Author

Aditi Lakshmi S
21f3001967
21f3001967@ds.study.iitm.ac.in
I am a passionate student and a highly motivated person who enjoys taking on challenges.

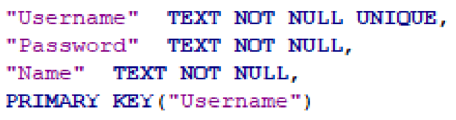
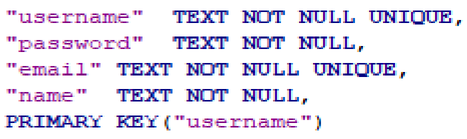
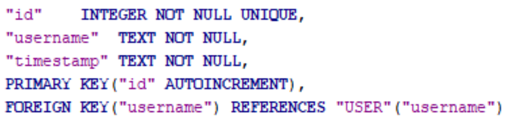
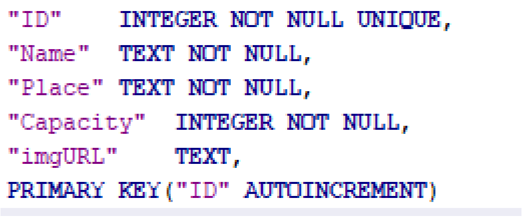
Description

We are expected to create an Application that will display shows running in theaters. We have to create two ends - Admin and User. Admin should be able to use CRUD on shows and Venues. Users should be able to book any show from the venue of their choice. We are to add additional functionalities like automated EMail generation, caching and Venue summary.

Technologies Used

- **Flask** (Web development Framework)
- **SQLAlchemy** (To work with Database)
- **Vuejs**(Frontend)
- **Bootstrap** (Frontend - Better UI)
- **Redis** (Server - caching and celery hosting)
- **SQLite3** (Database)
- **Celery** (Email generation)
- **Jinja2** (For templating)

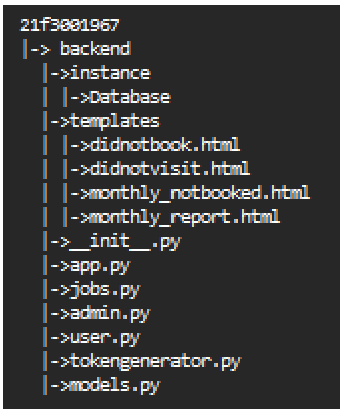
DB Schema Design

Table Name	Purpose	List of attributes and constraints
Admin	It has admin Details **Features** Username must be unique. Hence using this as a primary key.	 "Username" TEXT NOT NULL UNIQUE, "Password" TEXT NOT NULL, "Name" TEXT NOT NULL, PRIMARY KEY("Username")
USER	It has user Details **Features** Username must be unique. Hence using this as a primary key.	 "username" TEXT NOT NULL UNIQUE, "password" TEXT NOT NULL, "email" TEXT NOT NULL UNIQUE, "name" TEXT NOT NULL, PRIMARY KEY("username")
Userlogs	It maintains logs of User	 "id" INTEGER NOT NULL UNIQUE, "username" TEXT NOT NULL, "timestamp" TEXT NOT NULL, PRIMARY KEY("id" AUTOINCREMENT), FOREIGN KEY("username") REFERENCES "USER"("username")
Venue	It has the list of Venues **Features** Venue names can be the same hence creating an ID for each venue entered. It is auto incremented. ID is the primary key. Attribute “imgURL” is of text format.	 "ID" INTEGER NOT NULL UNIQUE, "Name" TEXT NOT NULL, "Place" TEXT NOT NULL, "Capacity" INTEGER NOT NULL, "imgURL" TEXT, PRIMARY KEY("ID" AUTOINCREMENT)

