MODERN APPLICATION DEVELOPMENT-1 PROJECT

MUSIC STREAMING APPLICATION

SEPTEMBER TERM-2023

NAME: G HAMSINI

ROLL NUMBER:22f3000767

EMAIL: 22f3000767@ds.study.iitm.ac.in

Greetings, everyone. I'm G Hamsini from Chennai, India and I'm in the diploma level of pursuing my Bachelors of Science degree in Data Science and Applications from IIT, Madras. I'm also in the second year of pursuing my Bachelors of Technology in Computer Science and Engineering (Cyber Security) from Amrita Vishwa Vidyapeetham, Coimbatore.

DESCRIPTION:

This is a Music Streaming Application which is created using create data driven flask web app with SQLite for database management where a normal user can search for songs, genres, albums, creators and create playlists. Users who register themselves as creators can upload songs, albums, edit the uploaded songs and albums. Finally, there's an admin who has access to all the users, creators and their albums and songs. The CRUD interface helps the admin to access, delete and update details regarding users, creators, ext.

TECHNOLOGIES USED:

1) Flask:

Flask is a lightweight and flexible web framework for Python.

It is designed to be simple and easy to use while providing the necessary tools for building web applications.

Flask follows the WSGI (Web Server Gateway Interface) standard and is often used for building small to medium-sized web applications and APIs.

2)Popper.js:

Popper.js is a JavaScript library for positioning popovers and tooltips.

It provides a set of utilities to manage the positioning of elements that "pop out" from the flow of the document.

Popper.js is commonly used in conjunction with front-end frameworks like Bootstrap to enhance the user interface

3)jQuery:

jQuery is a fast and lightweight JavaScript library.

It simplifies things like HTML document traversal and manipulation, event handling, and animation.

jQuery is often used to simplify common tasks in JavaScript and to provide a cross-browser-compatible way of interacting with the Document Object Model (DOM).

4)Jinja2:

Jinja2 is a modern and designer-friendly templating engine for Python.

It is commonly used with web frameworks like Flask.

Jinja2 allows you to embed dynamic content within templates, making it easier to generate HTML dynamically based on data from your application.

5)Bootstrap:

Bootstrap is a popular open-source front-end framework.

It provides a collection of pre-styled components (such as buttons, forms, navigation bars) and a responsive grid system.

Bootstrap makes it easier to create visually appealing and responsive web pages without the need for extensive custom CSS.

6)SQLite:

SQLite is a lightweight, serverless, and self-contained relational database engine.

It is widely used for embedded database applications and is often chosen for its simplicity and ease of integration.

SQLite databases are stored in a single file and provide a SQL interface for interacting with data.

DATABASE DESIGN SCHEMA:

1. User Table:

- Fields: id, username, email, password, name, location, role, permission.

- This table represents users in your application. It includes information such as username, email, password , name, location, role, and permission.

2. Playlist Table:

- Fields: id, user_id, name.

- Represents playlists created by users. Each playlist is associated with a user through the user_id field. It has a one-to-many relationship with the Song table.

3. Song Table:

- Fields: Songid, Song_name, Song_Lyrics, Song_Artist, Song_genre, Song_rating, creator_id, album_id, flag, playlist_id.

- Represents individual songs. It includes information such as song name, lyrics, artist, genre, rating, creator_id (user who created the song), album_id (album to which the song belongs), flag, and playlist_id (the playlist to which the song belongs). It has relationships with the User, Album, Playlist, and Rating tables.

4. Rating Table:

- Fields: `id`, `user_id`, `song_id`, `rating`.

- Represents ratings given by users to songs. It includes information such as the user who gave the rating, the song being rated, and the actual rating value.

5. Album Table:

- Fields: `album_id`, `album_title`, `album_artist`, `songs`, `flag`.

- Represents albums. It includes information such as album title, album artist (associated with a user), songs in the album, and a flag. It has a one-to-many relationship with the `Song` table.

Summary of Connections:

- The User table is connected to the Playlist, Song, Rating, and Album tables through various foreign key relationships.

- The Playlist table is connected to the Song table through a one-to-many relationship.

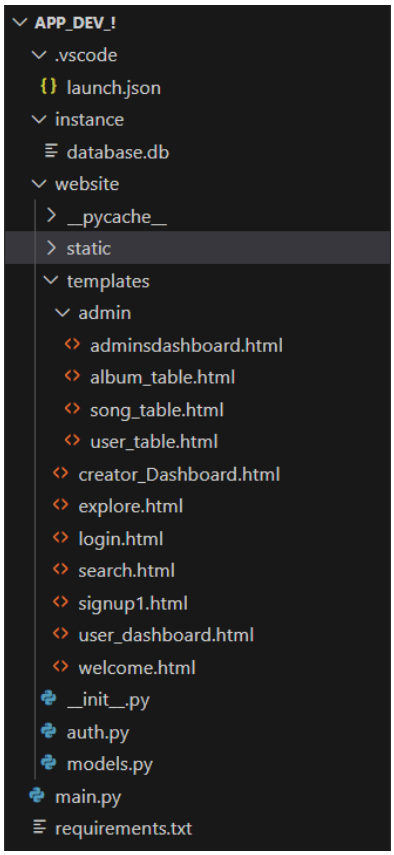
- The Song table is connected to the User, Album, Playlist, and Rating tables through various foreign key relationships.

- The Rating table is connected to the User and Song tables through foreign key relationships.

- The Album table is connected to the User and Song tables through foreign key relationships.

These connections create a relational structure where you can, for example, associate songs with users, albums, and playlists, and track ratings given by users to specific songs.

Views:



login.html: login page for user

signup1.html: sign up page

user_dashboard.html: user dashboard which contains details about the user and their playlsits

creator_dashboard.html: dashboard of users who have creator privilege

welcome.html: It is the home page for the project- has three entry points-admin log in, User log in and Sign Up as new user

explore.html: The page which has various song recommendations and through which songs can be added to one's playlist

search.html: The page where the user can search for any song ,album ,artist or genre and the results will be displayed.

admin folder:

- user_table.html: contains the album tables
- song_table.html: contains the album tables
- album_table.html: contains the album tables
- adminsdashboard.html: admin dashboard

__init__.py: Contains initialisation code

auth.py: This file contains the authentication-related logic for my web application. It contains routes, functions, or classes responsible for handling user authentication, login, logout, and related functionalities. It allows contains code which renders the html pages

models.py: The typically holds the data models for the application. It defines the structure of the database tables, including fields and relationships.

main.py: Serves as entry point for the project.

Video link:

https://drive.google.com/file/d/1rBeWe3qOXD81sFXvULdUByjsb6mnJyyx/view?usp=drive_link