ABOUT TICKETSHOW

- Signup in the app. (Core)
- Login with JWT Token based authentication
- Support for role-based login (admin/user)
- Manages multiple theatres and multiple shows per theatre.
- Search theatres and shows based on multiple parameters.
- Allows booking multiple tickets for shows, capped by the theatre capacity.
- Daily reminders emailed to all users who've not logged in or booked tickets in the app.
- Monthly admin pdf report containing details of bookings and ratings done by the users of the app during that month, emailed to the admin.
- Ability to export details of selected theatre, including current shows, bookings and user ratings.
- Caching-enabled results for improved performance.

APPLICATION DESIGN

TicketShow GUI is built using Vue.js, uses Flask APIs and has an SQLite backend powered by Flask-SQLAlchemy.

In addition to the above, following technologies also have been employed in this project.

- Vue CLI, Bootstrap, Jinja (UI)
- Flask-JWT-extended (Authentication/login)
- Celery, Redis (Async tasks)
- Pdfkit (Reports)
- mailHog (Email client)
- Flask-restful (APIs)
- Flask-caching (Performance)

Submitted by Anand K. Iyer (Email: 21f1001185@ds.study.iitm.ac.in)

File organization of the project is as follows.

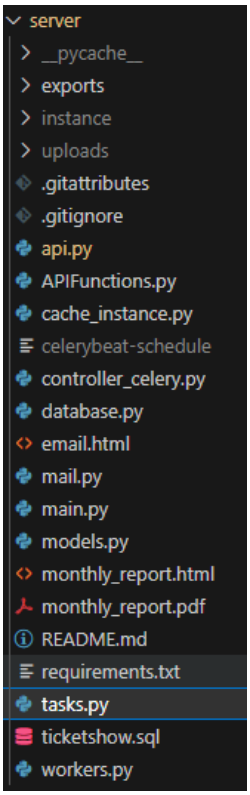
SERVER (ONLY THE MAJOR FILES HAVE BEEN LISTED HERE):

- main.py is the entry point, contains majority of the configuration for the project, and defines the mapping of APIs with the views.
- api.py contains majority of the API definitions, with references to APIFunctions.py.
- cache_instance.py defines caching strategy.
- controller_celery.py defines the internals of *Export Theatre* functionality.
- mail.py defines emailing functionality.
- models.py defines the SQLAlchemy classes.

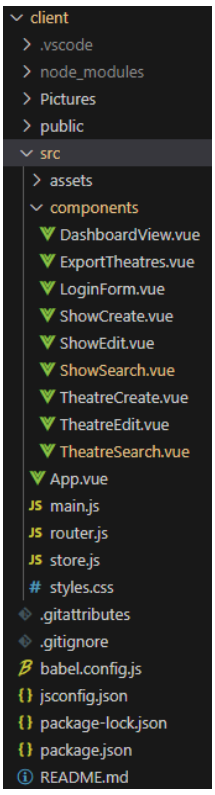
CLIENT (ONLY THE MAJOR FILES HAVE BEEN LISTED HERE):

- main.js is the entry point, initializes the app, router and store.
- App.vue represents the main app, its navigation and logout.
- src/components contains Single File Components (SFC), used by Vue-based GUI.
- router.js has definitions for all UI routes in the application.
- store.js has a barebone definition of Vue store.

Server



Client



DATABASE SCHEMA

The database consists of 7 tables, with the columns as follows:

Table	Columns	Details	Constraints
theatre	t_id	Theatre ID	Primary key
	name	Theatre name	
	place	Theatre location	
	capacity	Theatre capacity	
	image_url	Theatre image	
show	s_id	Show ID	Primary key
	t_id	Theatre ID	Refers *theatre* table
	t_name	Theatre name(location)	
	current	is the show current (True/False)?	
	name	Show name	
	rating	Average show rating (1 - 5)	
	tag1	Show tag1	
	tag2	Show tag2	
	tag3	Show tag3	
	price	Show ticket price	
	start	Show start time (minutes since midnight)	
	duration	Show duration (in minutes)	
	availabiity	Unused	
	image_url	Show image	
showdate	id	ID	Primary key
	s_id	Show ID	Refers *show* table
	t_id	Theatre ID	Refers *theatre* table
	s_date	Show date	
	availabiity	#seats available on s_date	
user	u_id	User ID	Primary key
	name	User name	
	email	User email address	
	password	User password	
	role	admin(True) or non-admin(False)	
booking	b_id	Booking ID	Primary key
	u_id	User ID	Refers *user* table
	s_id	Show ID	Refers *show* table
	t_id	Theatre ID	Refers *theatre* table

	date	Show date for which ticket is booked	
	month	Show month for which ticket is booked	
	seats	#seats booked	
	cost	Total cost of booking	
rating	r_id	ID	Primary key
	u_id	User ID	Refers *user* table
	s_id	Show ID	Refers *show* table
	t_id	Theatre ID	Refers *theatre* table
	date	Date on which user rated the show	
	month	Month on which user rated the show	
	rating	User rating for the show (1 - 5)	
audit	id	ID	Primary key
	u_id	User ID	Refers *user* table
	op	Operation (booking/login)	
	op_date	Date of *op*	

USER GUIDE FOR THE APPLICATION

The app supports two types of users = admin and non-admin. admin is present by default when the app is initialized. No signup is required for 'admin'. All others users are required to sign-up to the app.

During sign-up, user needs to provide a name, password and email address. User is allowed to login/sign-in using the user name and password. User's email address is utilized only when emails need to be sent to the user.

Once logged in, the user will be presented with a dashboard where all current shows are listed. The user will be allowed to book for any of the listed shows, for the current or next day, while remaining in the same screen. Multiple tickets can be booked for the same show, when the (single) ticket price will be multiplied by the number of seats required.

When no user has booked the show yet, the number of seats available equals the theatre capacity. With each booking, the number of seats gets reduced; when it reduces to 0, no more bookings are allowed.

Apart from booking tickets for a show, user will be able to rate the show on a scale of 1-5 from the dashboard.

Everyday at 5 pm, the app automatically sends email messages to all users who hasn't logged in or booked a ticket, prompting them to login to the app.

During the 1st of every month, every user registered with the app will be automatically sent out an email message with a pdf attachment containing details of shows booked and ratings he/she gave to various shows during the past month.

Unlike a normal user, the *admin* is able to do the following operations:

- Add, Update, Delete and Search theatres
- Add, Update, Delete and Search shows
- Export details about all shows/bookings/user ratings for a selected theatre. Note that this is an asynchronous operation, which implies that multiple such *export* operations could run concurrently.

More details about the project in this video.

Submitted by Anand K. Iyer (Email: 21f1001185@ds.study.iitm.ac.in)