on Songs***

- ➢ **Song and Album Search**:
  - Users can enter the name of a song or album on the start page.
  - The system uses Spotify's Search API to retrieve details such as Title, Artist, Album, Release Date, etc.
  - For albums, editorial notes from Apple Music's API (Get a Catalog Album: https://developer.apple.com/documentation/applemusicapi/get_a_catalog_album) are used to provide descriptions when albums are added to the database.
- ➢ **Adding Comments**:
  - If the song is found and not already in the database, it is added along with the user's comment.
  - Users can rate the song (1-10) and add an optional text comment.
  - The comment is then added to the song's list of comments.
- ➢ **Comment Submission Interface**:
  - Comments are submitted through a dedicated page or pop-up window where users can enter their ratings and text comments.

❖ ***Viewing and Managing Comments***

- ➢ **Song Review Page**:
  - Displays all comments left by users for the song.
  - Shows usernames, ratings, and includes links to the user's comment history page.
  - Comments are ranked based on weighted ratings. Higher weights are assigned to comments from users who have listened to the song frequently or have it among their top tracks. These weighted comments are prioritized and appear higher in the list. We will use Spotify's User - Get User's Top Items API.
- ➢ **User Comment History Page**:
  - Displays all comments left by a specific user.
  - Organized by song/album, showing each comment and linking to the relevant song or album.
  - Acts as a user profile page detailing the user's comment history.

❖ ***4. Initial Database Setup***

➢ **Database Initialization**:

- The website will start with an empty song database.
- The database will be populated using songs from charts like Billboard Hot 100 or playlists from Spotify's API, such as Today's Top Hits (Spotify Playlist API).

Technical Challenging Features

We will try to add these features however we might not be able to due to technical and time constraints.

- Analyzes a user's comments left on various songs across the platform and uses this data to infer their musical tastes and preferences.
  - Songs they left comments on: Spotify API would provide Energy, Tempo, Liveness, etc from Endpoint /audio-features
  - Comments: We could use NLP to extract useful insights from the words and phrases in user comments and then map those insights to standard Spotify song criteria, such as genre, tempo, energy, and other attributes
- Based on these insights, the function generates a personalized playlist recommendation for the user.
  - Spotify's Tracks - Get Recommendation API
  - For example, the Recommendation API needs seed tracks, seed artists, or seed genres. For seed genres, we can incorporate insights from the user's comments rather than relying solely on the songs' genres. This approach provides more comprehensive and personalized song recommendations, as it captures nuanced preferences that may not be reflected by the song genres alone.
- Build a social community for users with similar musical preferences, inferred from their comments and interactions with songs on the platform. It connects users who share similar tastes, allowing them to discover new music, engage in discussions, and participate in group activities like collaborative playlist creation or song recommendations.

Usefulness

1. **Overview:** Spotify has 626 million monthly active users but there is no place for users to put comments or rate songs. Our platform enhances the Spotify

experience and anyone who listens to music by allowing users to rate and comment on songs, a feature currently unavailable on Spotify.
    - Sharing comments is useful because it allows people to express their thoughts, engage in discussions, and connect with others who share similar music tastes, fostering a more interactive and community-driven listening experience.
2. **Additional Features:** Additional features would include giving song recommendations based on user history and song ratings.
3. **Similar websites/applications and how we are different:** In comparing our project to platforms like Genius, we aim to create a simpler and cleaner user interface. While Genius provides detailed song lyrics and annotations, our focus is on making it easy for users to comment and share their thoughts about songs. Our platform will allow users to rate songs, which helps create a sense of community and gives feedback to artists. Additionally, we will include a feature for nested comments, so users can reply directly to each other's comments. This way, users can have meaningful discussions about music without any distractions.

Realness

The data for this project is sourced from two primary platforms:

1. **Spotify**: We will leverage the Spotify API to extract various datasets for our platform, focusing on key features that enhance user interaction and content display. The data would be in JSON format. Key datasets we may use include (subject to adjustment during implementation):
    - **Song Information**: Title, Artist, Album, Release Date (from Spotify's Search API), users' comment, etc.
    - **Comment Table**: comment_id, song_id (connect with song information table), user_id (connect with User Data table), Rating (1-10), weighted ratings, etc.
    - **User Data**: Spotify username, comment history (connect with the comment table), listening history, etc.
2. **Apple Music**: Descriptions and additional contextual information about various music tracks are obtained from Apple Music API (https://developer.apple.com/documentation/applemusicapi/get_a_catalog_album). This data helps enrich the comments and recommendations with detailed descriptions and context, which are also provided in JSON format.

