Choicemaker Guide

Implemented and Documented By	Sachith Senarathne
Version	v1.0.0
Date	25.10.2023

Table of Content

[Choicemaker Guide](#)
- [Introduction](#)
- [Design](#)
  - [Components](#)
- [Dev Environment and Local Setup](#)
  - [Backend API](#)
  - [Froned App](#)
- [API Documentation](#)
- [AOP Logging for debugging](#)
- [App Demonstration with User Journeys](#)
  - [Create Session Flow](#)
  - [Join Session Flow](#)

Introduction

Choicemaker is here to simplify the process and make dining decisions an effortless and enjoyable experience. Choicemaker is an application that combines React-based frontend and a Spring Boot backend, equipped with both RESTful APIs and WebSocket communication. With Choicemaker, it will facilitate a more efficient and collaborative approach to choosing the perfect restaurant for your day.

Key Features:

1. **Session Creation**: Any user can initiate a dining session, taking the lead in selecting where to eat. The creator has the privilege to add other users to the session, establishing a central hub for the decision-making process.

2. **User Collaboration**: Choicemaker has the ability to bring people together. Users can join an existing session (undergoing the fact that the user should be invited by the session creator), contributing their opinions and preferences to the decision-making process.
3. **Real-time Updates**: Choicemaker employs WebSockets to keep all users in the session instantly informed. When a participant submits their restaurant choice, the rest of the group receives immediate notifications, ensuring everyone is on the same page throughout the process.
4. **Session Conclusion**: The session creator holds the key to finalizing the choice. When they decide the time is right, the creator can end the session, and this action will trigger the final selection process.
5. **Randomized Selection**: To decision-making, Choicemaker randomly selects the restaurant from the submitted options. Selected restaurant will be notified by all the users in the session.

Design

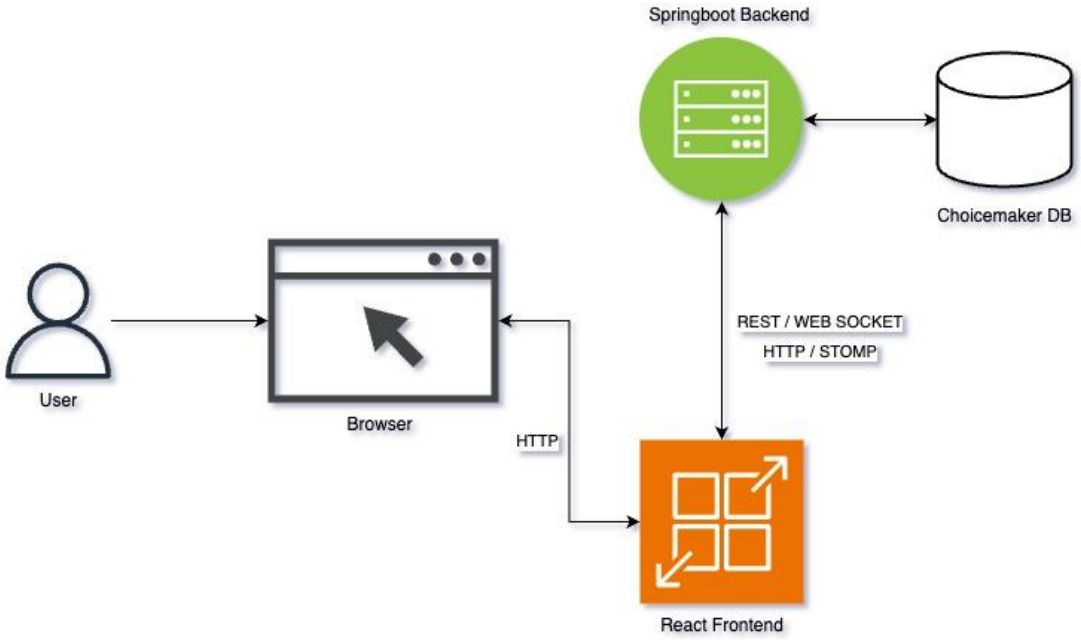
Components

Component	Usage	Technology
Backend	Backend API and Websocket server	Springboot, AOP, OAPI
Frontend	Froned app for UI components	React, SockJs
Database	Storing data related to sessions	MySql

