Student Details:

Name: Hamees Ul Hasan Sayed
Roll Number: 23F1001168
Email: 23f1001168@ds.study.iitm.ac.in

Project Description:

The aim of this project is to provide users with a free music streaming app where the user can surf the recent songs and albums by their favourite creator. A user is also able to register itself as a creator to upload songs and albums or update and delete them. A user also has the ability to create playlists and add songs to it.

Technologies Used:

HTML, CSS, Javascript, Jinja, Bootstrap, Flask, VueJS, Flask-SQLAlchemy, Flask-Bcrypt, Mutagen, JWT, Redis and Celery.

- Python is the core programming language used to write the app.
- Flask is the core framework to build the Backend.
- JWT is used for managing user login and their session.
- Flask-SQLAlchemy is an ORM tool used to interact with the database.
- VueJS is used to build a seamless user interface along with Bootstrap.
- Flask-Caching is used for caching while celery is used for async jobs.

Database Schema:

The ER Diagram for the Database Model can be seen here:
https://dbdiagram.io/d/Fourier-65378240ffbf5169f05399fe

API Design:

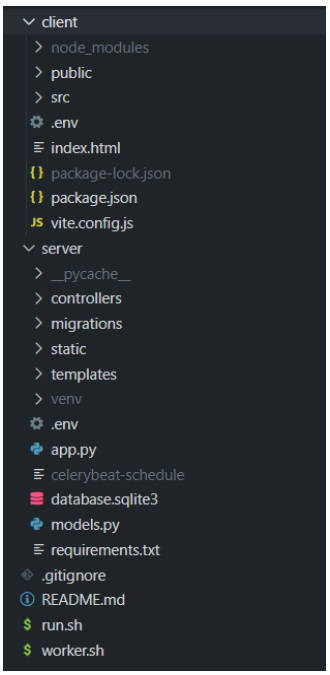
I have implemented GET, PUT, POST and DELETE for User, Creator, Album, Playlist and Song Table in the Database Model.

https://app.swaggerhub.com/apis/hamees-sayed/fourier-music_streaming_app_open_api_3_0/1.0.0

Here you can have a look at all the available endpoints and their respective use cases.

App Architecture:

This Flask application follows a modular structure with controllers handling specific features. Database models are defined in `models.py`, and Database migrations are managed using Alembic. The static directory includes images, styling and serves as a filesystem to store media (audio songs in this case), while client organize vuejs components for rendering frontend. The `app.py` file serves as the entry point to run the project on the backend and main.js file serves as the entry point to run the project on the frontend. The bash scripts are used to start the backend, frontend and celery workers with a single command.



Features:

1. Authentication and Authorization:
    - Users can register and login with a unique username and password. Passwords are securely hashed using Bcrypt.
    - Users can also logout of the current session.
2. Admin Management:
    - Admins can be created from the command line and have a dedicated login page.
    - Admins can view and supervise entire App statistics through the Admin Dashboard. The Dashboard visualizes the highest performing songs and user activity through graphs.
3. Creator:
    - Users can register as creators and creators can manage songs and albums and also view their statistics through graphs on how their songs are performing.
4. Search Functionality:
    - Users can search for songs and albums based on title, lyrics, creator and rating
5. Home Page:
    - Displays a list of all the songs with genre, average rating, creator name and lyrics. Users can also play these songs.
    - Also provides a link to albums for further exploration, clicking on an album displays a list of songs in that album.
6. Users:
    - Users have the ability to rate songs and also add songs to the playlist and manage the playlist.

Presentation Video:

Link to my Presentation Video