# Project Report

Project Name: Music Streaming Application

Video Link: Click here to view a short demo clip of the app.

Created by: Akash Sharma

Roll no.: 22F2000700

Student Email Id: 22f2000700@ds.study.iitm.ac.in

Short Description: It is a simple app for streaming songs, where a user can listen to music, read its lyrics, create their own playlist, rate songs, search through the vast range of songs available to them and browse through all the available albums & genres. One of the key features that the app offers is that it allows a user to register themselves as creator, giving them the power to upload songs and create new albums out of them. While there's also a special type of user called "the admin". A admin can browse through all the songs uploaded by the creators, view the app statistics and also mark a creator as blacklisted/whitelisted depending on the app policies.

Schema:

- Users (id, username, email, password, creator, blacklisted)
- Songs (id, title, singer, release date, lyrics, genre, filename, user_id)
- Playlists (id, name, user_id)
- Songs_in_Playlist (id, playlist_id, song_id)
- Albums (id, name, user_id)
- Songs_in_Album (id, album_id, song_id)
- Ratings (id, rating, song_id, user_id)

Journey through the Project: Upon receiving the question statement, I thoroughly reviewed it and attended a few live sessions conducted by the course team. Once I understood the requirements and the core functionalities of the app, I began building my very first project. Initially, I consulted the Flask documentation to set up the app, then used flask-login to build a proper login system. I utilized sqlite3 and flask-SQLAlchemy for the database schema and finally, matplotlib for creating graphs.

The first pages I created were the index, registration, and login pages for both the admin and user. Subsequently, I developed the user dashboard and registration pages for the creator, along with its dashboard. After successfully creating pages for

song upload and edit, I added functionality for song upload, edit, and delete by the creator, and designed a page for viewing song details.

Next, I introduced a feature for album creation from songs and included an option on the user dashboard for viewing albums, available genres, and playlist creation. I also added a feature on the song details page to rate songs. Searching for a song and playing it proved to be the most challenging part for me.

Finally, I moved on to the admin side, where I encountered additional challenges in creating graphs and implementing a functionality for blacklisting/whitelisting creators based on app policies.

API: I have used Flask-RESTful for making CRUD operations on songs.

- GET : /api/user/song/<int:song_id>
- DELETE, PUT : /api/user/creator/<int:uploader>/song/<int:song_id>
- POST : api/user/creator/song