Author

- Name:- Harshavardhan Su
- Roll Number:- 21f1005522
- Student email:- 21f1005522@student.onlinedegree.iitm.ac.in
- About me:- I am an enthusiastic individual with a winning track record in competitive programming, national rank 1 in IEEExtreme, and a keen interest in web development.

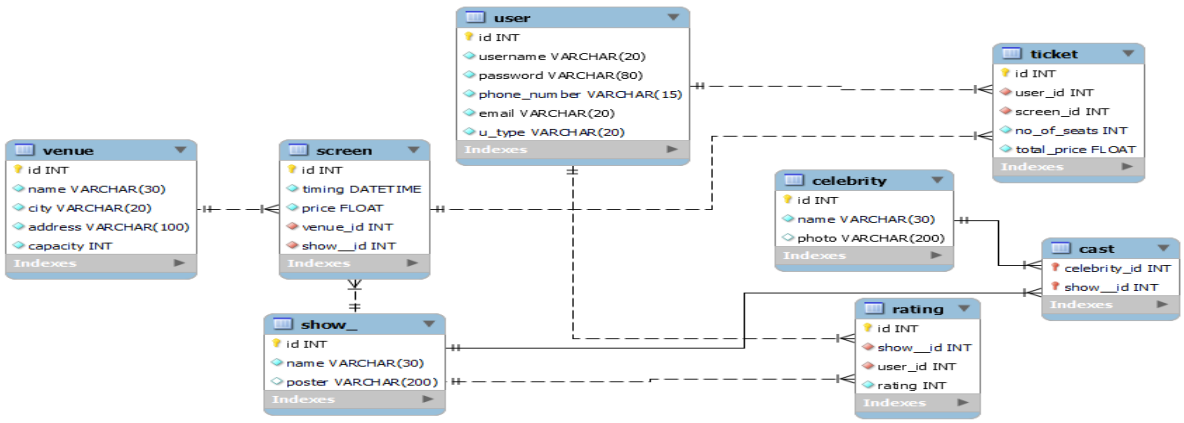
Description

A ticket booking website is to be developed using Flask and SQLite with the core functionalities: User signup and login, Admin Login, Venue Management, Show management, Search for Shows/Venues, Search for Shows/Venues, API for interaction with venues and shows, backend jobs using celery and caching using Redis.

Technologies used

- HTML, CSS, Bootstrap:- Frontend
- Flask:- Application code for Backend
- Vue js:- For frontend
- Celery and Redis - for scheduled tasks (backend jobs) and caching respectively
- Flask_bcrypt:- for hashing user password
- Flask_sqlalchemy:- Managing database operations using SQLAlchemy ORM
- SQLalchemy:- Python ORM for interacting with relational databases
- Jinja2:- For backend templates.
- Sqlite:- for database

DB Schema Design



- Separate tables have been created for user, venue, cast, show, screen, celebrity, ticket, and rating to avoid redundancy.
- Email field and phone number fields are unique as users must have a distinct email id and phone number.
- All the attributes except photo and poster are not null.

- Relationships between the table have been created with the help of foreign keys. All the created models are present in models.py

API Design

- Shows
    - ShowRelated
        - GET, PUT, DELETE - fetch, update, or delete a particular show's details
    - AllShow, AddShow
        - GET, POST - fetch all shows' details and create a new show respectively
- Screens
    - ScreenRelated
        - GET, PUT, DELETE - fetch, update, or delete a particular screen's details
    - AllScreen, AddScreen
        - GET, POST - fetch all screens' details and create a new screen respectively
- Venues
    - VenueRelated
        - GET, PUT, DELETE - fetch, update, or delete a particular venue's details
    - AllVenue, AddVenue
        - GET, POST - fetch all venues' details and create a new venue respectively
    - DisplayVenue
        - GET - fetch all venues' details along with their shows and timings.
- Flask_restful used for API integration with backend all API are stored in api.py file

Architecture and Features

- In the backend directory app.py, api.py, models.py, tasks.py. tasks.py contains all celery tasks, model.py contains all the database models and instances of the database. api.py inherits model.py and creates the controllers for api. app.py the main application code.
- In the frontend directory src has assets for storing static data, components for pages, and router for routing.
- **Admin and User Login:** Proper login system with protected view, frontend, and backend validation.
- **Venue Management:** Options for updating, deleting and adding venues
- **Show Management:** Options for updating, deleting and adding shows and cast
- **Screen Management:** Options for updating, deleting and adding screens
- **Book Show Tickets:** Users can books tickets for a screen with preferred show and venue.
- **Search:** Search for shows, venues, and venue locations.
- **Book Feed:** Webpage for users to book tickets for preferred show and venue.
- **API:** Flask_restful is used to implement api for shows, screens, and venues.
- **Rate Shows and view tickets:** Users can give ratings for the shows and view tickets.
- **Basic Statistics for admin:** Admin can view top 5 movies by rating and collection
- **Backend jobs:** Scheduled jobs (daily remainder, monthly report) and export jobs (csv)
- **Caching:** Caching was done for frequently used resources.

Video

Demo link