Designing Decisions

- **Real-time Updates** - To get real time updates, there can be two or more solutions. Using web-socket and using an event broker based event driven system. Selected web socket considering the small scope and scale of the application.
- **Combining REST and WebSocket Communication** - RESTful APIs are excellent for standard HTTP-based interactions, while WebSockets provide real-time communication. This combination ensures that this application can handle both standard data exchange and instant updates for a seamless user experience.
- **React for Frontend Development** - React is a popular and robust JavaScript library for building user interfaces. It provides a component-based architecture, making it easier to manage and update the application's UI elements efficiently.
- **Material-UI for UI Components -** Material-UI is renowned for its reliability and ease of use, offering a comprehensive set of pre-designed UI components. This decision ensures a consistent and visually appealing design across your application.

- **Improved Tracing and Debugging with AOP** - AOP enhances the traceability and debugging capabilities of "Choicemaker" by allowing for consistent and centralized logging across the codebase, making it easier to identify and resolve issues during development and runtime.
- **Database with SQL -** This decision was made to ensure structured and organized data management. By using a relational database and SQL, "Choicemaker" can efficiently store and manage data, including user profiles, restaurant choices, session information, and relationships between these data points.



Dev Environment and Local Setup

PreRequisite

- Java 17
- Maven 3
- MySql
- Node and npm
- Git

Backend API

Note: Please configure the db connection properly by adding below properties as per the running system db before starting the application

```
spring.datasource.url=jdbc:mysql://localhost:3306/choicemaker
spring.datasource.username=
spring.datasource.password=
```

Step 01:

Clone the project

```
git clone https://github.com/sachithsen/choicemaker
```

Step 02:

Go to the directory

```
cd choicemaker
```

Step 03:

Install maven dependencies

```
mvn clean install
```

Step 04:

Run the project

```
mvn spring-boot:run
```

You can verify the application status using actuator health endpoints.

http://localhost:8080/actuator/health which will give {"status":"UP"}
as a result.

Froned App

Frontend app is inside the same repo and version controlled as a mono repo

Step 01:

Go to the directory

```
cd choicemaker/frontend
```

Step 02:

Install node modules

```
npm install
```

Step 03:

