Author

Siddharth Pandey

22f2001147

22f2001147@ds.study.iitm.ac.in

I am a full time student, currently pursuing two degrees, an on-campus degree in Bachelor of Arts (Mathematics), and the other is an online BS degree by IIT Madras. My interests particularly include computer science, mathematics and statistics.

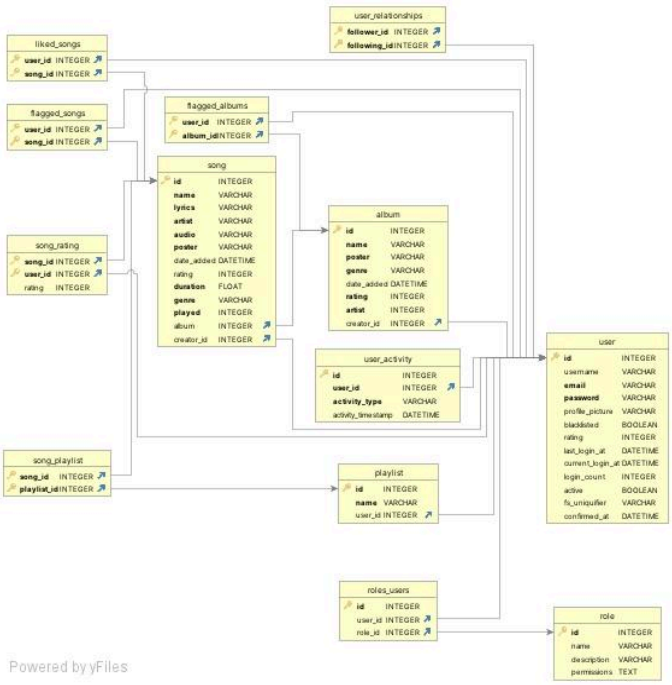
Description

The project (Music Streaming App) is assigned to me as a practical course on MAD-2 theory. This is a multi-user application, where the users have the functionality of listening to music, reading lyrics, etc. it also provides creators with the functionality to upload songs and manage them.

Technologies used

- Flask
- SQLite
- SQLAlchemy
- DB Browser
- Flask-Restful
- Flask-Security
- Bootstrap
- Vue.js (Vue CLI)
- Bootstrap-vue
- Celery
- Redis
- Flask-Caching

DB Schema Design



The above image describes the structure of the database schema. The primary tables include; Song, Album, User, Role and Playlist. There are a number of relationships described to make the application work.There are one-to-many relationships described between song and album, song and user, album and user and playlist and user. Secondly, the relationship tables include liked_songs, song_rating, song_playlist, user_relationship, flagged_songs and flagged_albums establishing a many-to-many relationship between user and songs, song and user and song and playlist, user and user, user and songs and user and albums respectively. The below design helped to make the application work the way it was supposed to.

API Design

API is designed using flask-restful. It has been designed for CRUD operation on the tables; Song, Album and Playlist.

Architecture and Features

My project has the root folder named 'Project', which consists of two subfolders, backend and frontend. The backend contains all the server side code, it has another subfolder named application which contains files like models.py, views.py, resources.py, etc which contains all the code required for the application to run, other than that there are other files like main.py and initial_data.py which is also required for the application. The frontend has all the client side code, it contains subfolders like public and src, and other configuration files. The folder src contains the necessary frontend code including the codes for vue-router and .vue files.

The application has all the core features of the user, creator and the administrator. An user can play songs, read lyrics, create/update playlists, rate songs, search for songs/artists/genre/album ,etc. A creator has all functionalities of the user and also the ability to upload/update songs, create/update albums, etc. On the other hand the administrator has the ability to monitor the application with features like deleting a song/album, flagging creators, viewing various statistics and graphs related to the application. The app uses redis for caching and celery for jobs like sending monthly reports to creators and also sending daily reminders to visit the application. The application uses flask-security to facilitate token based authentication and role based access control.It also has recommended features like API, login system using flask-login. The application is styled using bootstrap, bootstrap-vue and some custom css. Other than this, the extra features implemented are to be able to like a song, follow creators and be able to update their profiles.

Video

DemoVideo.mp4