

# RSATE – restaurant management platform

## *Introduction*



*Course: Integrative Software Engineering*

*Group: 2024 semester B, Wednesday 8:00*

*Name: 2024b.aviad. korakin*

*Version: Fourth Sprint*

*Last update:16/07/2024*

## Table of contents

RSATE – restaurant management platform <i>Introduction</i> .....	1
Software Requirements .....	3
Introduction .....	3
General Description .....	3
The Vision & Objectives .....	3
Purpose of system.....	3
Scope of system .....	4
Beyond the scope.....	4
Actors and Goals .....	5
Functional Requirements.....	6
Use case diagram .....	6
Use case details.....	7
Make a reservation .....	7
View reservations.....	7
Delete reservation .....	8
Take-away ordering .....	8
Dine in ordering .....	9
Browse and prepare to order .....	9
Add/View tables.....	9
Add menu item.....	10
Modify menu item .....	10
Non-Functional Requirements.....	11
System bonuses - Appendix no. 1 .....	12
Technologies Document – Appendix no. 2 .....	13
Project report – Appendix no. 3.....	14
Team members .....	14
General Summary of work on sprint 4.....	15
General Summary of work on the project .....	15
Kanban Boards .....	16
Sprint 1 .....	16
Sprint 2 .....	17
Sprint 3 .....	18
Sprint 4 .....	19

# Software Requirements

## Introduction

### General Description

RSATE is an integrated software solution designed to streamline and enhance the operational efficiency of restaurants. The server-side component of the restaurant management system serves as the backbone, handling the core processing, data storage, and business logic that powers the entire application. This component is designed to manage high volumes of transactions, maintain data integrity, and ensure seamless communication between various modules.

It also offers an intuitive interface that is friendly both for the workers of the restaurant - implementing administrative actions, and for the customers that come to hang out and get the best experience .

### The Vision & Objectives

RSATE's vision is to produce a convenient and user-friendly platform for both restaurant employees and customers. The system is designed to cater to various types of dining establishments, from small cafes to large restaurants, providing a scalable and customizable solution to meet diverse business needs. The goal is to make the management of the restaurant as easy as possible so that the entertainment experience of the customers will be ultimate, while allowing each restaurant to adapt the system to the convenience of the employees.

## Purpose of system

RSATE offers a convenient and reliable platform through which managing a complex business such as a restaurant with a large number of departments will become intuitive and user-friendly, and will reduce the complexity and redundancy of using several different systems.

The system focuses on the needs of the users. both of the employees in the restaurant - the waiters, the managers, and the customers. Customers can make reservations and view the detailed menu easily, while managers can make real-time changes and manage the shift efficiently. The platform allows changes to be made during the shift and is designed to allow all restaurant departments to work in a synchronized and efficient manner. In order to meet these needs, the system provides a powerful and expandable server.

## Scope of system

The system offers a variety of options and functionalities designed to make restaurant management as easy as possible, including:

- An application for table reservations for customers which is updated in real time regarding the orders made, allows the restaurant employees at any given moment to follow the future orders.
- An interface which presents the future reservations, and current orders for the employees.
- Updating and creating a menu for the restaurant, updating dishes that ran out during a shift, adding allergens to certain dishes if they exist, and all this both in the interface of the employees and in a friendly interface for the customers.
- A food ordering system with a convenient interface for the waiters in which an option will be given to sit in the restaurant at a specific table or order takeaway/delivery for a customer.
- A convenient and expandable database that keeps within it all the necessary information for the restaurant, both the details of customers and employees, the various dishes available in the restaurant, future reservations, and existing orders.

