Ticket Show Booking Web Application

Author

Paidisetty Sai Amrutha
22f1000029
22f1000029@ds.study.iitm.ac.in
I am currently pursuing B-Tech 3rd year at VNRVJIET ,Hyderabad .I am a technophile with motivated attitude ,passionate about drawing conclusions from data.

Description

A web application (Ticket Show) where admin has the privilege to add venues and create shows and then these shows can be assigned to venues .The user can view all the shows in different venues and can book tickets for a particular show in a venue of their choice.

Technologies used

Flask - For building web applications, handling HTTP requests, routing, rendering templates.
Flask-SQLAlchemy - For connecting to relational databases from Python,using Python classes and objects.& querying database.
Flask-RESTful - For building RESTful APIs with Flask, define API resources and handle HTTP methods (GET, POST, PUT, DELETE, etc.) for these resources.
Flask-Login- For user authentication and session management, for easy integration of login and logout functionality in Flask applications.
Flask-WTF - define and validate forms in Flask applications, and handles form submissions .
WTForms - For working with web forms in Python. It provides a set of classes for defining form fields, validators, and rendering form HTML.
Flask-CORS - For Cross-Origin Resource Sharing (CORS) support. controls which domains are allowed to make requests to the Flask application,useful for securing APIs.
Flask-BCrypt - for securely hashing passwords and verifying password hashes.
Jinja2,Html,Bootstrap,Css- Frontend design and Styling.

DB Schema Design

User: Represents users who can register and log in to the system. It has columns- id (primary key), username (string with maximum length of 20 characters, not nullable, and unique), and password (string with maximum length of 50 characters, not nullable).

Venues: Represents venues where shows can be booked. It has columns- venue_id (primary key), venue_name (string with maximum length of 20 characters, not nullable), place (string with maximum length of 20 characters, not nullable), location (string with maximum length of 20 characters, not nullable), capacity (integer, not nullable), and s_rel which represents a many-to-many relationship with the Shows table using a secondary table called venue_shows. This relationship is defined by the backref attribute which creates a reverse relationship from Venues to Shows.

Shows: Represents shows that can be booked at different venues. It has columns - show_id (primary key), show_name (string with maximum length of 20 characters, not nullable), rating (integer, not nullable), start_timing (time, not nullable), end_timing (time, not nullable), tags (string with maximum length of 20 characters, not nullable), price (integer, not nullable), and v_rel which represents a many-to-many relationship with the Venues table using the venue_shows secondary table. It also has a foreign key svenue_id which references the venue_id column in the Venues table.

Venue_shows (Association Table):Represents the many-to-many relationship between Venues and Shows. It has columns - venue_id (foreign key referencing the venue_id column in Venues table, primary key) and show_id (foreign key referencing the show_id column in Shows table, primary key).

Bookings: Represents bookings made by users for shows at different venues. It has columns - bk_id (primary key), bkuser_id (foreign key referencing the id column in User table), bkvenue_id (foreign

key referencing the venue_id column in Venues table), bkshow_id (foreign key referencing the show_id column in Shows table), bkvenue_name and bkshow_name (string with maximum length of 20 characters, not nullable), bkshow_stiming and bkshow_etiming (time, not nullable), n_tickets and tot_price (integer, not nullable).

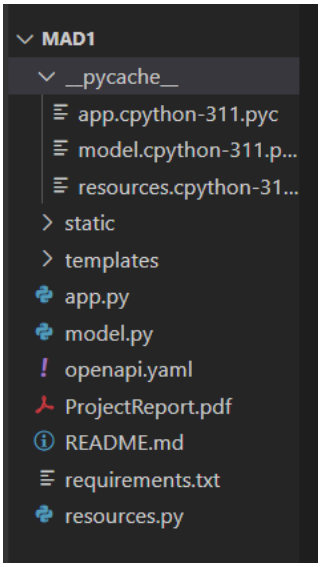
API Design

Apis are implemented for Venues ,it includes:
"/api/all_venues"
All_Ven_Api
GET- displays all venues
POST- to create a new venue
“/api/venue/<int:id>”
Ven_id_Api
GET - retrives venue with id given if not present display error message
PUT - updates venue with id given if not present display error message
DELETE – deletes venue with id given if not present display error message
Yaml File link:
https://drive.google.com/file/d/1-ByVG6OqDQsAcOy1VhTMkBwdOVX_XQeW/view?usp=sharing

Architecture and Features

The project consists: **pycache**, **static**, and **templates**
Three Python files: **app.py**, **db1.py**, and **resources.py**.
The static folder contains images and CSS files
The templates folder contains HTML files.
Features:
- Creating venues and shows.
- Updating and deleting venues and shows.
- Search for venues.
- Visualisation of ratings of shows.
- Book shows at venues of your choice.
- Search for shows you booked.
- Different logins for admin and users.

Additional Features:
- CRUD on APIs for venues



Video

Link to Presentation Video:-
https://drive.google.com/file/d/1PhnDiKTE_dNeo3AIUlRSck3KB5atA32L/view?usp=share_link