Project Documentation

Author
Name : Kartikey Bhandari
Roll Number : 21f3002484
Email : 21f3002484@ds.study.iitm.ac.in

About :

"I am a 21-year-old passionate sports enthusiast and a dedicated business-startup growth enthusiast. I firmly believe that the fusion of technology and hands-on learning serves as the cornerstone for skill development, providing the most effective pathway to implement these skills in real-life scenarios. My commitment is to seamlessly integrate my acquired knowledge and philosophy into practical, real-world situations. As a storyteller and business developer, I consider myself fortunate to have these roles and am relentlessly committed to enhancing and refining my work."

Description :

This music streaming platform, akin to Spotify, features essential components such as User, Admin, and Creator, providing a secure and user-friendly environment. Admins manage tasks, ensuring platform integrity. The integrated module enriches user experience by calculating ratings, likes, and providing predefined genres, fostering interaction and discovery.

The platform features a comprehensive database of artists and songs. Admins oversee platform management, enabling tasks such as adding new songs and lyrics, monitoring user likes, viewing song and album statistics, and managing creator blacklisting/whitelisting and checks to ensure user data confidentiality. Flask-Login is employed for session management and access control.

In essence, this project offers a secure and user-friendly platform, allowing music enthusiasts to easily discover and enjoy songs, explore artists, and access lyrics. Admin has the ability to remove the artist, user and song they want to.

It also displays the likes, ratings and genre which has been listened to the most in the admins dashboard.

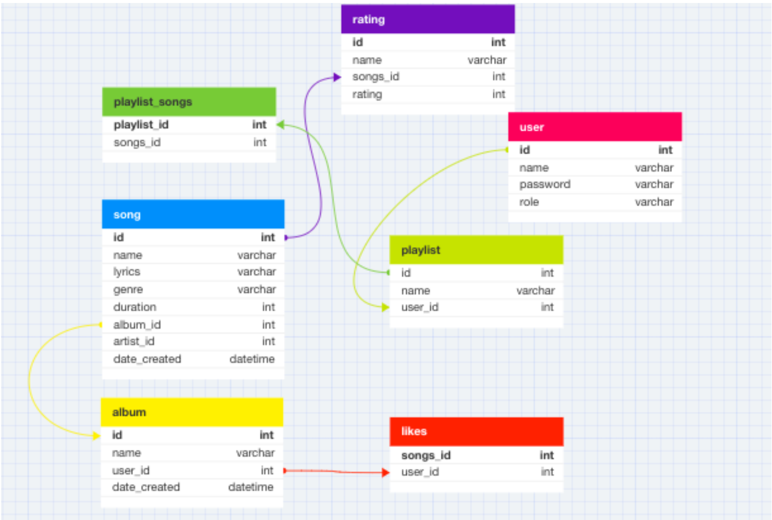
Technologies Used:

1. Flask: A micro web framework utilized for building the web application.
2. SQLAlchemy : An Object-Relational Mapping (ORM) library facilitating interaction with the database.
5. Login: A Flask extension managing user sessions and providing authentication features.
6. Flask-SQLAlchemy :A Flask extension simplifying the integration of SQLAlchemy with Flask.
7. Jinja2 : A template engine used to render dynamic HTML templates in Flask.
8. HTML, CSS, Bootstrap :Web development technologies utilized for crafting the user interface of the web application.
9. SQLite : A lightweight relational database management system employed for storing data in the application.
10.JavaScript : A language used in the development of search functionality.

DB Schema Design

The database design employs SQLAlchemy in a Flask application. It includes tables for user likes ('likes') and playlist-songs associations ('playlist_songs'). The core models consist of 'Rating,' 'Playlist,' 'Album,' 'User,' and 'Song.' Relationships between tables facilitate complex queries and data retrieval. Users can be artists, rate songs, create playlists, and upload albums, providing a robust structure for the music streaming platform. The 'Song' model captures essential details like name, lyrics, duration, genre, and artist, enhancing the platform's functionality and user experience.

The database includes tables such as 'likes' for user-song relationships, 'playlist_songs' for playlist creation, and 'Rating' for user ratings. It also features models like 'Album' for organized storage. This schema supports various user interactions and content organization in the music streaming platform.



rating	
id	int
name	varchar
songs_id	int
rating	int

playlist_songs	
playlist_id	int
songs_id	int

user	
id	int
name	varchar
password	varchar
role	varchar

song	
id	int
name	varchar
lyrics	varchar
genre	varchar
duration	int
album_id	int
artist_id	int
date_created	datetime

playlist	
id	int
name	varchar
user_id	int

album	
id	int
name	varchar
user_id	int
date_created	datetime

likes	
songs_id	int
user_id	int

API Design:

- CRUD functionality implemented for Songs,Album,Playlist.
- CRUD functionality implemented for User,Creator.
- Delete option incorporated for Songs,Album,User and Content creator profile.
- Edit Option for the creator to edit the song , album .

Architecture and Features:

The app.py file, the instance folder containing the SQLite database file, and the ticket folder containing the HTML+CSS templates and other significant.py files are all located in the root folder.
The ticket folder's primary Python files are as follows:
1. __init__.py: This code sets up a Flask web application, configures it using settings from a 'config' module, initializes a SQLAlchemy database, and imports various modules (models, routes, middlewares, api, utils) for building and extending the functionality of the Flask application.
2. routes.py: This file contains all of the routes to various pages and serves as the application's controller.
3. models.py: This file includes Python code for creating database models for a platform that plays music using SQLAlchemy .
4.config.py : This code generates a random secret key and sets up a SQLAlchemy database URI by loading environment variables from a `.env` file.
5.populate.py: Defines functions to create dummy instances of database models, and populates the database with dummy data, including users, songs, ratings, playlists, and albums, with relationships between them.
6.graphs.py : Functions to generate base64-encoded PNG charts using Matplotlib for displaying insights on popular genres, top-rated songs, and top-rated artists in a single line.
7.creator_utils.py : This module contains functions for calculating average ratings for an artist, total likes for a song, and retrieving a list of music genres.

8.middlewares.py : These Flask middleware functions enforce user login (`login_check`) and admin role (`admin_check`) requirements for accessing specific routes.
9.api.py : This code defines a set of Flask RESTful resources for handling user authentication, song management, liking, rating, and playlist/album creation in a music-related web application.

The music app features a robust registration system with thorough front-end and back-end validation. Separate login forms for Users, Creator and Admins ensure secure access to their respective portals. The intelligent Search bar allows users to explore songs and albums seamlessly. The app supports CRUD operations for playlist and songs , allowing users to create, edit, and delete music . Additionally, admins can manage their profiles, including viewing ratings, top genre and likes for a comprehensive and user-friendly experience.

Acknowledgement

"I extend my sincere gratitude to the Indian Institute of Technology, Madras' Team BS Degree Programme for the invaluable opportunity to collaborate on this exceptional project. I appreciate the guidance and expertise shared by the professionals and course instructors, which greatly contributed to the success of the project. Finally, I express my heartfelt thanks to my friends and family for their unwavering support throughout this endeavor."

Presentation Video Link : Here