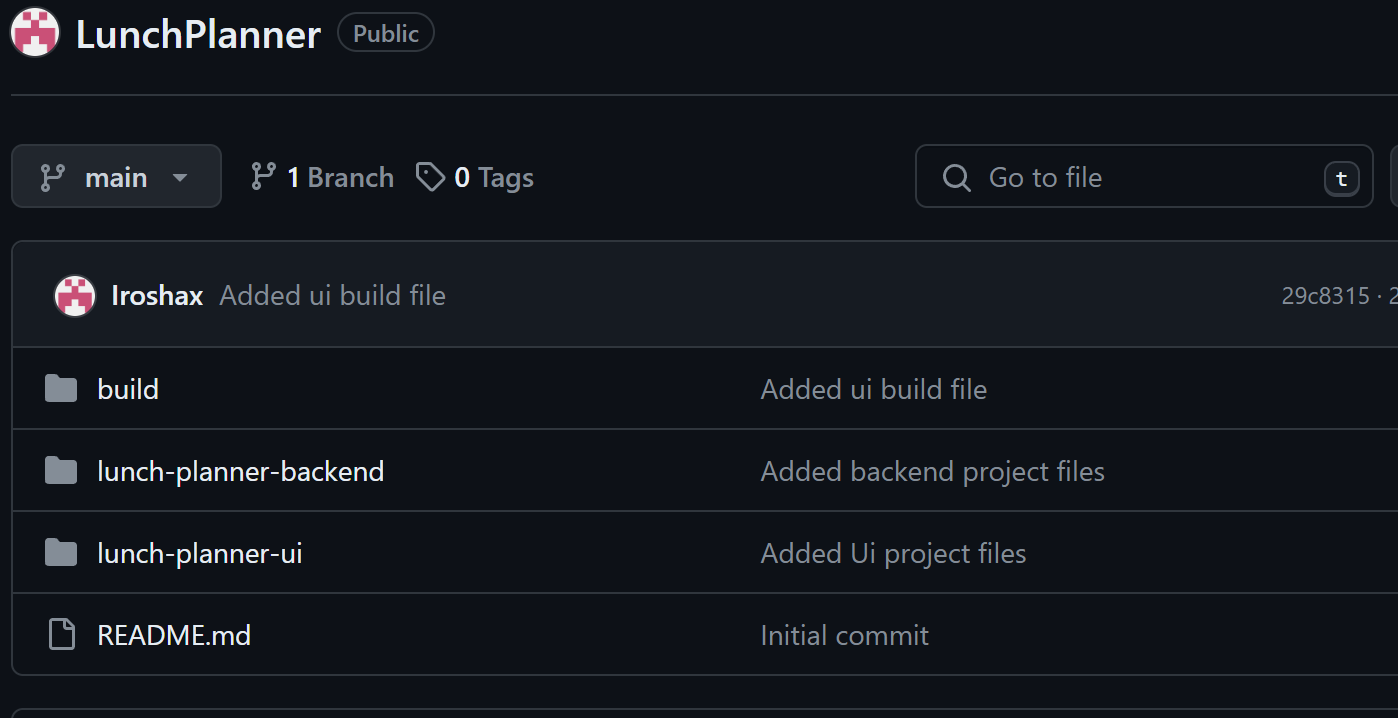
**How to run the Lunch Planner application?**

**Prerequisites:**

* Node.js version 16.20 or higher
* JDK/JRE 17 or Higher
* MYSQL server

**Git repository structure**

There are two folders for the UI and backend. Please clone the folders to your local pc to run the application. Alternatively, I have included the UI build file inside the build folder.



**To run react UI –**

In the “lunch-planner-ui” project directory, you can run:

* *npm install*

This will create the folder node\_modules inside the project folder including all the npm modules required to run the project. This needs to be executed once.

* *npm start*

Runs the app in the development mode.

Open http://localhost:3000 to view it in your browser.

**To run spring boot backend –**

Build the project with Maven:

If you’re using Maven, you can use the mvn clean install command to build your project and create a .jar file.

* *mvn clean install*

If the build is successful, you’ll see a BUILD SUCCESS message at the end.