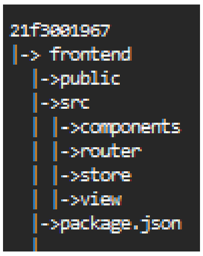
Shows	It has the list of Shows
	Features
	Show names can be the same hence creating an ID for each showentered. It is auto incremented.
	ID is the primary key. Attribute "imgURL" is of text format. I have used images only from the internet. However, that is not a necessity. Venue_id is a foreign key. It refers to the venue from the Venue database.
		`"ID"    INTEGER NOT NULL,`
		`"Name"  TEXT NOT NULL,`
		`"imgURL"    TEXT,`
		`"Description"   TEXT,`
		`"Ratings"   INTEGER,`
		`"pos"   FLOAT NOT NULL,`
		`"venue_id"  INTEGER,`
		`"Tags"  TEXT ARRAY NOT NULL,`
		`"Dates" DATETIME ARRAY NOT NULL,`
		`"As"    INTEGER,`
		`"Allseats"  INTEGER,`
		`FOREIGN KEY("venue_id") REFERENCES "Venue"("ID"),`
		`PRIMARY KEY("ID" AUTOINCREMENT)`

Bookings	It has the list of bookings made by users
	Features
	ID keeps track of total number of bookings (past and present). The column "date" is a text as it is easier to print. Python can convert into Datetime whenever necessary.
		`"ID"    INTEGER NOT NULL,`
		`"Username"  TEXT NOT NULL,`
		`"show_id"   INTEGER NOT NULL,`
		`"nos"   INTEGER NOT NULL,`
		`"price" NUMERIC NOT NULL,`
		`"date"  TEXT NOT NULL,`
		`"Rating"    FLOAT,`
		`"timestamp" TEXT,`
		`PRIMARY KEY("ID" AUTOINCREMENT)`

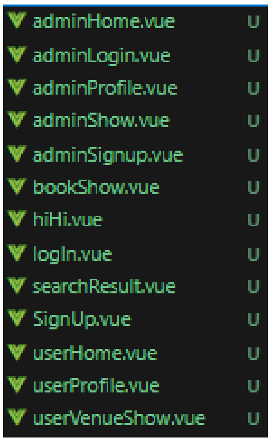
Architecture and Features



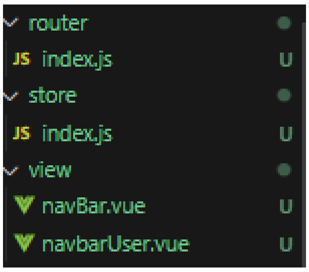
```
21f3001967
|-> backend
  |->instance
  |  |->Database
  |->templates
  |  |->didnotbook.html
  |  |->didnotvisit.html
  |  |->monthly_notbooked.html
  |  |->monthly_report.html
  |->__init__.py
  |->app.py
  |->jobs.py
  |->admin.py
  |->user.py
  |->tokengenerator.py
  |->models.py
```



```
21f3001967
|-> frontend
  |->public
  |->src
  |  |->components
  |  |->router
  |  |->store
  |  |->view
  |->package.json
  |
```



```
adminHome.vue       U
adminLogin.vue      U
adminProfile.vue    U
adminShow.vue       U
adminSignup.vue     U
bookShow.vue        U
hiHi.vue            U
logIn.vue           U
searchResult.vue    U
SignUp.vue          U
userHome.vue        U
userProfile.vue     U
userVenueShow.vue   U
```



```
router
  index.js        U
store
  index.js        U
view
  navBar.vue      U
  navbarUser.vue  U
```

The project features a typical web application architecture with frontend and backend components. The frontend manages user interface elements, while the backend handles server-side logic and data storage. Notable backend elements include database management, HTML templates, and Python modules for user, admin, jobs and token-related functionalities. The frontend of this project leverages Vue.js to create an interactive and modular user interface. It employs Vue Router for navigation and Vuex for state management, enhancing the user experience in the web-based ticket booking system.

Features

- **Login:** Token based authentication and role based access
- **UI**: The project is mainly inspired from the Wireframe.
- **Admin side**:
    1. CRUD on venues and Shows
    2. Download summary
- **User side:**
    1. Booking Tickets
    2. Searching
    3. Viewing Tickets (cache enabled)
- **Jobs:** Email generation

Video Link

https://drive.google.com/file/d/1DQ-qQ3G_8fOkmpRwZ1PgpvuH-o1qn-MQ/view?usp=sharing