550 Project Outline - Spotify Songs

1. Motivation for the idea

Our website's goal is to provide a genuinely distinctive and customized music discovery experience while overcoming the drawbacks of the widely used music apps available today. Our website enables users to search for music according to their own needs and create their own private playlists customized to their own interests, unlike many other platforms where users are limited to listening to playlists made by others. We understand that everyone has distinct tastes and that a one-size-fits-all playlist strategy frequently leaves consumers unsatisfied. We provide customers the power to create a customized library that precisely suits their tastes and moods by letting them browse music using various features and hear song samples before adding them. Our website aims to give users full control over their music journey, enhancing their discovery process in a way that feels entirely their own.

2. List of Features

1. Homepage Suggestion: where show top popular songs/albums, and a daily song advice
2. Search bar: where the user can search the song's name/album name/artist name
3. dropdown menu: where the user can find songs based on their genre
4. Another drop-down menu: where users can find songs based on their functions(e.g.dance, etc)
5. A filter: where users can choose how many results they want to see on one page

3. List of Features (Optional)

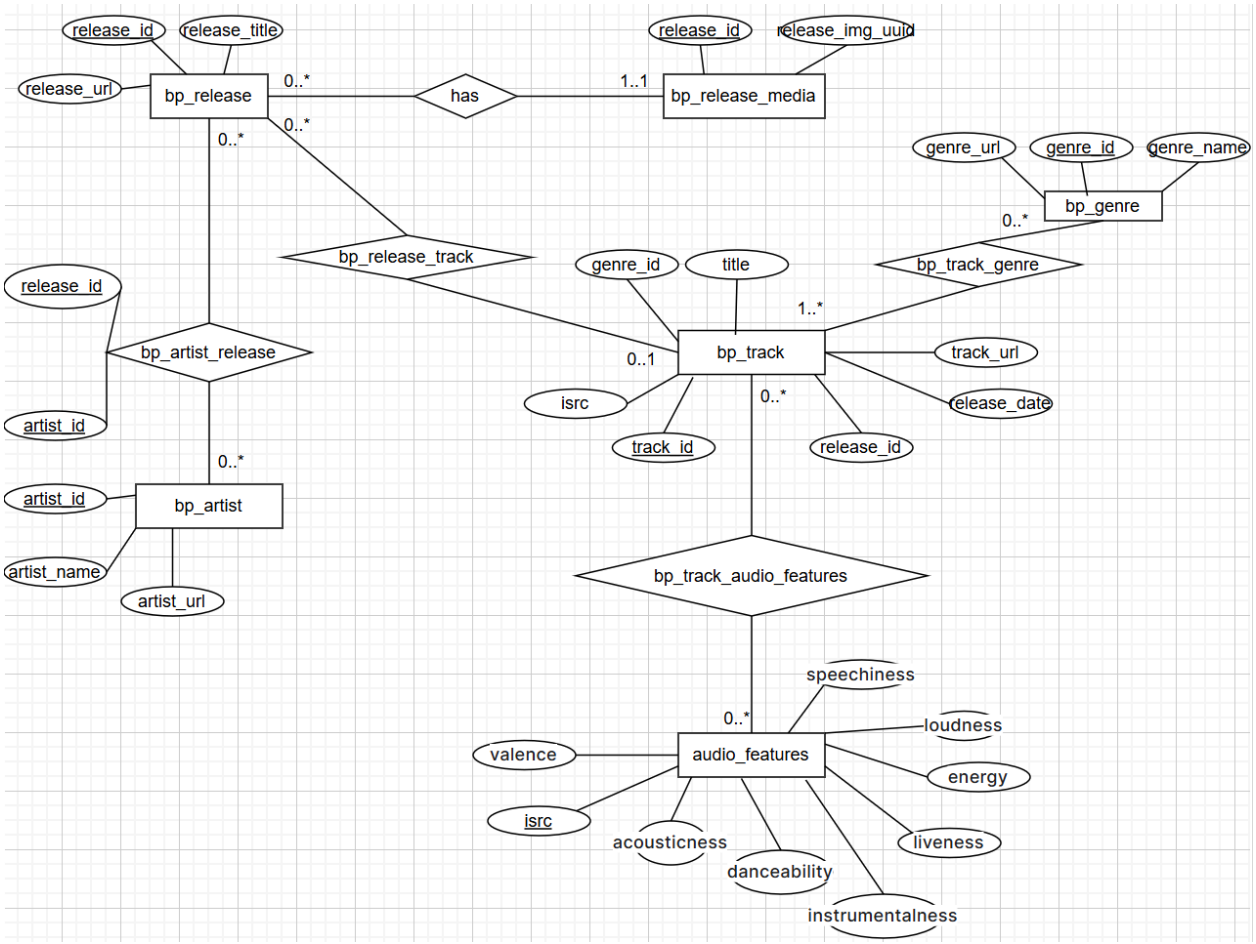
1. A login page with a Google/Facebook account
2. An add button allows the user to add songs to their own playlists.

