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

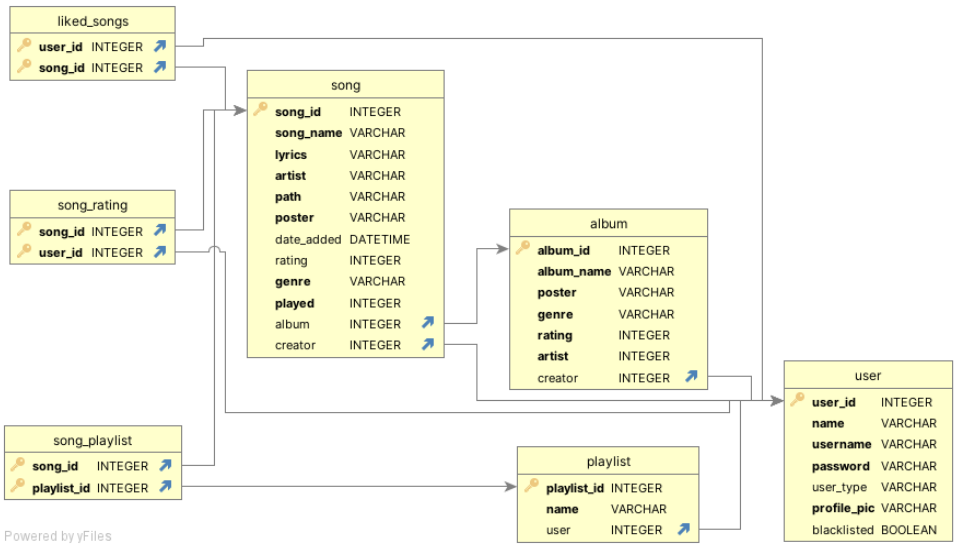
The project (Music Streaming App) is assigned to me as a practical course on MAD-1 theory. This is a multi-user application, where the users have the functionality of listening to music, reading lyrics, etc. it also provides creators with the functionality to upload songs and manage them.

Technologies used

- Flask
- Jinja2
- SQLite
- SQLAlchemy
- DB Browse
- Flask-Restful
- Flask-login
- Bootstrap

DB Schema Design

The below image describes the structure of the database schema. The primary tables include; Song, Album, user 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 and song_playlist establishing a many-to-many relationship between user and songs, song and user and song and playlist 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 and also for Read operation on the User table.

Architecture and Features

My project has the root folder named as ‘PROJECT (MAD-1)’, which consists of all the templates, located in the templates folder, the style.css file in static folder along with other folder such as audios, posters, album_posters, graphs, profile_pictures, etc. which consists of the images required for the application (the subfolders in static are not submitted at the time of submission). The root folder consists of the .py files, namely three of them models.py containing the models of the application, api.py containing the designed api and app.py which consists of all the controllers of the application. It also has a README file which contains the basic instructions on how to run the application.

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. It also has recommended features like API, login system using flask-login. The application is styled using bootstrap and some custom css. Other than this, an extra feature implemented is to be able to like a song,

Video

Demo video