Music Streaming App

Project Report

By: Bhavya Jain

Roll No: 21f1003868

About the project:

This is a project to create a multi-user music streaming app with the help frameworks like Flask and SQLAlchemy. We had to an sqlite database to store the data. The app would have the following functionalities-

- User can register as creator.
- User can create playlist and view them in their profile.
- Creator can upload albums and songs. He/she also has the option to update or delete the songs/albums he/she has created and can view them in their profile.
- Creator also has the option of creating his/her own playlist.
- Admin has the ability to view all the songs, albums created. Admin can also view all the creators in the app.
- Admin has the power of deleting songs and albums as he/she deems fit.
- Admin also has power to blacklist any creator as he/she deems fit. Admin can also whitelist the blacklisted creators.

Models in the app:

There were 7 Models used by me while making this project:

1. **User**- This model is used to store user data. The user is asked about his/her first name, last name, email ID, preferred username and password on registering as a user on the app. This data is stored in the User model. Each user has his/her own unique user id that s also stored in the User model.
2. **Creator**- As mentioned in the functionalities, a user can register as a creator. This model captures that data. The user is asked about his/her username, his/her preferred artist name while registering as a creator. This data is stored in the Creator model along with a unique creator id.
3. **Album**- This model is used storing the data about the albums creators create. A new album is required to have an album name and artist name

when it is been created. This data is stored in the Album model along with a unique album id for each album. The Album and Creator models have many-to-one relationship.

4. **Songs**- This model is used to store data about the songs that are being created. A new song is required to have a title, lyrics, genre, album name and artist name. Each song is also assigned with a unique song id. The Songs and Album models have many-to-one relationship.
5. **Playlist**- This model captures the playlist been creator by user/creator. A new playlist is required to have a playlist name, description. Each playlist is assigned with a unique playlist ID. This model captures the many-to-many relationship between Songs and User.
6. **SongRatings**- This model is used to store data about the ratings of a song. It stores the number of likes and dislikes a song has.
7. **Admin**- This model is used to store data about the administrator. Each administrator has a unique admin ID assigned to himself/herself. The model also stores administrator's first name, last name, preferred admin username.