Run the project

```
npm start
```

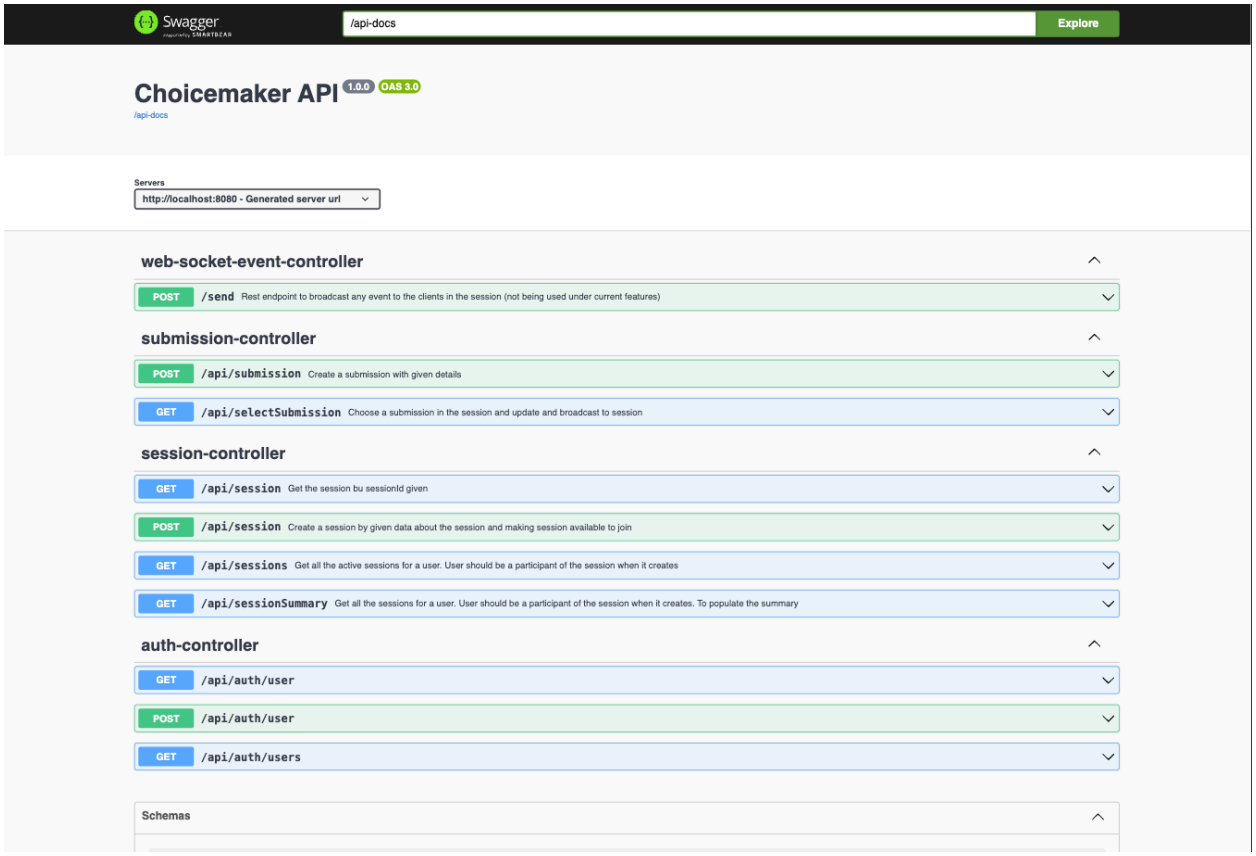
You will be able to access the frontend application using the URL below.

http://localhost:3000

API Documentation

You will be able access Open API documentation once the backend application is up and running and try the endpoints as well.

