TICKET SHOW

Author

Mallela Karthik Reddy
21f2000776
21f2000776@ds.study.iitm.ac.in
I am currently pursuing B-Tech 3rd year in the stream of Information Technology at VNRVJIET and B.S degree in Data Science and Applications at IIT Madras.

Description

TICKETSHOW is a user-friendly web application that allows users to book movie tickets quickly and easily. The application provides a seamless experience, enabling users to browse movies and venues, select seats with ease.

Technologies used

Flask for application code

- Flask-RESTful is an extension for Flask that adds support for quickly building REST APIs. It is a lightweight abstraction that works with your existing ORM/libraries.
- Flask-SQLAlchemy is an extension for Flask that adds support for SQLAlchemy to the application. It simplifies using SQLAlchemy with Flask by setting up common objects and patterns for using those objects.
- Flask-Login provides user session management for Flask. It handles the common tasks of logging in, logging out, and remembering your users' sessions.
- Flask-WTF is a built-in module of the flask which provides an alternative way of designing forms in the flask web applications.
- Flask-Bcrypt is a Flask extension that provides hashing utilities for your application.

Jinja2 templates + **Bootstrap** for HTML generation and styling
SQLite for data storage

DB Schema Design

This application uses **5** tables in an **SQLite** database named **database.sqlite3**
Table Name: **Users**

- user_id: integer (primary key), username: string (20 characters, not nullable, unique)
- password: string (80 characters, not nullable), user_type: string (20 characters, default 'user')

Table Name: **Venues**

- venue_id: integer (primary key), venue_name: string (50 characters, not nullable),
- venue_location: string (50 characters, not nullable), venue_capacity: integer (not nullable)

Table Name: **Shows**

- show_id: integer (primary key), show_name: string (50 characters, not nullable)
- show_date: date (not nullable),show_time: time (not nullable)
- show_tags: string (50 characters, not nullable),show_rating: integer (not nullable)
- show_price: integer (not nullable),svenue_id: integer (foreign key referencing venues.venue_id)

Table Name: **Association**

- venue_id: integer (foreign key referencing venues.venue_id, primary key)
- show_id: integer (foreign key referencing shows.show_id, primary key)

Table Name: **Bookings**

- booking_id: integer (primary key), buser_id: integer (foreign key referencing users.user_id)
- bvenue_id: integer (foreign key referencing venues.venue_id), total_price: integer (not nullable)
- bshow_id: integer (foreign key referencing shows.show_id), num_tickets: integer (not nullable)

Constraints:

- The username column in the Users table should be unique, meaning that no two users can have the same username.
- The Association table is used to create a many-to-many relationship between Venues and Shows tables, using their respective primary keys as foreign keys.

ER Diagram link:
https://drive.google.com/file/d/1mFkXktFDQ28mCqPccGid4Ai6HithnF_7/view?usp=sharing

API Design

HTTP methods used in this project are:

Api_Venues:

- GET - used to retrieve all venues.
- POST - used to add a new venue to the database.

Api_Venue:

- GET - used to retrieve a specific venue details by ID.
- PUT - used to update an existing venue by ID.
- DELETE - used to delete an existing venue by ID.

Search_Venue:

- GET - allows for searching venues by name (partial match).

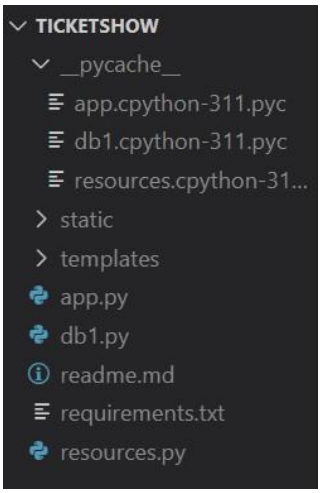
The API endpoints are:

/api/venues and **/venues**, **/api/venues/search/<string:id>, api/venue/<int:id>**

Yaml file link:
https://drive.google.com/file/d/17vOjv_hsk1c6TlGELu05g4GfX_d60q5A/view?usp=sharing

Architecture and Feature

The project consists of three folders: **pycache**, **static**, and **templates**, and three Python files: **app.py**, **db1.py**, and **resources.py**. The static folder contains images and CSS files, while the templates folder contains HTML files.



Feature Implemented:

- Admin login and User login
- Venue Management and Show Management
- Booking show tickets
- Search for shows/venues

Additional Features:

APIs for interaction with venues

- CRUD on venues

Validation

- All form inputs fields - text, numbers, dates etc. with suitable messages

Video

Link to the presentation video :
https://drive.google.com/file/d/1YD9XgEEEsWU_rpfdfPVICDZRW1Poubli/view?usp=sharing