## Beyond the scope

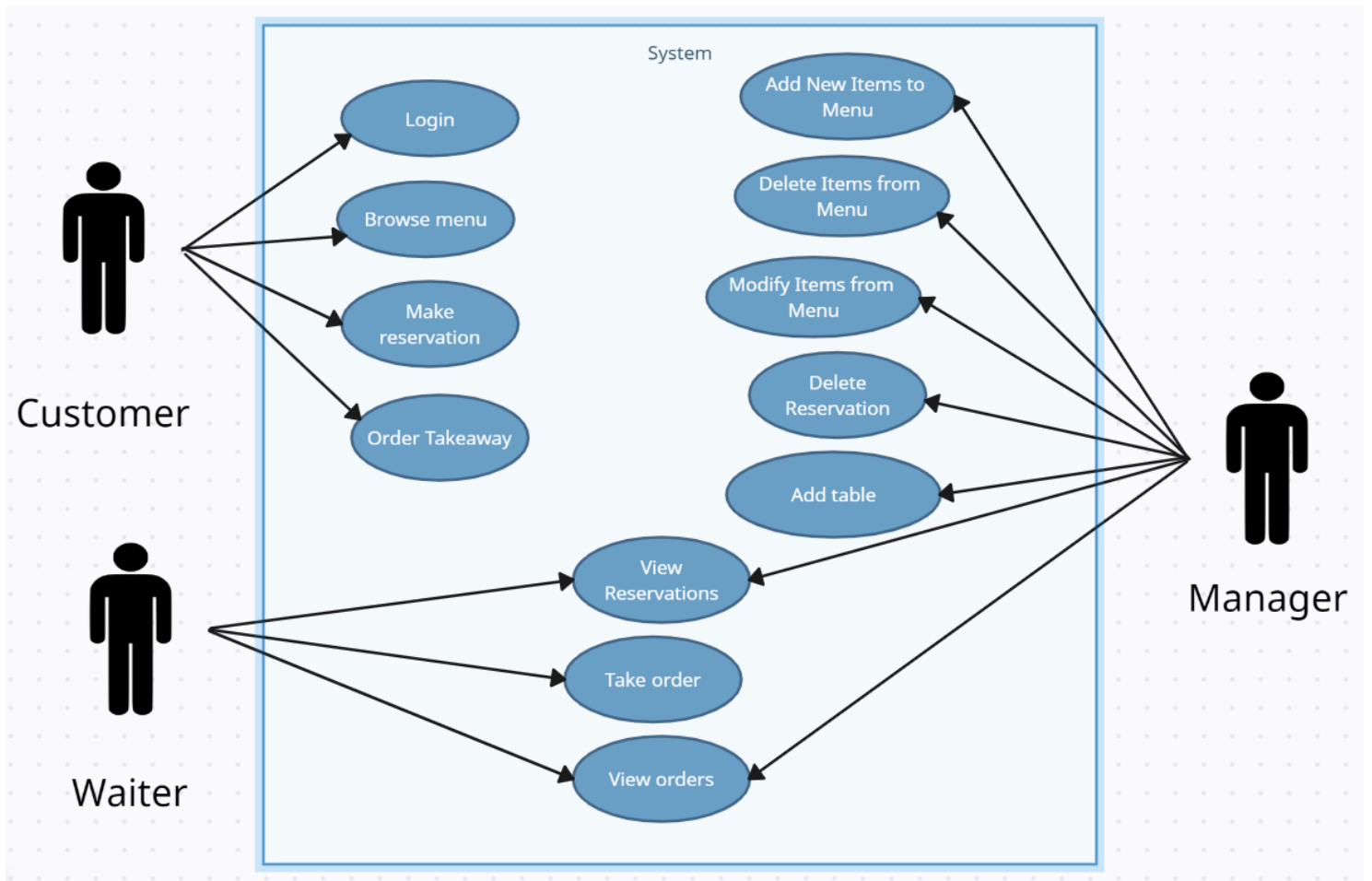
- The system does not work with payments. Payments will be made to an external company that will coordinate with the system and close open orders accordingly.
- The system does not handle the ordered deliveries, but only holds a customer address and the delivery itself will be made by couriers who do not work under the restaurant.
- The system relies on the reliability of the users and does not deal with anything related to data security.
- The kitchen staff is out of scope. The system does not manage the orders from the moment they are received by the kitchen until they are returned to the staff.