http://localhost:8080/swagger-ui/index.html

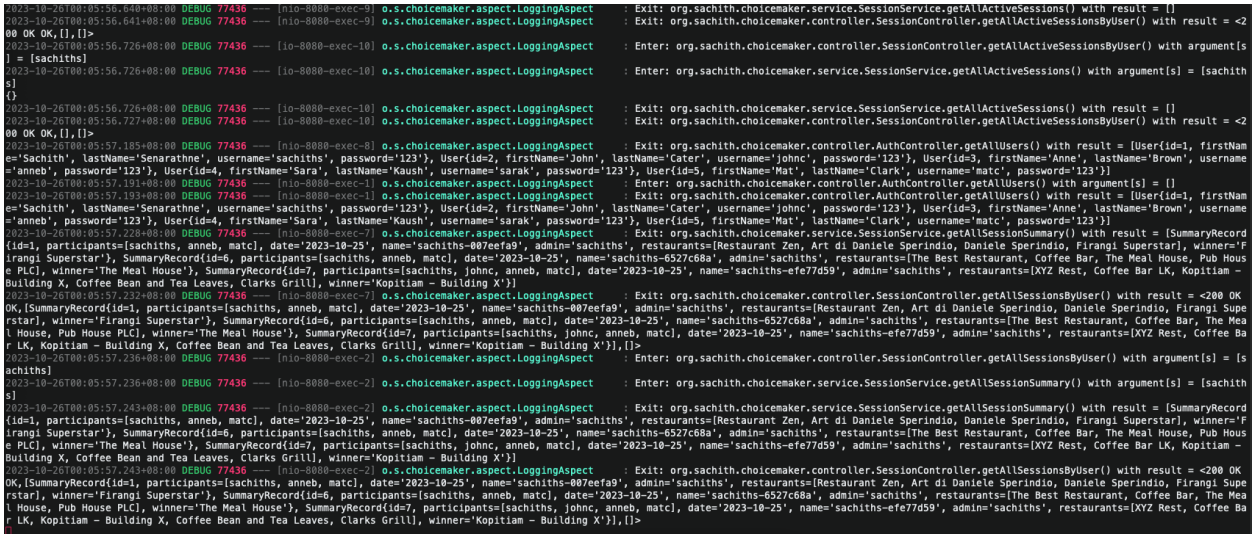


AOP Logging for debugging

Log levels have been set to debug the application and logging has been introduced using Spring AOP to improve the traceability.

```
logging.level.org.springframework.web=INFO
logging.level.org.springframework.security=INFO
logging.level.org.hibernate=ERROR
logging.level.org.sachith.choicemaker=DEBUG
```

