Author

Karan Narayan Patil
Roll No: 22f2001061
Email: 22f2001061@ds.study.iitm.ac.in
I am extremely interested in Computer Science and Programming. I like learning new technologies related to this field and this course is one of those.

Description

This project is highly dependent on CRUD operations on Shows/Venues which are the core factors in the Models. Other part deals with the User/Admin) Interaction with the Show/Venue . Those functionalities are essential for the application.

Technologies used

- Flask- for app MVC, (Click, Werkzeug,etc that comes with Flask)
- Flask-SQLAlchemy, sqlalchemy- for db operations
- Jinja 2 - Dynamic Templates
- PIL for image manipulation
- Matplotlib for plotting graphs in summary
- Sqlite extension FTS5 for full text search

DB Schema Design

- Guideline : ___ keys, **bold UNIQUE,** *italic number(int/float))*

Admin(admin_id, **admin_name**, ***password***, ***email***, mobile)
User(user_id, **username**, mobile, ***email***,***password***)

Show(show_id, title, stat_time, end_time, *rating*, description, *ticket_price*, *fill_count*, **venue_id**)
Venue(venue_id, name, place, *capacity*)

Rel: Venue-Shows (One venues can have many shows at it)
Venues has attribute that relates to shows where show backref to venue_id

Rel: Bookings-venue-show-user(Ticket booking for given show at given venue for the user)
Booking(***bid***, user_id, show_id, venue_id, show_title, venue_name, start_time, end_time, *num_seats*, *user_rating*)
Rel: Tag- Show-tags (one show can have many tags)
Virtual Tables using fts5:
Search_city(show_id, title) : to search using venue or City name
Search_show: to search using show title

API Design

All the REST Api are listed into the YAML file(TicketShowApi) which contains CRUD on Venue and Show Model

Architecture and Features

I have created the Main project package where all the different functionalities for the app to be collected together. The app_<app_name> are the packages which act as a separate app for different functionality in this way it is easily manageable for the developer. All the requirements have independent implementation in the app so that modifications are not the hassle for the main application each time we update somethin. Also we can extend this by adding new functionalities with the design structure

The app_admin package manages CRUD api for the Shows and Venues . The admin model contains details for admin we can only add admin through command line for app hence no one outside the team can use admin functionalities Here dashboard and summary views are implementate in admin_crud module which include CRUD view on Venues and CRUD views on Shows and admin authentication is defined in auth module it handles login and register part for admin.

The app_show_manager handles bookings , general views and detail pages. All the core models and relations defined in models.py module which then are used across modules to query the database. The query interface uses flask sqlalchemy's Legacy Query API and also some sqlalchemy query api too. Here the bookings view, rating views, search view, and the Home view for the user

The app_user handles authentication and model definitions for the User. and User model is also implemented in the app_user. The auth module it handles the authentication for the user.

Video Link:

https://drive.google.com/file/d/1aB0hpBgNfxcmKiLVEh7GG62KIKbldoHU/view?usp=share_link