Name: Aditya Shanker Srivastava

Roll No: 23f1001545

Student's Email: 23f1001545@ds.study.iitm.ac.in

Hello everyone I am Aditya Shanker Srivastava and I am from Lucknow, Uttar Pradesh. Write now I am a student and perusing BS in Data Science and Applications from IIT Madras. I did my intermediate with Physics, Chemistry, Mathematics from St. Xavier's Convent School, Lucknow in 2022.

DESCRIPTION

In this Music Streaming Application, we are supposed to create data driven flask web app with SQLite for database management that play songs, create playlist and albums for a particular user where user can also be a creator and upload songs and create a CRUD interface for admin who can create, update and delete songs.

TECHNOLOGIES USED - Flask: Flask is a micro-web framework with little to no dependencies to external libraries. This makes the framework light with little dependency to update and watch for security bugs, which is best suited for beginners. When we install flask some default extensions of flask also get installed in the machine, for example;

• Click: It's the "Command Line Interface Creation Kit". It is a python package for creating beautiful command line interfaces in a composable way with as little code as necessary.

• ItsDangerous: It helps in security, when some data is sent to untrusted environments, then get it back later. To do this safely, the data must be signed to detect changes. We can assign a key which helps to get the data without any changes.

• Jinja2: Jinja2 is extensible templating engine which have a very similar coding syntax like python, which is very easy to work with. And its logical and arithmetical operations are very useful to manage the data for a particular page.

• MarkupSafe: MarkupSafe escapes characters so text is safe to use in HTML and XML. Characters that have special meanings are replaced so that they display as the actual characters. This mitigates injection attacks, meaning untrusted user input can safely be displayed on a page.

• Werkzeug: It is a collection of libraries which is used to create web server gateway interface. It helps in requests processing and routing.

Flask-SQLAlchemy - Flask-SQLAlchemy is an extension for Flask that adds support for SQLAlchemy to flask application. It simplifies using SQLAlchemy with Flask by setting up common objects and patterns for using those objects, such as a session tied to each web request, models, and engines. In this application we are going to use.

• SQLite: It is a self-contained full featured embedded database in file format, with no client-server model which doesn't need separate process to run the server, by which we can run it in the application itself. And as it is a file, we can delete, copy or retrieve this anywhere.

Bootstrap: It is used for making the body of the application, because bootstrap provide inbuilt JavaScript plugins and default CSS, which lessen the efforts to make a good-looking responsive website.

DB SCHEMA DESIGN - For this web application I have made 5 entities:- User, Album, Song, Playlist and Rating. In this schema I have made one to many relationship, where user can have multiple playlists and albums, albums can have multiple songs, playlists can have multiple songs.

ARCHITECTURE AND FEATURES - In my project folder named as “23f1001545”, there are 2 folders “templates” and “static” and 4 files “app.py”, “database.sqlite3”, “ReadMe.txt” and “requirements.txt”. A. “templates” folder contains 14 HTML files;

1. “base.html”: It have the code for navbar which contains a search bar and logout button. It will be extended for every other page.

2. “index.html”: This is the first page, where it shows admin login, login and signup.

3. “adminlogin.html”: This page is for entering admin username and password.

4. “admin_dashboard.html”: This page is for admin, with admin dashboard to monitor app statistics like total users/creators/albums, statistics on how songs are performing, ability to flag or remove songs/albums that breach company policy.

5. “home.html”: This page is to show the play songs, song name, song artist, song rating, song lyrics and the functionality to rate a song, create a playlist, add songs to the playlist and to register as a creator.

6. “signup.html”: This page is for the new User to sign up with the email id, username, full name and password.

7. “login.html”: This page is for the registered user to login with the help of username and password.

8. “playlist.html”: This page is to show the songs added in a playlist, to remove a song for the playlist and to delete the playlist.

9. “c_dashboard.html”: This page is for the creator’s dashboard to create an album with album’s name, artist name S. No, album id and shows the song count, album rating and total number of albums and to update, delete, and add a song.

10. “flagsongs.html”: This page is to delete the songs.

11. “search.html”: This page is to search the song with song name and genre.

12. “updatealbum.html”: This page is to update the album name and artist.

13. “updatesong.html”: This page is to update song in the album.

B. “static” folder contains an image folder that contains all the images used in the web app.

C. “database.sqlite3” is an SQLite 3 file for the data where the data will be stored and retrieved.

D. “ReadMe.txt” is made for the users who want run the code in their machine with detailed explanation to ‘How to run this web app?’.

E. “requirements.txt” contains all packages details that need to be installed of import.

VIDEO LINK - https://drive.google.com/file/d/1318KEAj5Fb-JgMTYWF8TavEj-auZkUkP/view?usp=drivesdk