Author

Snehil Nair
21f2000602
21f2000602@ds.study.iitm.ac.in

I am an amicable lisztomaniac who is always on the lookout for new opportunities to learn. I have and inquisitive mind and am always ready to try something new.

Description

Create a Web based Ticket Booking app. Users can signup and login and based on role, Admin can do CRUD operation on Theatres, Movies and Shows (where a Show is defined as an instance of a movie and a theatre at a particular time slot) and general Users can search and book shows for movies.

Technologies:

- Python and Flask: (like Flask Security too, Flask Sqlalchemy, Flask jwt extension, Flask Caching etc. as described in the requirements.txt) to handle backend processing.
- Redis: As a message broker for celery workers and for caching
- Celery: For generating task queues and scheduled jobs.
- Vue: For Implementing a reactive frontend with javascript

DB Schema Design

The Schema description follows the following design:

Attribute Name	Type	Constraints

Users:

id	Integer	Auto-Increment, Primary Key
username	String	Unique, Not Null
email	String	Unique, Not Null
password	String	Not Null, Hashed
active	Boolean	Not Null, Default True

Roles:

Id	Integer	Auto-Increment, Primary Key
Name	String	Unique, Not Null

Theaters:

Id	Integer	Auto-Increment. Primary Key
userid	Integer	Foreign Key (Users)
name	String	Not Null, Unique
location	String	Not Null
capacity	Integer	Not Null

Movies:

Id	Integer	Auto-Increment, Primary Key
userid	Integer	Foreign Key (Users)
name	String	Not Null, Unique
rating	String	
userRating	Float	
tags	String	
duration	Time	Not Null

Shows:

id	Integer	Auto-Increment, Primary Key
theaterid	Integer	Foreign Key (Theaters)
movieid	Integer	Foreign Key (Movies)
title	String	Not Null
caption	String	
start_on	DateTime	Not Null

end_on	DateTime	Not Null
s_time	Time	Not Null
e_time	Time	Not Null
price	Float	Not Null
active	Boolean	Not Null, Default True

Available:

Id	Integer	Auto-Increment, Primary Key
showid	Integer	Foreign Key (Shows)
date	DateTime	Not Null
s_time	Time	Not Null
e_time	Time	Not Null
availableseats	Integer	Not Null

Bookings:

Id	Integer	Auto-Increment, Primary Key
userid	Integer	Foreign Key (Users)
bookingdate	DateTime	Not Null
bookedshow	Integer	Foreign Key (Available)
showid	Integer	Foreign Key (Shows)
theatername	String	
theaterlocation	String	
moviename	String	
bookedshowdate	DateTime	Not Null
bookedshowtime	Time	Not Null
seats	Integer	Not Null, Default 1
total	Float	Not Null

The database is designed with the following concept in mind: each show is an instance of a Movie hosted by a Theater during a specific time slot. Each show can have multiple instances (Eg: a show running everyday at 9) and hence each instance can be booked given constraints are satisfied. There is a secondary association table to map users to Roles for RBAC.

API Design

The API supports all core features required by the application including all admin CRUD operation and User actions. Except for Signup and login, all other routes require authorization token. The YAML file was made using Insomnia and exported in insomnia YAML 4 format that apparently only works with the same program/application.

Architecture and Features

The Project Folder contains readme, Report, YAML and all primary python codes including the main.py, api.py, controllers, celery etc. The root folder also contains 5 folders: ***configs*** containing config files, ***generated*** containing intermediate generated files (like csv), ***resource*** that contains static resources, ***static*** containing static javascript files and components (contained in */static/components*) and ***templates*** containing html templates.

The features of the Application include:
User Signup and Login: Supports signup and login and generation of role based tokens
Role Based Access Control
Administrative CRUD: Supports CRUD operation for Theaters, Movies and Shows based on constraints
User Actions: Supports various user actions like searching for movies based on name, tags, rating, location etc. and booking shows in various theaters for the same.
Celery Based Tasks: Celery based scheduled tasks like reminders and monthly reports
Redis Based Caching: Redis Based caching for required segments of code for better performance

Video

Link to Video or Copy the following URL:
(https://drive.google.com/file/d/1sWO7V3uFCqOlcqCXd6XT0lcBp6n0rHvm/view?usp=sharing)