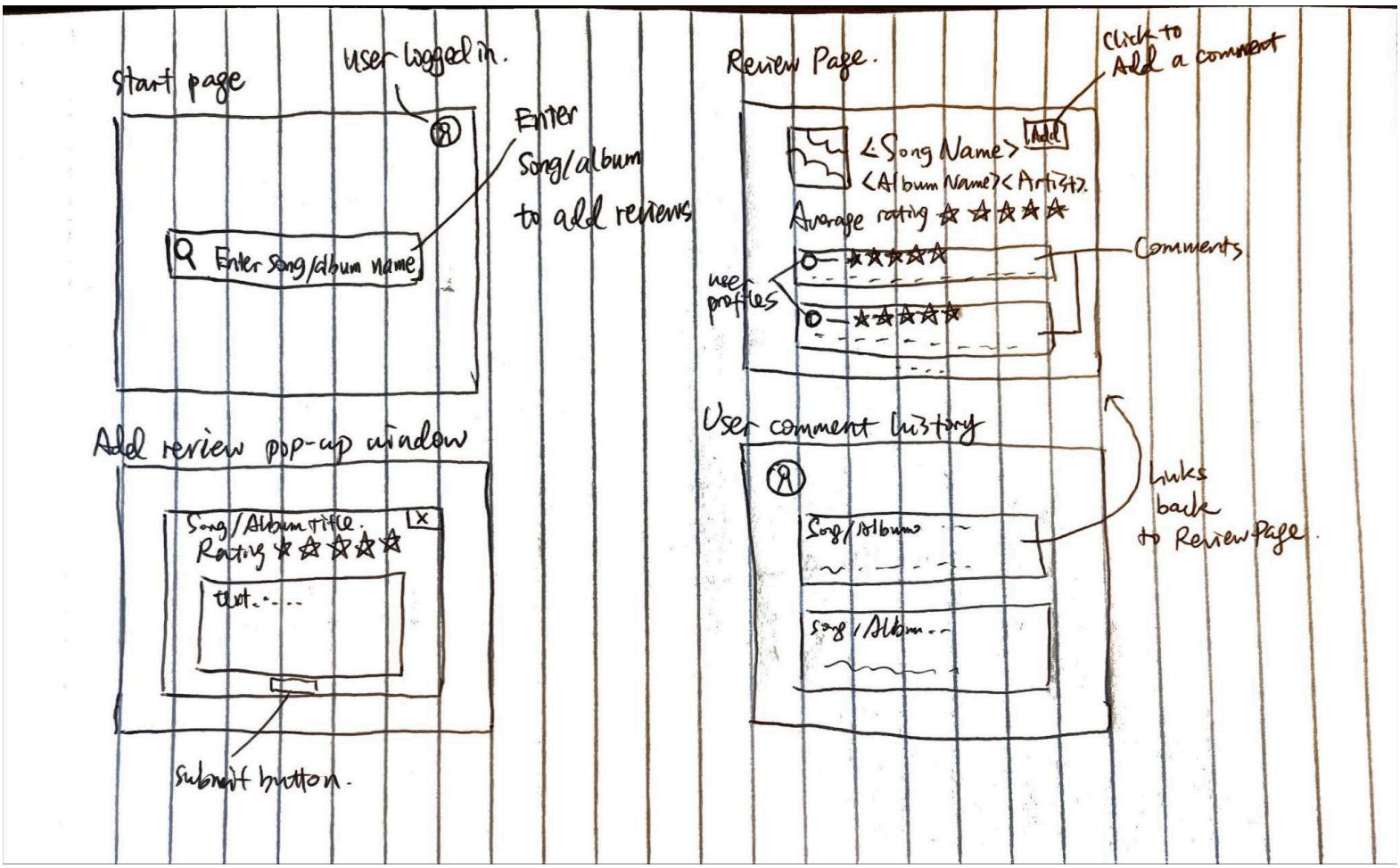
Sample schema would be like:

- id: The unique identifier for the album in Apple Music's catalog.
- name: The title of the album.
- artistName: The name of the artist(s) who released the album.
- releaseDate: The date the album was released.
- genreNames: A list of genres associated with the album.
- trackCount: The number of tracks on the album.
- url: A URL to the album on Apple Music.
- **Editorial Notes:**
    - short: A brief description or promotional text about the album (optional).
    - standard: A more detailed description or editorial note about the album.

The dataset's size will vary based on the number of songs and playlists queried but will generally encompass a high cardinality with a large number of unique songs and attributes. The degree of the data includes song titles, artists, albums, release dates, and detailed descriptions from Apple Music.

Description of functionalities

a. UI mockup



b. Project work distribution

- **Frontend** - Pradyumann Singhal , Sally Xue , Lily Zhang
- **Backend**
  - CRUD Operations - Sally Xue , Pradyumann Singhal
  - Apple API - Sally Xue, Pradyumann Singhal
  - Spotify API - Lily Zhang , Joseph Schanne
- **Database** - Joseph Schanne, Lily Zhang