Ticket Show Application

Author

Name :- SANKALP SHRIVASTAVA
Roll No. :- 21f1006134
Stu. Mail :- 21f1006134@ds.study.iitm.ac.in

I am a full time student of IITM BS Degree program. I am in the diploma level. I have finished my diploma in data science and am currently pursuing my diploma in programming. I am also working as a data science intern at Tata Communications Limited.

Description

Ticket Show is a ticket booking platform where users can book shows of different venues and admin can add shows and venues in it. This is a multiuser but single admin application.

Technologies Used

Flask :- Flask is a web framework written in python. It is used in the backend of the Ticket Show.

Flask-SQLAlchemy :- Flask-SQLAlchemy is a Flask extension that makes using SQLAlchemy with Flask easier.It is used in the database of the Ticket Show App.

Flask-Bcrypt :- Flask-Bcrypt is a Flask extension that provides hashing utilities for your application. It is used in hashing the passwords.

Matplotlib :- Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. It is used in visualizing summary in Application.

Html | Css | Jinja | Bootstrap :- These are the basic building blocks of any application. First three are used to make the application templates and minor Bootstrap is used for styling.

DB Schema Design

There are 4 tables in the database. These are UserLogin , Venues, Shows and Bookings.
UserLogin for storing the data about users, Venues for storing the data for venues and Show and Bookings for storing the data of shows and booking respectively. Schema of the tables is given Below.

UserLogin Table:

- user_id: auto-incremented unique identifier for each user
- user_name: username of the user
- user_mail: email of the user
- user_uname: login username of the user
- user_pass: password of the user

Bookings Table:

- booking_id: auto-incremented unique identifier for each booking
- booking_user: references user_id
- booking_venue: references venue_id
- booking_show: references show_id
- booking_count: number of tickets booked
- booking_total: total price of the booking

Venues Table:

- venue_id: auto-incremented unique identifier for each venue
- venue_name: name of the venue
- venue_place: place of the venue
- venue_location: location of the venue
- venue_capacity: capacity of the venue

Shows Table:

- show_id: auto-incremented unique identifier for each show
- show_name: name of the show
- show_venue: references venue_id
- show_rating: rating of the show
- show_date: date of the show
- show_time: time of the show
- show_tags: tags associated with the show
- show_price: price of the show
- remaining_cap: remaining capacity of the show

Architecture and Features

In the project folder, an app.py file is present along with the templates and static folder. All the backend part of the application is written in app.py. Templates and static folders contain HTML templates and images used in the application respectively.
Starting from the home page, one can login as a user (Signup in case of new user) or login as an admin(with admin credentials).
After login as admin, admin can create new venues, edit them and remove them. After creating a venue, the admin can add shows in the venue , edit them and remove them. Admin can click see the summary of the venue and shows by clicking on the summary page.
After login as a user, users can see the shows of different venues, and book them. Users can see all the bookings in their profile, they can cancel their bookings from there if needed.
Both can logout after doing their respective tasks.

Video

Project Video