Run the application:

After the build is successful, a .jar file will be created in the target directory. You can run this .jar file using the java -jar command. If your .jar file is named lunch-planner.jar, you would type:

* *java -jar target/ lunch-planner.jar*

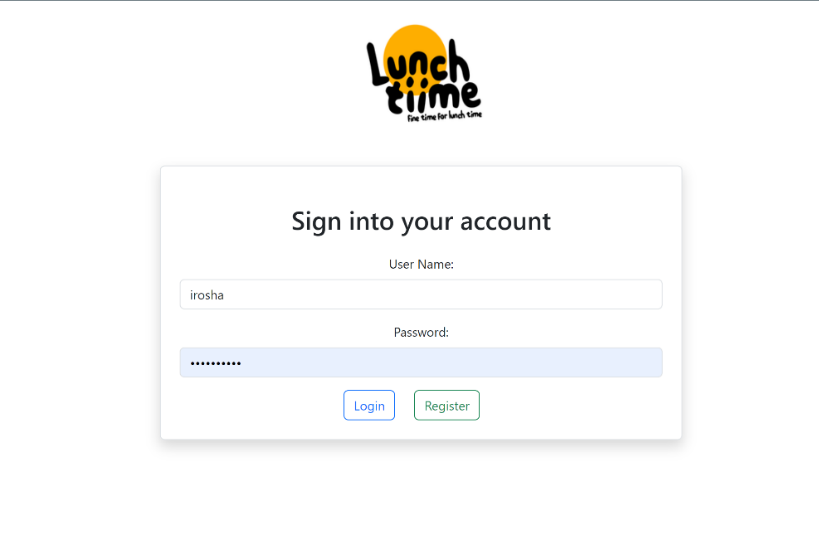
lunch-planner application should now be running, and you can access it at <http://localhost:8087>

**User Guide**

To submit a restaurant for lunch, a team member has to follow below steps.

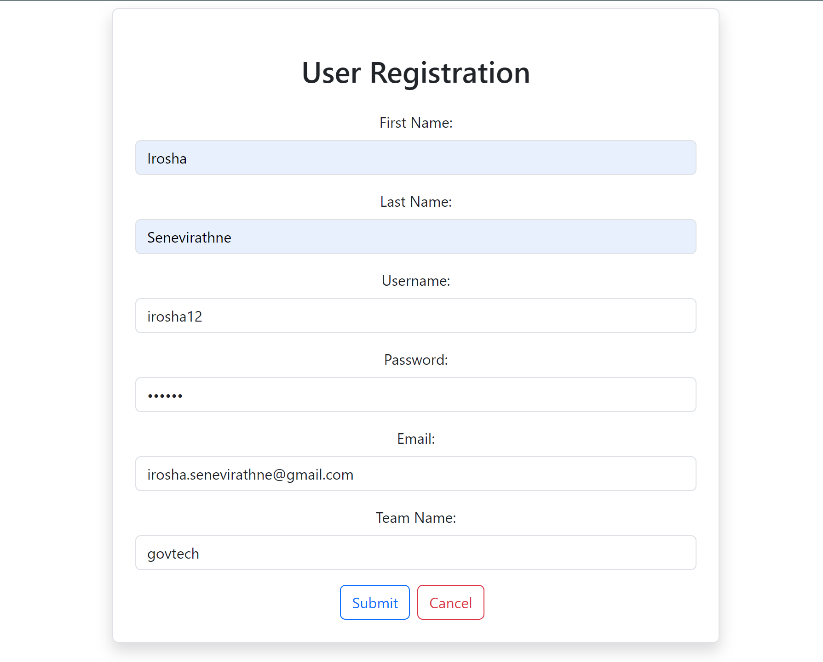
**Step 1:**

Login to the lunch planner application by typing username and password.