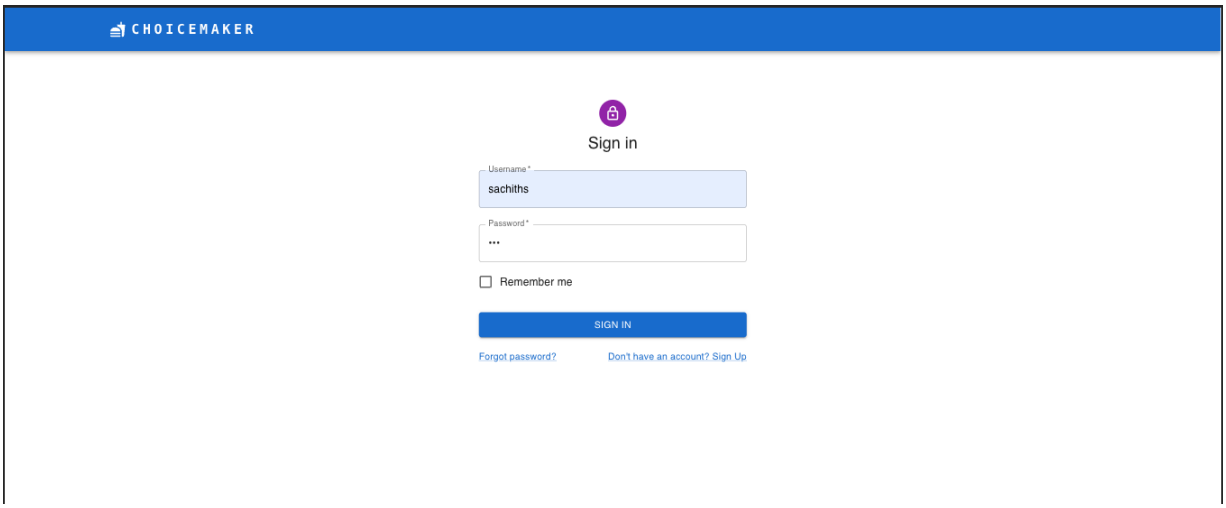
Sample logs



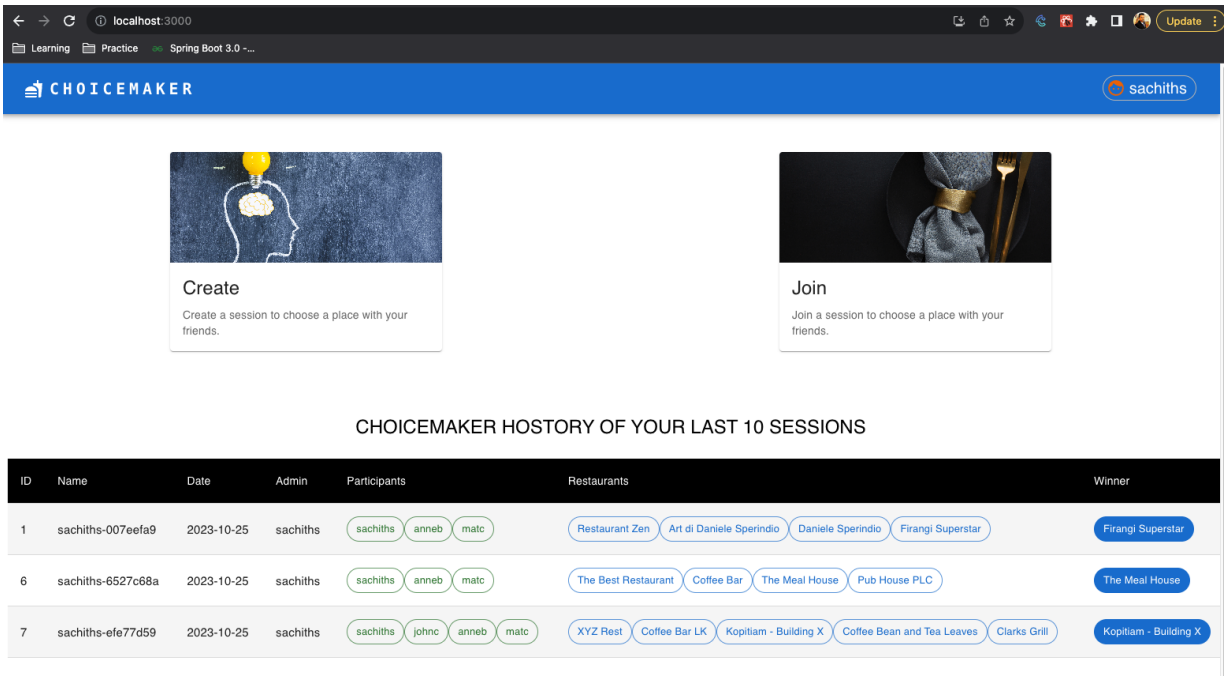
App Demonstration with User Journeys

Create Session Flow

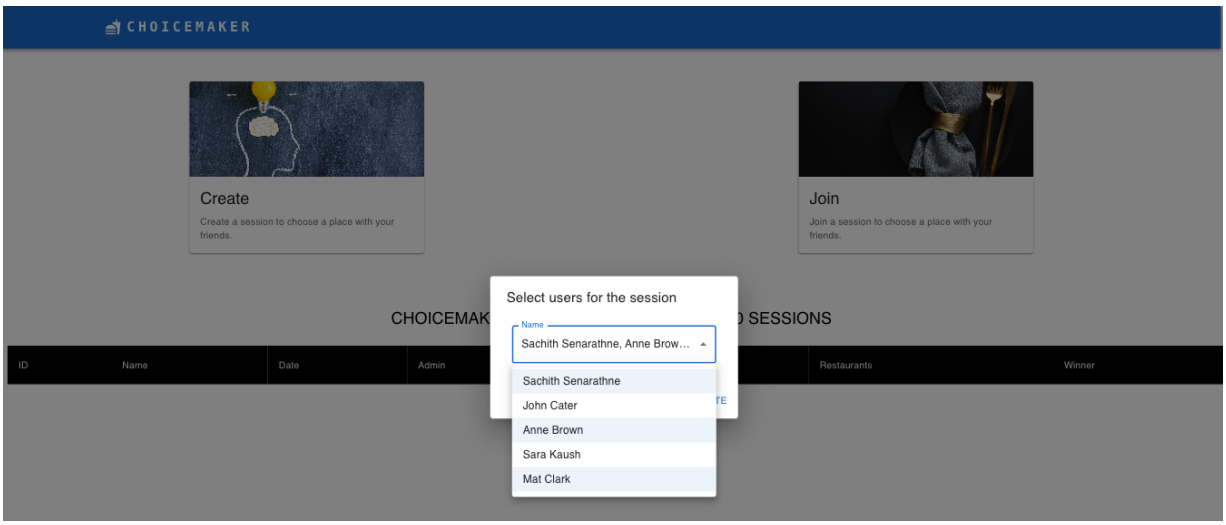
01: Login to the application using username and password



02: User will land on a Home page. Home page has links to Create or Join Session and last 10 session details of the user logged in.

