Author

Sree Narayanan
22f1000950
22f1000950@ds.study.iitm.ac.in
Hi there, my name is Sree and I'm a Computer Science student studying at PSGCAS Coimbatore. As someone who is passionate about technology and its potential to change the world, I'm excited to be pursuing a degree in this field. I'm constantly learning and exploring new concepts and ideas, and I'm eager to see where this path will take me in the future.

Description

In this project, the goal is to build a movie ticket booking system using Flask and Celery. The Flask app manages user authentication, booking records, and sends reminder emails to users who haven't booked in a while. Celery is used to handle asynchronous tasks, such as sending emails, with Redis as the message broker. Vue.js is utilised as a frontend JavaScript framework, enabling dynamic and interactive user interfaces.

Technologies used

1. Flask: A lightweight Python web framework for building the backend of the movie ticket booking system.
2. Vue.js: Vue.js is a popular progressive JavaScript framework used for building user interfaces. It emphasises the development of interactive, dynamic, and responsive web applications.
3. Celery: An asynchronous task queue system for managing background jobs like sending emails.
4. Redis: A message broker used by Celery to handle task distribution and communication.
5. SQLAlchemy: An Object-Relational Mapping (ORM) library for working with the database.
6. Flask-JWT-Extended: An extension for Flask to provide JSON Web Token (JWT) based authentication.
7. Flask-Bcrypt: An extension for Flask to handle password hashing and authentication.
8. Flask-CORS: An extension for handling Cross-Origin Resource Sharing (CORS) in the Flask app.
9. SMTP: Simple Mail Transfer Protocol for sending emails.

DB Schema Design

1. The user table contains information about users, including their unique email, name, password, and language. The primary key used is email.

2. The venue table contains information about venues, including their unique venue_id, name, and location. The primary key used is venue_id.
3. The shows table contains information about movie shows, including their unique show_id, the associated venue_id, the ticket price, the number of available seats, and the show timings. The primary key used is show_id, and there is a foreign key referencing the venue tables.
4. The booking table contains information about bookings made by users, including their unique booking_id, the associated user_email, venue_id, show_id, and seat number and booking_date. The primary key used is booking_id, and there are foreign keys referencing the user, venue, and shows tables.

API Design

/register - Register a new user account.
/login - Log in to an existing user account.
/logout - Log out the currently logged-in user.
/filter - Shows page filter of shows.
/bookticket - Book tickets for a specific show.
/booking/get - View the user's booking history.
/venue/get - Get the list of venues.
/venue/create - Create a new venue.
/venue/update - Update details of the given venue.
/venue/delete - Delete a venue.
/show/get - View a list of available movie shows and details.
/show/create - Create a new show for a venue.
/show/update - Update details of the given show.
/show/delete - Delete a show.
/export - Export data to a CSV file.

Architecture and Features

The Project can be divided into four sections. The database file, the static folder which holds static files such as images, CSS, and fonts, the template folder which has the HTML files of different endpoints and finally the main.py file which is the web application.
Core Functionality:

1. Multi-user functionality with session-based logins. The sign-up page has multiple client side and server-side validations before signing up a user. The passwords are then bcrypted and stored as a hash into the database. When a user logs in, then an encrypted session token will be generated and stored as a cookie in his browser. This is then removed when the user logs out.
2. Separate login page for administrative users. Upon logging in, it gives the admin access to the administrative dashboard. In the dashboard, an admin can manage venues and shows.

3. Venue Management System – An admin user can create a new venue, read, and update its details, and delete venues altogether.
4. Show Management System – Within each venue, an admin user can create new shows, read, and update current show details, and delete shows from a particular venue.
5. Using the search bar, a client can type the name or title of a movie and search for results (shows) containing that string.
6. A user can book for shows and select available seats in the venue. During the booking process, a user can see if a seat is available or not. And upon confirming the seats, the users will be able to see the full price and confirm booking.
7. User will be able to see his booked tickets on the booking page.
8. At the start of every month, the user will receive an email about the monthly entertainment report, which is an html with all the shows booked at the previous month.
9. Checks for the last booking period of the user and sends an email accordingly.
10. Create a csv export of details of a venue for admins.

Video

https://drive.google.com/file/d/1GQa0iKpiffGDsr-QONL02BNWcOYD33oj/view?usp=sharing