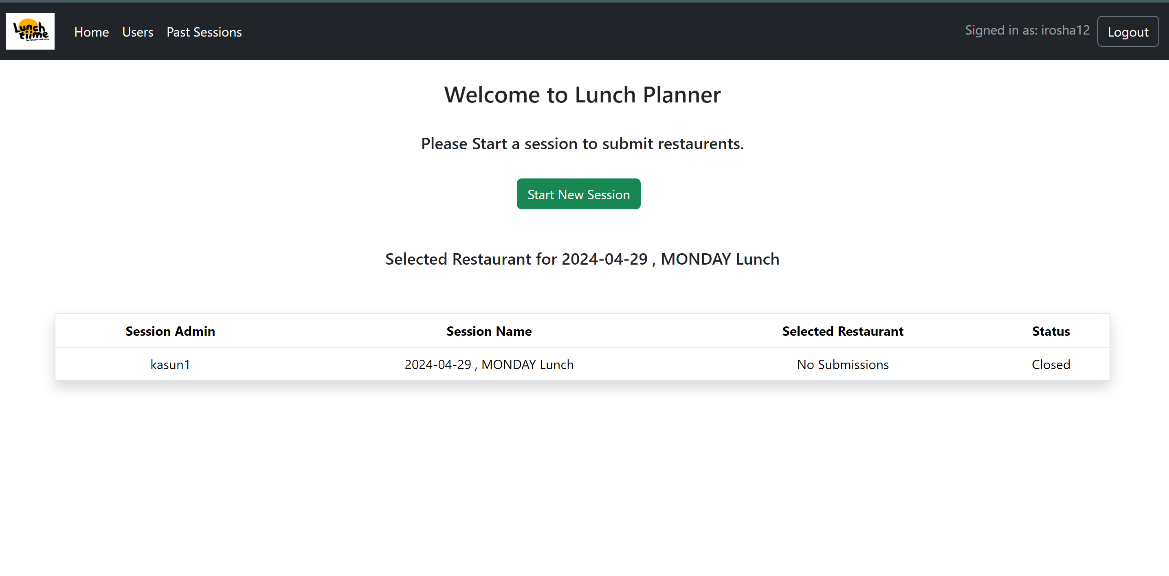
If the user is new, then click the register button to create an account. User has to fill below form and click submit button.

Once the registration is successful, user can login to the application.

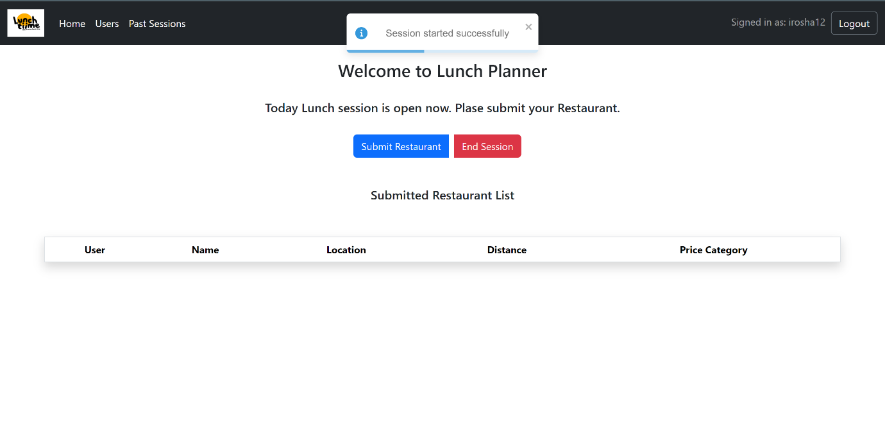


**Step 2:**

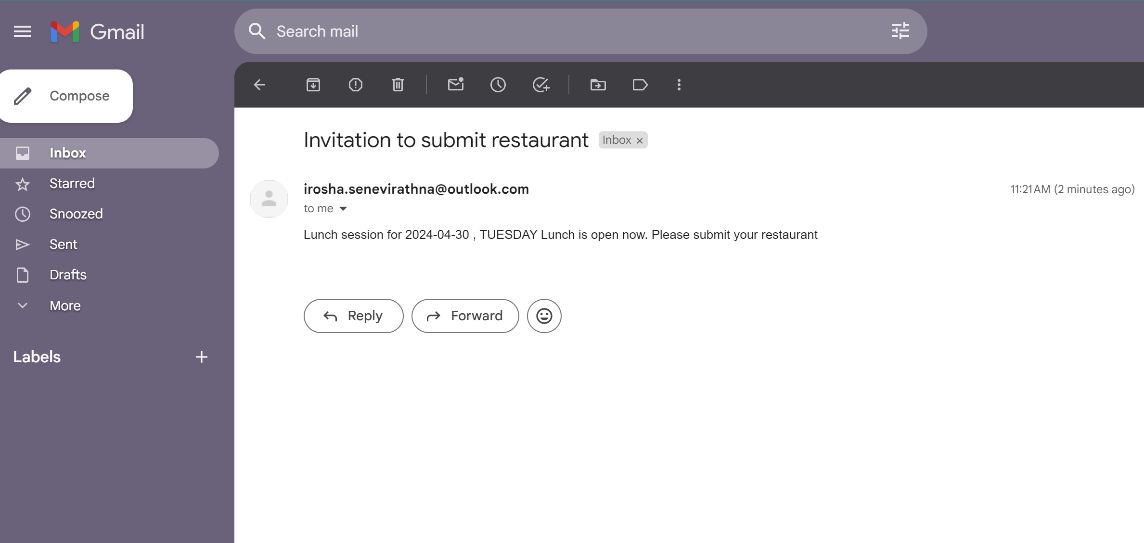
Successful login redirects user to the home page. User can start a new session by clicking the “Start New Session” button. Previous day lunch session summary also visible in the home page.