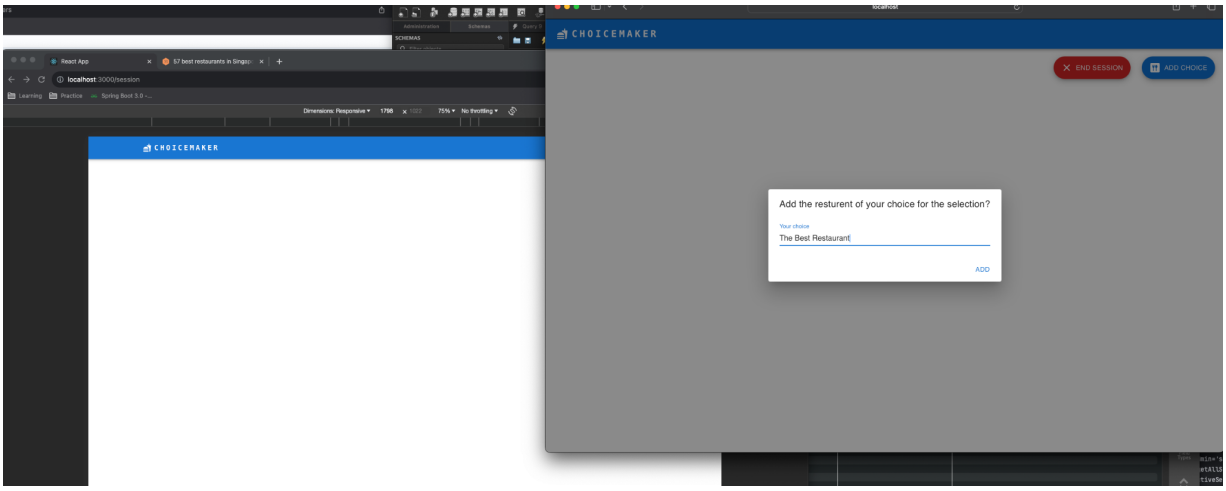


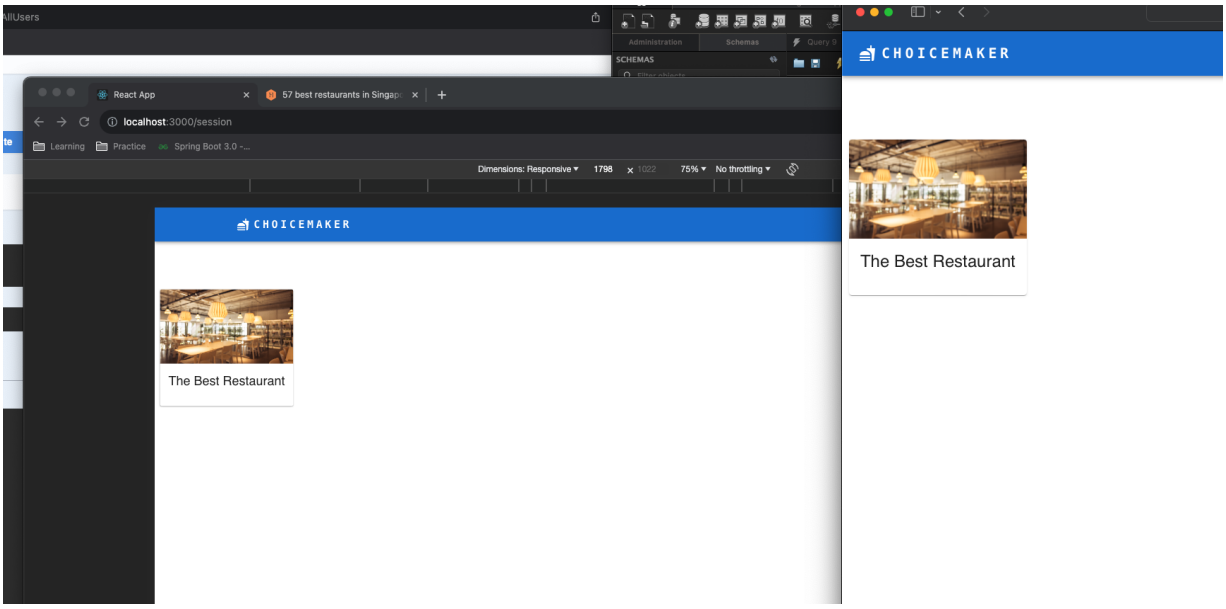
03: User can create a session by adding users who registered to the application. (Pre created users via API as user registration is not provided as a feature)



