Author

Gautam Chirag Gala
21F1003495
21f1003495@ds.study.iitm.ac.in
CSE undergraduate with a strong interest in finance, technology, and data-driven decision-making and problem-solving.

Description

The movie ticket booking system allows admins to create, update, delete, and read operations on the venue and show tables within the backend. On the other hand, users can reserve tickets and conveniently search for various shows and venues through the system. Here the main purpose is the usage of advanced technologies.

Technologies used

- DateTime: Used for working with date and time data, particularly timestamping ticket purchases.
- Flask: A micro web framework in Python that handles routing, request handling, and response generation.
- render_template, request, redirect, url_for, jsonify, Response: These are various components from Flask used to render HTML templates, handle incoming requests, perform redirects, generate URLs, jsonify data, and create custom HTTP responses.
- CORS (Cross-Origin Resource Sharing): A Flask extension to manage cross-origin requests and responses, ensuring secure data exchange between the front and backend.
- SQLAlchemy: An Object-Relational Mapping (ORM) library that simplifies database interactions by providing a high-level interface to work with databases using Python classes.
- Flask-Login: A Flask extension that manages user authentication, user sessions, and user roles.
- pytz: A Python library for working with time zones, used here to handle time zone conversions for timestamps.
- Flask-APScheduler: An extension that integrates the APScheduler library with Flask, allowing the scheduling of tasks at specified intervals.
- smtplib: A standard Python library used for sending emails using the Simple Mail Transfer Protocol (SMTP).
- email.mime.multipart, email.mime.text: These are sub-modules from the standard Python email library, used for constructing MIME (Multipurpose Internet Mail Extensions) messages for sending emails.
- CSV, io: Python's built-in libraries for working with CSV files and I/O operations on streams.

DB Schema Design

My database has four tables:

1. Show
    a. Id - primary key and identify a show uniquely which increments automatically
    b. Name - Name of the show which cannot be null
    c. Description - Description of the show i.e. actors or About the movie
    d. Rating - categorise show based on content permissible to age
    e. Tag - order the show based on the genre of content
    f. DateTime - Date and time of the show in the venue which cannot be null
    g. Venue_id - venue in which the show is created, foreign key (venue. id), which cannot be null
2. Venue
    a. Id - primary key and identify a venue uniquely which increments automatically
    b. Name - Name of the venue which cannot be null
    c. Location - city of the platform which cannot be null
    d. Capacity - seating capacity of the platform which cannot be null
3. User
    a. Id - primary key and identify a user uniquely which increments automatically
    b. Name - Name of the user which cannot be null
    c. Email - The email of the user which cannot be null but has to be unique
    d. Password - password of the user which cannot be null
    e. Role - Role of the user: {admin, user}
4. Ticket
    a. Id - primary key and identify a ticket uniquely which increments automatically
    b. User_id - Id of the user that has booked the ticket, foreign key, which cannot be null
    c. Show_id - the show for which the ticket has been booked, foreign key, which cannot be null
    d. Date_purchased - datetime at which tickets have been booked with timezone as Kolkata, Asia

API Design

All the API required are in a file called main.py. The api uses the database design defined above to perform CRUD operations on the data.

Architecture

- The project root folder contains the database file and main.py i.e. the primary file for running the backend.
- In the frontend folder, we have src, views, components, router, assets and different files which help to render the frontend of the system built using Vue.js.
- The docs folder has the document for the project and a readme file.

Features

- The system is designed with an aim for admins to perform CRUD operations, basically create, read, update, and delete on shows and venues.
- The user can book tickets, and download CSV files for shows and venues.

- The system sends monthly reports and daily reminders to user's email ID
- Admin and user can login through the same form.
- At the top there is a navigation bar to navigate across the system.
- Styling has been taken care of to an extent.
- User information is cached in the browser's local storage to improve performance.

Video

LINK - https://drive.google.com/file/d/1QIEZM2qu3LX0Qj-0a8EaBe0Zb26XTLwr/view?usp=sharing