Once the session started successfully user will see the success message in the screen.



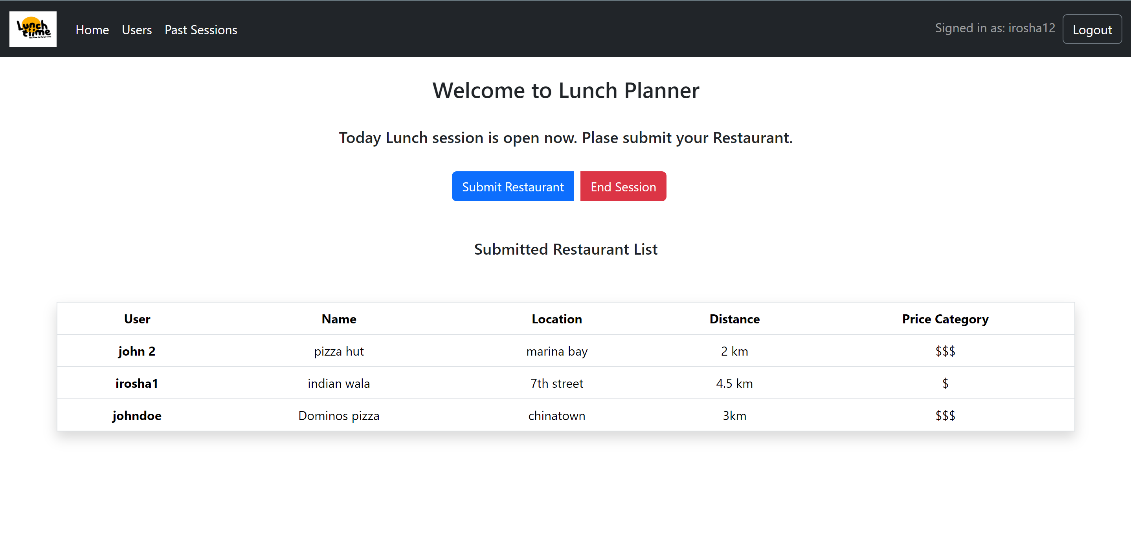
All the users registered in the system will get an email invitation to submit their restaurants.



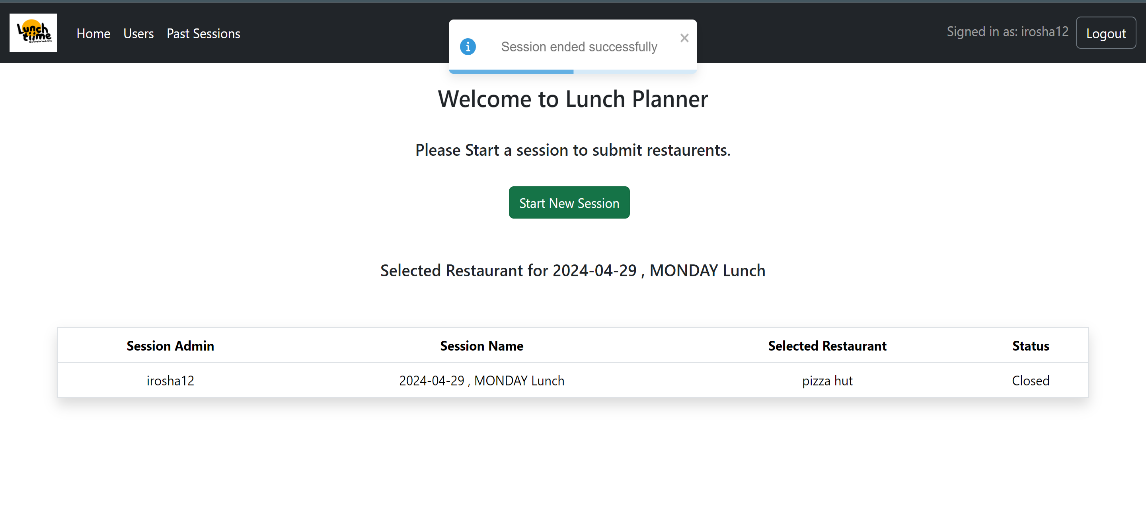
Step 3:

The user who started the session is the session admin and only session admin can end the session.