04: User will land in the Session Room and will be able to add the choices and end the session via UI controls.

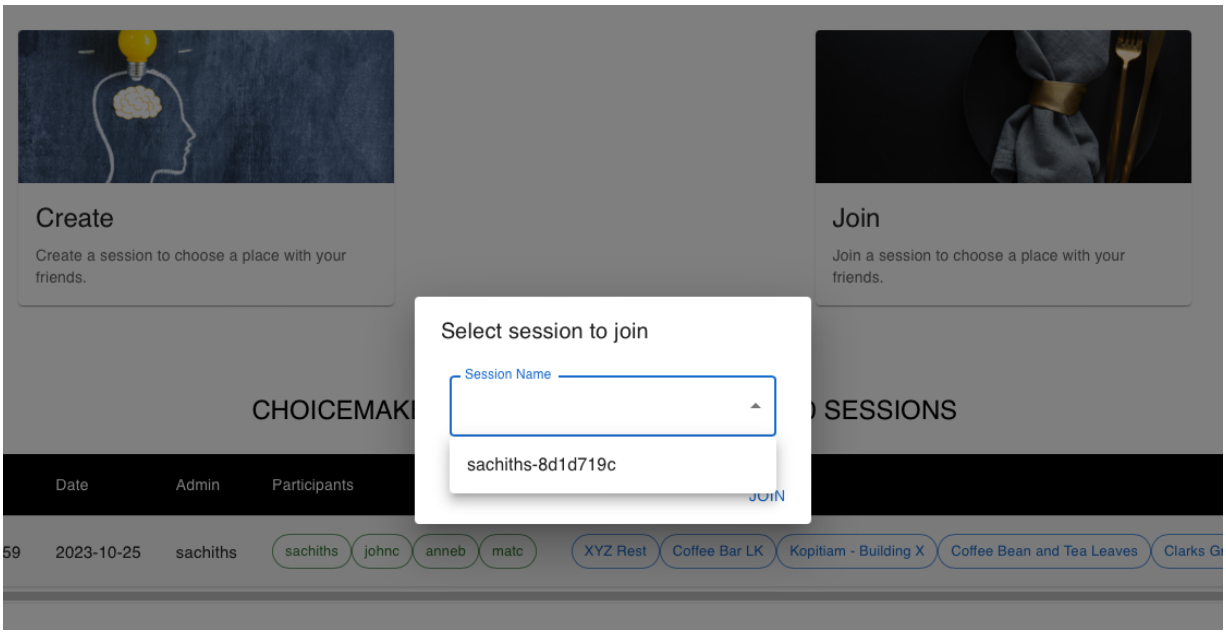
05: Add restaurant choices and it will be visible to other users instantly.