4. List of pages the application

1. Home Page: user can choose what they want to explore on our website
2. Search Page: user can search for a song based on the album name, track name, release year, artist name, etc., as well as choose how many songs they want to show on a single page
3. Functionality filter page: users choose functionalities of songs they want and filter them.
4. Create my own List page: The user can add their favorite songs to their private lists, as well as rate the song, categorize the song, and add some notes to the song. (Optional)
5. Login Page: users can create accounts and access their own databases, or as guests. (Optional)

5. Relational schema

https://www.kaggle.com/datasets/mcfurland/10-m-beatport-tracks-spotify-audio-features?resource=download&select=bp_genre.csv



6. SQL DDL_Ver 1

CREATE TABLE bp_release (
release_id INT PRIMARY KEY,
release_title VARCHAR(255) NOT NULL,
release_url VARCHAR(255));

CREATE TABLE bp_release_media (
release_id INT PRIMARY KEY,
release_img_uuid INT NOT NULL,
FOREIGN KEY (release_id) REFERENCES bp_release(release_id) ON DELETE CASCADE);

CREATE TABLE bp_artist (
artist_id INT PRIMARY KEY,
artist_name VARCHAR(255) NOT NULL,
artist_url VARCHAR(255));

CREATE TABLE bp_artist_release (
artist_id INT,
release_id INT,
PRIMARY KEY (artist_id, release_id),

FOREIGN KEY (release_id) REFERENCES bp_release(release_id) ON DELETE CASCADE,
FOREIGN KEY (artist_id) REFERENCES bp_artist(artist_id) ON DELETE CASCADE);

CREATE TABLE bp_genre (
genre_id INT PRIMARY KEY,
genre_name VARCHAR(255) NOT NULL,
genre_url VARCHAR(255));

```
CREATE TABLE audio_features (
isrc CHAR(12) PRIMARY KEY,
valence FLOAT,
acousticness FLOAT,
danceability FLOAT,
instrumentalness FLOAT,
liveness FLOAT,
energy FLOAT,
loudness FLOAT,
speechiness FLOAT
);
```

```
CREATE TABLE bp_track (
track_id INT PRIMARY KEY,
genre_id INT,
title VARCHAR(255) NOT NULL,
isrc CHAR(12) UNIQUE,
track_url VARCHAR(255),
release_date DATE,
release_id INT,
FOREIGN KEY (genre_id) REFERENCES bp_genre(genre_id) ON DELETE SET NULL,
FOREIGN KEY (isrc) REFERENCES audio_features(isrc) ON DELETE SET NULL,
FOREIGN KEY (release_id) REFERENCES bp_release(release_id) ON DELETE CASCADE
);
```

CREATE TABLE bp_track_genre (
(track_id, genre_id) PRIMARY KEY,
track_id INT,
genre_id INT,
FOREIGN KEY (track_id) REFERENCES bp_track(track_id) ON DELETE CASCADE,
FOREIGN KEY (genre_id) REFERENCES bp_genre(genre_id) ON DELETE CASCADE);

CREATE TABLE bp_track_audio_features (
(track_id, isrc) PRIMARY KEY,
track_id INT,
isrc CHAR(12),
FOREIGN KEY (track_id) REFERENCES bp_track(track_id) ON DELETE CASCADE,
FOREIGN KEY (isrc) REFERENCES audio_features(isrc) ON DELETE CASCADE);

CREATE TABLE bp_release_track (
(release_id, track_id) PRIMARY KEY,

release_id INT,
track_id INT,
FOREIGN KEY (release_id) REFERENCES bp_release(release_id) ON DELETE CASCADE,
FOREIGN KEY (track_id) REFERENCES bp_track(track_id) ON DELETE CASCADE);

