Ticket Tales (MAD-1 Project Report)

Author:

- **Name:** Vishesh Phutela
- **Roll No.:** 21f3001040
- **Email:** 21f3001040@ds.study.iitm.ac.in
- **About Me:** I love to Design Stuff, and this passion for designing also made me a graphic designer during my journey as Data Scientist I am also an avid gamer. I love to play competitive strategy games and first-person shooters. Lastly, I love learning. Every day I push myself to learn something new, whether that be about machine learning, software engineering, or miscellaneous facts about the universe.

Description:

Ticket Tales is a web application that allows users to book and manage tickets online. It is built using Flask, a Python web framework, and uses Jinja2 templates for frontend development. The application follows the Model-View-Controller (MVC) architecture pattern, Ticket Tales also features a RESTful API, which allows for easy integration with other systems.

Technologies used:

- **Python:** It is used as the base programming language i.e., to develop the controllers, API's etc.
- **Flask:** Flask is used to serve the web-app.
    - **Flask-Restful:** Flask-RESTful is a simple, easy to use Flask extension that helps you construct APIs.
    - **Flask-SQLAlchemy:** Providing you tools and methods to interact with your database in your Flask applications through SQLAlchemy.
    - **Flask-CORS:** Handling Cross Origin Resource Sharing (CORS).
- **HTML:** HTML is used to create the webpages for the app.
- **Jinja2:** Jinja2 Templating is used with html.
- **CSS:** CSS is used to stylize the app.
- **Sqlite:** Sqlite is used as a database engine.

API Design:

The Ticket Tales API follows RESTful principles and supports CRUD operations for the following resources:

users, venues, shows, bookings

The API uses the HTTP methods GET, POST, PUT, and DELETE to perform the CRUD operations. The endpoints are designed to be self-explanatory and follow the standard RESTful conventions.

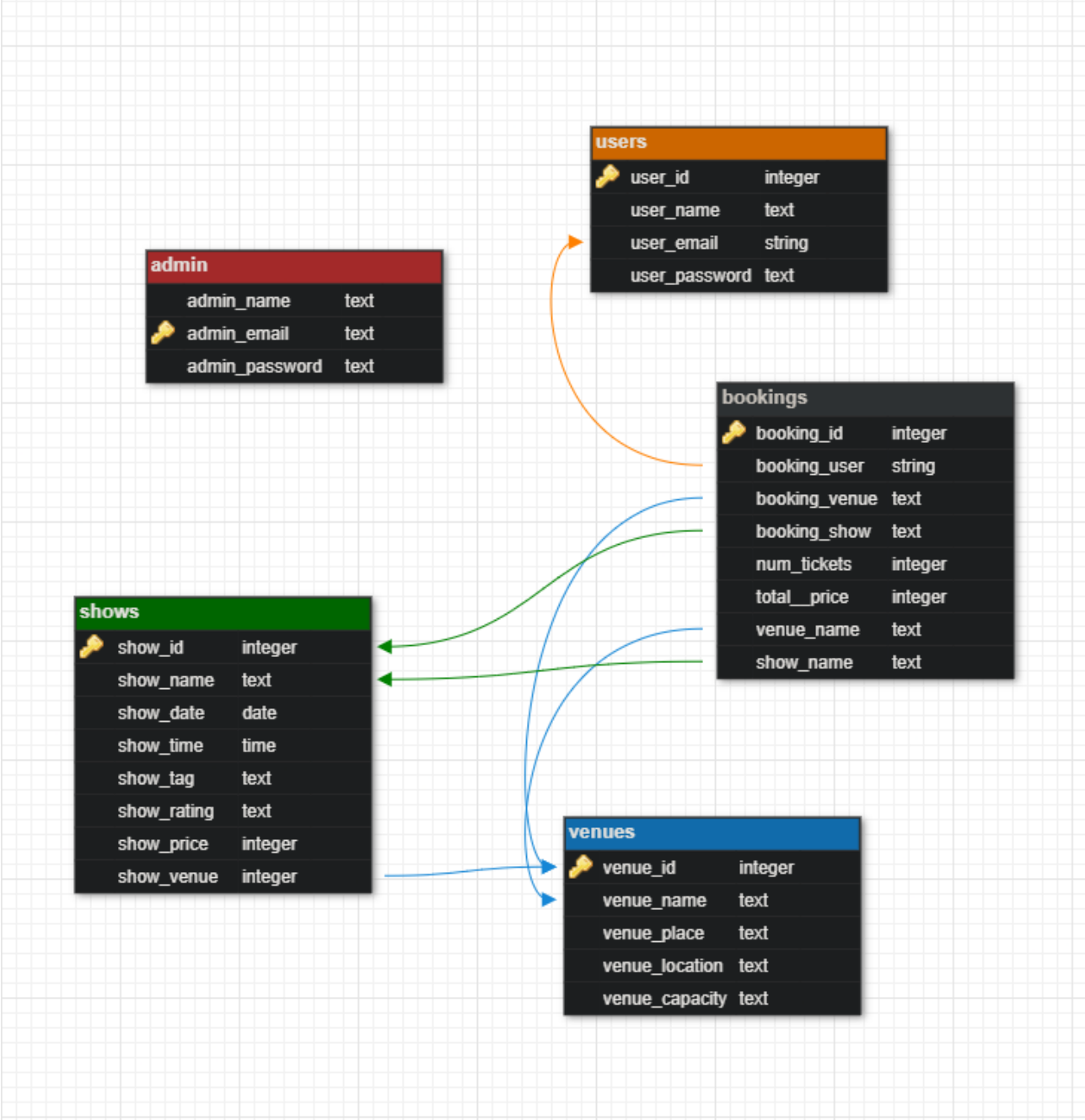
Architecture and Features:

The application follows the Model-View-Controller (MVC) architecture pattern, which helps to separate the presentation layer from the business logic and data storage. The Model represents the database schema and operations, the View is responsible for rendering the HTML pages, and the Controller handles the user input and business logic.

Features: Some feature of the App is as follows:

- Login and Register page for both admin and users
- Separate page for every venue
- User can book shows by opening the venue page and select the preferred show and book seats.
- App shows no available seats for new bookings in case of a House-Full.
- User can view his/her previous and upcoming bookings/shows in My shows tab.
- User can search on the basis of name of a show or a venue.
- User can also get an Invoice based on his/her bookings in My Shows tab.
- App is capable of handling multiple users and admins.
- An Admin can perform CRUD on both Shows and Venues.

Database Schema Design:



Video Link: video