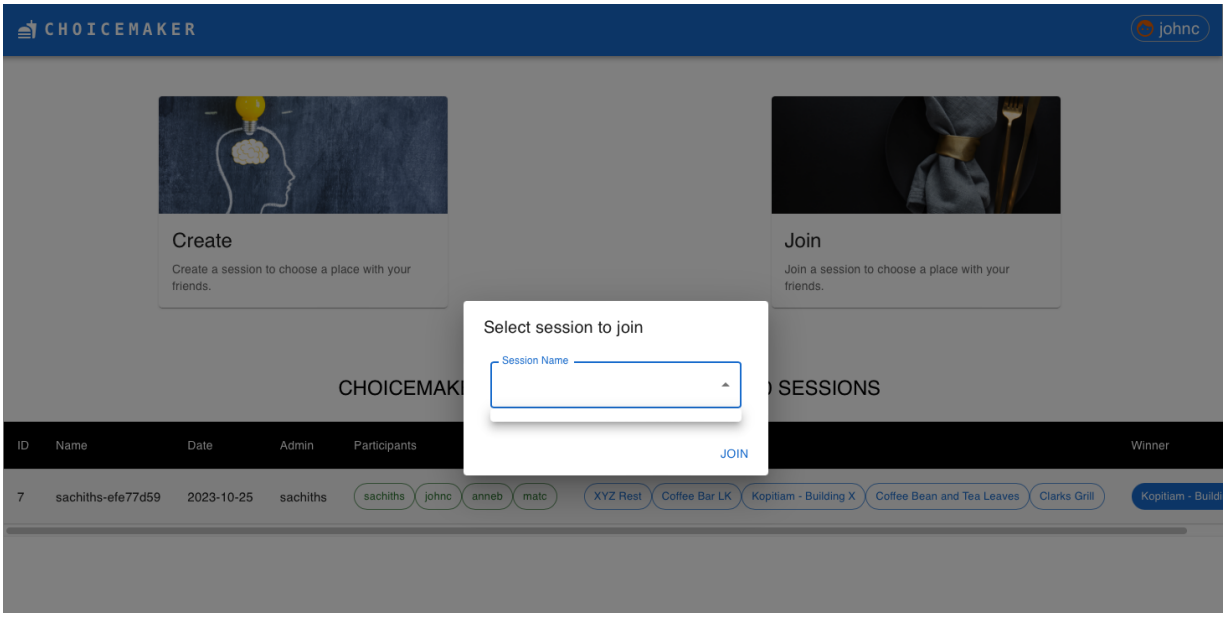


Join Session Flow

01: User login and join a session. List of sessions will be available to select and if the user has been invited, the session name will show in the dropdown. If you are not invited, you will not see the session in the dropdown.

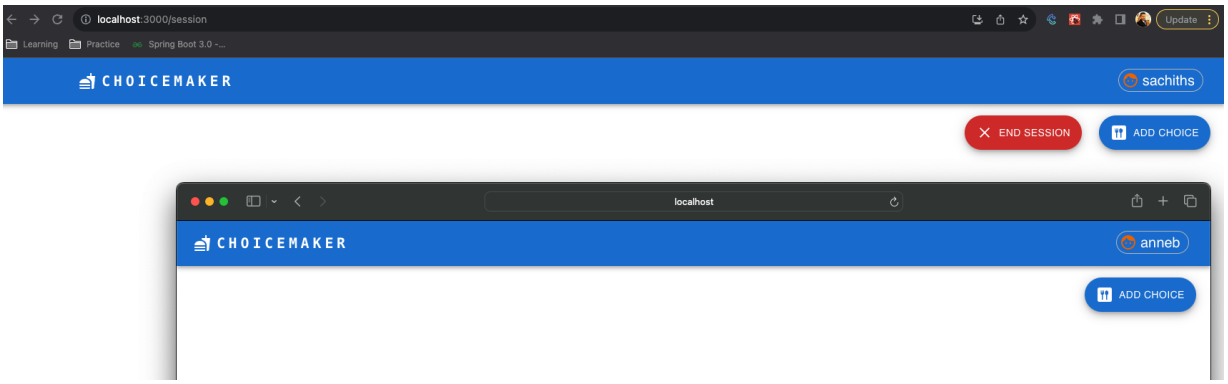


See Step 03 above, John C is not invited 😕



02: User Joins a session selected and will be able to submit choices as required.

03: As a user joins as a guest / non admin, the user will not be displayed a “END SESSION” button.



04: When the user who created the session ends the session. All the users will be instantly notified of the selected Resturent for the session. 😀
Then will land on the home page and the session ended will be added to the summary. 👋