Author

Richik Majumder
21F1000460
21F1000460@ds.study.iitm.ac.in
I am from West Bengal, Kolkata. I have completed B.Tech in Mechanical Engineering in 2019 and currently pursuing IITM BS in Data Science and Applications as a standalone course

Description

A ticket booking app should provide a convenient, user-friendly, and efficient way for users to purchase tickets to their favourite shows. It is used to simplify the process of booking show tickets.

Technologies used

Applications:

- **VS Code** for coding
- **DB Browser** for viewing database
- **Insomnia** for api testing
- A few more

Python packages:

- **flask** for backend code
- **flask_sqlalchemy** for database model
- **flask_jwt_extended** for token based authentication
- **flask_bcrypt** for hashing passwords
- **celery** with **redis** for task scheduling and caching
- **pdfkit** for generating PDF out of jinja2 templates
- **imgkit** for generating png files out of jinja2 templates
- **qrcode** for generating qr codes
- **cv2** for compressing png file
- A few more mentioned in requirements.txt

Others:

- **VueJS** (CLI) for frontend code
- **Bootstrap** and **CSS** for designing the pages
- **Alertify.js** for frontend notifications
- **date-fns** for formatting date and time
- **Swagger** and **Insomnia** for Api documentation

DB Schema Design

The database consists of seven tables: **Venue, Show, Link, User, Booking**, **Rating** and **PasswordResetToken**. Each table has a unique primary key, auto-incrementing integer ID column.

The **Venue** table has columns for the name, place, location, capacity, and a relationship with the Link and Booking tables. The **Link** table has foreign key relationships with the Venue and Show tables, as well as columns for the show time, ticket price, and tickets left.

The **Show** table has columns for the name, rating, tags, and a relationship with the Link and Booking tables. The **Booking** table has foreign key relationships with the Show, User, and Venue tables, as well as columns for the number of tickets booked, the ticket price, and the user's rating of the show.

The **User** table has columns for the username, password, first name, last name, last login time, wallet balance, whether the monthly report is required in pdf or not and whether the user is an admin.

Finally the **PasswordResetToken** table is used to store tokens used to change passwords in case the user/admin forgot his/her password.

The tables have constraints such as primary keys and foreign keys, which ensure data integrity and consistency. The Link table's foreign keys ensure that every link is associated with a venue and a show, while the Booking table's foreign keys ensure that every booking is associated with a user, a venue, and a show.

Architecture and Features

All the api and backend jobs are located inside “app.py” and the code for the database is located inside “models.py” and all the templates are located inside the “templates” folder. Static files are generated in the “files” folder and most of the email,pdf,image,qr,compression tasks are located in the “mail.py” and all the frontend codes are available in the “src” folder. Features like signup for users/admin , login for users/admin, logout for users/admin, create/update/delete venues for admin based login, create/update/delete shows for admin based login, linking/delinking/updating shows to venues for admin, booking show tickets, booking multiple show tickets up to the number of tickets available, stop accepting bookings in case of a housefull, searching shows based on venue name/show name/location and filtering search results based on ratings/tags, booking tickets based on location based search on venues, allowing users to check their booking history inside the user dashboard and rate shows (once rated, the ratings for that particular show will automatically be updated) are the default features that are implemented.

Apart from the default features, features like **dynamic pricing** is implemented for show ticket booking, which results in a 10% increase of the ticket price for every 10% of total seats booked for a particular show (i.e 10% increase in price), **updating profile information**, **forgot/reset password**, **wallet**, **show/venue stats** is also implemented.

Video

https://drive.google.com/file/d/1C77YolF-g9DKJPOlJeqfiDpXWw-Va8f3/view?usp=sharing