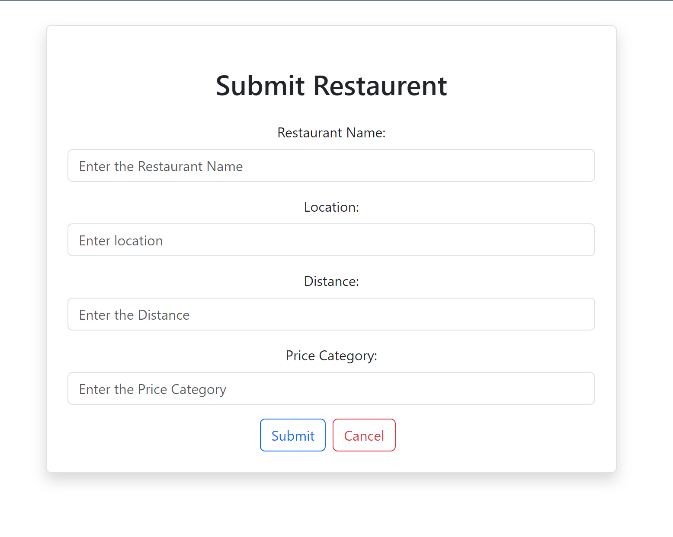
To end the session user can click “End Session” button.



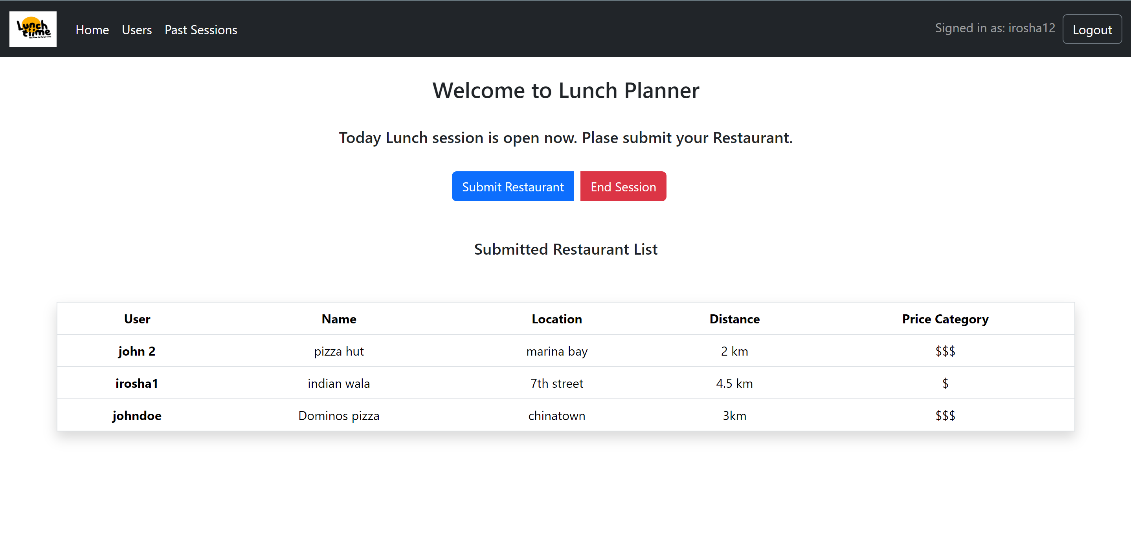
Once the session is successfully ended, user will see the success message on the screen. A random restaurant will be selected from the submission list and details will be shown in the home screen.