7. SQL DDL_Ver 2 (with correct order of tables):

```
CREATE TABLE bp_release (
release_id INT PRIMARY KEY,
release_title VARCHAR(255) NOT NULL,
release_url VARCHAR(255)
);

CREATE TABLE bp_artist (
artist_id INT PRIMARY KEY,
artist_name VARCHAR(255) NOT NULL,
artist_url VARCHAR(255)
);

CREATE TABLE bp_genre (
genre_id INT PRIMARY KEY,
genre_name VARCHAR(255) NOT NULL,
genre_url VARCHAR(255)
);

CREATE TABLE audio_features (
isrc CHAR(12) PRIMARY KEY,
valence FLOAT,
acousticness FLOAT,
danceability FLOAT,
instrumentalness FLOAT,
liveness FLOAT,
energy FLOAT,
loudness FLOAT,
speechiness FLOAT
);

CREATE TABLE bp_track (
track_id INT PRIMARY KEY,
genre_id INT,
title VARCHAR(255) NOT NULL,
isrc CHAR(12) UNIQUE,
track_url VARCHAR(255),
release_date DATE,
release_id INT,
FOREIGN KEY (genre_id) REFERENCES bp_genre(genre_id) ON DELETE SET NULL,
FOREIGN KEY (isrc) REFERENCES audio_features(isrc) ON DELETE SET NULL,
FOREIGN KEY (release_id) REFERENCES bp_release(release_id) ON DELETE CASCADE
);
```

```
CREATE TABLE bp_release_media (
release_id INT PRIMARY KEY,
release_img_uuid INT NOT NULL,
FOREIGN KEY (release_id) REFERENCES bp_release(release_id) ON DELETE CASCADE
);

CREATE TABLE bp_artist_release (
artist_id INT,
release_id INT,
PRIMARY KEY (artist_id, release_id),
FOREIGN KEY (release_id) REFERENCES bp_release(release_id) ON DELETE CASCADE,
FOREIGN KEY (artist_id) REFERENCES bp_artist(artist_id) ON DELETE CASCADE
);

CREATE TABLE bp_track_genre (
   track_id INT,
   genre_id INT,
   PRIMARY KEY (track_id, genre_id),
   FOREIGN KEY (track_id) REFERENCES bp_track(track_id) ON DELETE CASCADE,
   FOREIGN KEY (genre_id) REFERENCES bp_genre(genre_id) ON DELETE CASCADE
);

CREATE TABLE bp_track_audio_features (
   track_id INT,
   isrc CHAR(12),
   PRIMARY KEY (track_id, isrc),
   FOREIGN KEY (track_id) REFERENCES bp_track(track_id) ON DELETE CASCADE,
   FOREIGN KEY (isrc) REFERENCES audio_features(isrc) ON DELETE CASCADE
);

CREATE TABLE bp_release_track (
   release_id INT,
   track_id INT,
   PRIMARY KEY (release_id, track_id),
   FOREIGN KEY (release_id) REFERENCES bp_release(release_id) ON DELETE
CASCADE,
   FOREIGN KEY (track_id) REFERENCES bp_track(track_id) ON DELETE CASCADE
);
```

8. Clean and Pre-process

Delete the rows with null values; Drop columns we do not need; Entity Resolution (change column names); Replace Categorical Variables with Indicators (genre_id for genres); Standardize feature audio_feature(loudness) to [0,1] (from mostly negative to small positive decimal right now)

9. List of Technologies

- Front-end: HTML; CSS; React.js
- Back-end: Node.js; PostgreSQL Database; SQL in PostgreSQL for queries; AWS RDS; Javascript; Github; (> 80% test for backends planned)
- Others for EDA and Integration (Planned):
  Colab and Bing search; Integrated with APIs to fetch streaming data

10. Each Member's Role

Yiting: Front-end development of the website.
Haorui: Back-end like DB and/or .js files
Jason Pan: Front-end development of the website
Kris Zhang: Back-end like DB and/or .js files
(And we think we will support each other when we need help/discussions)

Extra Parts - Not Submit, as the 4th Page

3. Extra Credits

Here are some of the aspects that may be awarded with extra credit. However, individual TAs may grant additional extra credit under their discretion. 2 points per EC feature for a maximum of 4 points may be awarded, and the final project score cannot exceed 100%.

- Used NoSQL in addition to SQL
- Integrated with apis to fetch streaming data
- Code coverage (unit testing >80% for backend and/or >80% frontend)
- Application Security (implemented password hashing, privacy modes, SSL certificates etc) (hashing alone will not count)
- Integration with other applications like colab, bing search etc.
- User login experience (Integrating with at least 2 of the following: Google, Facebook, Twitter, etc. Sign In and in addition to having standard sign in)
- Awards (rate 1-5 on technical complexity, best looking, and all-around best)
- Anything cool we might've missed (Subjective and depends on complexity)
- Deployment (need to have the website deployed so TAs can access during presentation)

Bp_release (release_id, release_title, release_url)
Release_id: primary key

Bp_release_media (release_id, release_img_uuid)
Release_id: primary key
Release_id: foreign key reference Bp_release(release_id)

Bp_artist_release (artist_id, release_id)

Primary key: (artist_id, release_id)
Release_id: foreign key reference Bp_release(release_id)
Artist_id: foreign key references Bp_artist(artist_id)

Bp_artist (artist_id, artist_name, artist_url)
Artist_id: primary key

Bp_track (track_id, genre_id, title, isrc, track_url, release_date, release_id)
Track_id: primary key
Genre_id: foreign key reference Bp_genre(genre_id)
Isrc: foreign key references audio_features(isrc)
Release_id: foreign key references Bp_release(release_id)

Bp_genre (genre_id, genre_name, genre_url)
Genre_id: primary key

Audio_features (isrc, valence, acousticness, danceability, instrumentalness, liveness, energy, loudness, speechiness)
isrc: primary key