## Actors and Goals

Actor name	Role	Description	Goals
Customer	Primary	The most important character in a restaurant is its customers. People who want to go to the restaurant, take away or dine in.	Customers generally represent the end users who interact with the system to access various services provided by the restaurant. The customer's interaction with the restaurant starts with placing the order until the stage when he actually arrives to eat. The main goal of the customer actor is to fulfill his food needs and preferences using the functionality offered by the system. Customers will be able to use our system to book a place in the restaurant and order take-away or delivery to their home.
Manager	Primary	The manager is responsible for everything that happens. Posting a work arrangement for waiters, making changes to the menu, monitoring customer's reservations and orders, updating the database, etc.	Managing the shift in real time in a reliable and efficient manner. To follow all the necessary details for running a restaurant, whether following up on open or closed orders, managing the dishes on the menu. Manager plays a crucial role in overseeing restaurant operations, managing staff, and ensuring the overall efficiency and profitability of the restaurant. The manager will be able to use our system to have an overview of everything that is happening, in one place.
Waiter	Primary	Working in a restaurant. The part that connects the customers and the kitchen. RSATE allows the waiters to RSATE makes their job easier for the waiters by providing the customer with most of the details he needs, with a detailed breakdown of the dishes currently available, a detailed breakdown of the process, etc.	Waiters represents the restaurant staff responsible for managing customer orders and ensuring a smooth dining experience. The primary goal of the Waiter actor is to efficiently handle customer interactions, manage orders, and provide high-quality service. The waiters are the ones who translate the customers' requests to the rest of the staff while using RSATE, and are the ones responsible for making the customer enjoy themselves and want to return to the restaurant in the future. Using our system, waiters can view incoming orders in real time, prepare for the arrival of customers, make orders for the customer with accurate detail for each item that leaves no room for mistakes.

## Functional Requirements

Use case diagram



## Use case details

### Make a reservation

**Goals:**

- Presentation of the options available for booking a place in the restaurant for the customer.
- Real-time updating of the system for a number of customers wishing to place an order at the same time.
- Allowing the customer to choose a date, time and preferences.
- Presentation of the updated orders for the restaurant employees so that they can prepare themselves accordingly during the shift.

**Participating actors:** Customer

**Basic workflow :**

- 1a. Customer logs in to the system.
- 2a. The customer navigates to the reservation section of the restaurant's application.
- 3a. The system displays all available hours within the next 7 days.
- 4a. the user to picks preferred day, time and number of guests.
- 5a. The system asks if the customer has any comments or preferences that he would like to send to the restaurant regarding his reservation.
- 6a. Customer types any special requests or preferences (e.g., seating preferences, dietary requirements).
- 7a. The customer reviews their reservation details and submits the reservation request.
- 8a. The system asks to enter the customer's phone number.
- 9a. The customer enters his contact information.
- 10a. Once validated, the system confirms the reservation and generates a unique reservation ID, and links an available table to the reservation with matching capacity.
- 11a. A confirmation message is displayed to the customer, and a confirmation email with the details is sent.

### View reservations

**Goals:**

- Allows the restaurant staff the option to watch the upcoming reservations, manage the shift according to the number of people who are going to arrive and prepare accordingly.

**Participating actors:** Waiter/Manager

**Basic overflow:**

- 1a. User logs in as a waiter/manager
- 2a. User navigates to reservations section in the system.
- 3a. System displays an interface with all upcoming reservations.
- 4a. The Waiter or the Manager viewing the reservation list through their respective interface.
- 5a. Staff prepare for the upcoming reservation by ensuring the table is set up according to the customer's preferences and any special requests are noted.

## Delete reservation

### Goals:

- Allows the manager to delete reservations from the database in case of cancelation.

### Participating actors: Manager

### Basic workflow:

- 1a. User logs in as manager.
- 2a. User navigates to reservations section in the system.
- 3a. The manager viewing the reservation list through their respective interfaces.
- 4a. Manager picks reservation to delete, and presses the delete button.
- 5a. System removes reservation details from the database.

## Take-away ordering

### Goals:

- Allow customers to place orders for take-away or delivery.
- Ensure orders are processed and communicated to the kitchen staff efficiently.
- Provide real-time updates to customers about the status of their orders.