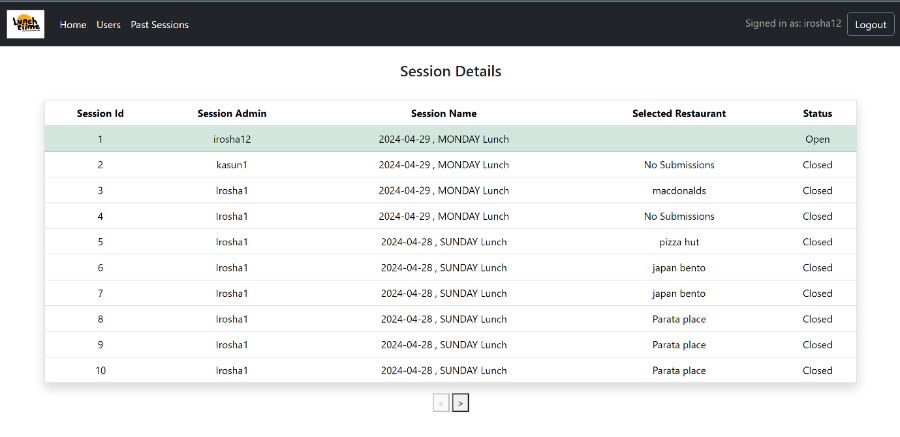
To submit a restaurant user can click “Submit Restaurant” button and it will open a form to fill restaurant details as below.



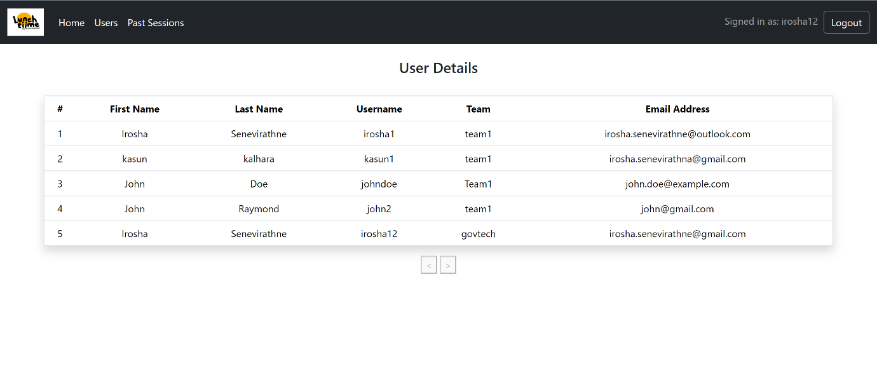
Once the restaurant is submitted user will be redirected to the home page where all the submissions are listed for the active session.



To view past sessions user can click “Past Sessions” from the nav bar. It will list down all the past sessions and highlight the active session in green.



To view all the users, click “Users” from nav bar. It will list down all the registered users in the system.



To logout click the “Logout” button from the top right corner of the navbar.

**Backend API endpoint list**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Method** | **Endpoint** | **Description** | **Request Body** | **Response** |
| POST | /api/lunchPlanner/session/initiate | Initiates a session. | UserDto object | Map<String, RestaurantDto>: A map representing the session cache. |
| POST | /api/lunchPlanner/session/end | Ends a session. | UserDto object | RestaurantDto: The selected restaurant for the session. |
| GET | /api/lunchPlanner/session | Gets the list of sessions. | None | List<SessionDto>: A list of session DTOs. |
| GET | /api/lunchPlanner/session/latest | Gets the latest session. | None | SessionDto: The latest session DTO. |
| GET | /api/lunchPlanner/session/status | Gets the status of the session. | None | String: The status of the session. |
| GET | /api/lunchPlanner/session/active | Gets the active session. | None | Map<String, RestaurantDto> A map representing the active session cache. |
| POST | /api/lunchPlanner/user/register | Register new user | UserDto | None |
| POST | /api/lunchPlanner/user/login | User login | LoginDto | UserDto: UserDto containing user information. |
| GET | /api/lunchPlanner/user | Get the list of users | None | List<UserDto>: A list of user DTOs. |
| POST | /api/lunchPlanner/restaurant/submit | Submit restaurant to the session | RestaurantSubmitDto object | Map<String, RestaurantDto>: A map representing the session cache. |

Design Decisions

* More than one session can be initiated for a given day.
* Session admin can also submit a restaurant for an active session.
* A time limit has not been implemented for the active session at the moment.

Special notes:

I have removed mail server account info from application.properties since I have used my personal email to configure and test email notifications. Please use a different email credential in your application.properties to run the application.