### Participating actors: Customer, Waiter.

### Basic workflow:

- 1a. Customer logs in to the system.
- 2a. The customer navigates to the menu section of the restaurant's application and enters the takeaway/delivery section.
- 3a. The system presents a detailed interface that shows all the dishes currently available in the restaurant.
- 4a. The customer explores the menu and views dishes with descriptions and prices.
- 5a. Customer selects desired dishes and drinks.
- 6a. The system asks if there are any comments or special requests regarding the selected dishes.
- 7a. Customer Indicates any special requests if exists.
- 8a. System asks the customer to choose between pick up time or delivery, Customer chooses **pick up option**.
- 9a. System asks the customer to insert contact information (. e.g. phone number).
- 9a. Customers insert all relevant data.
- 10a. The customer reviews their order details and submits the order request.
- 11a. System presents payment interface.
- 12a. Customer inserts his payment details.
- 13a. Once validated, the system confirms the order and generates a unique order ID, and sends to the customer detailed email with order summary and receipt.

### Alternate workflow:

- 8b. System asks the customer to choose between pick up time or delivery, customer chooses the **delivery option**.
- 9b. System asks for customers contact information (. e.g. phone number, address)
- 10b. Customer inserts all relevant data.
- 11b. System saves the location of the customer in the order details.
- 12a. The customer reviews their order details and submits the order request.
- 13a. System presents payment interface.
- 14a. Customer inserts his payment details.
- 15a. Once validated, the system confirms the order and generates a unique order ID, and sends to the customer detailed email with order summary and receipt, and also contacts delivery company for delivery.



## Dine in ordering

### Goals:

Placing an order with a waiter after the customer has browsed the menu and chosen his favorited dishes. The waiter takes care of physically placing the order and sending it to the kitchen for preparation.

**Participating actors:** Customer, Waiter

### Basic workflow:

- 1a. Customer calls the waiter when he is ready to order.
- 2a. The waiter approaches to the table and asks the customer what dishes he would like to order.
- 3a. The customer lists the dishes he would like to order or was interested in.
- 4a. The waiter selects these dishes in the ordering app and verifies with the customer.
- 5a. The customer confirms the order.
- 6a. The waiter sends the order with the table number and a unique order number to the kitchen for preparation.

### Alternate workflow:

- 6b. The customer finds a mistake in the waiter's order and requests a change.
- 7b. The waiter presses a delete button on the screen, for the wrong dishes and changes the order according to the customer's requests.

## Browse and prepare to order

**Goals:** The customer browses the menu before placing an order. He looks at the pictures of the dishes, reads the description, checks for allergens and chooses his favorited dishes in order to shorten the ordering process with the waiter and optimize the service.

**Participating actors:** Customer.

### Basic workflow :

- 1a. The user navigates to the "Show menu" button in the system.
- 2a. The system displays the menu of all dishes currently available in the restaurant, divided into categories (starters, mains, drinks, etc.), With a description for each dish, presentation of allergens if any and a picture, and "Request a waiter" button.
- 3a. The user browses the menu, reads the description and views the pictures.
- 4a. The user presses the "call the waiter" button when he is ready to order.

## Add/View tables

**Goals:** Allow the manager of the restaurant to manage the tables that exists in the restaurant.

**Participating actors:** Manager

### Basic workflow :

- 1a. The user logs to the system in as a manager
- 2a. User navigates to the tables section in the application.
- 3a. System shows interface with all tables that exists in the restaurant, with table numbers for each, capacity and description for each table.
- 4a. User presses on "add table" button.
- 5a. System asks to insert all relevant data about the table – table number, capacity and description (near a window, aisle, outside, etc.).
- 6a. User inserts all relevant data and presses on save.
- 7a. System saves the data and adds the table to the list of tables in the restaurant.

## Add menu item

**Goals:** Allows managers and admins to add new menu items – food or drinks, with detailed description, price and picture.

**Participating actors:** Manager

**Basic workflow :**

- 1a. User logs in as manager and navigates to menu section in the interface.
- 2a. System presents interface with all menu items.
- 3a. User presses on “add item” button.
- 4a. System asks the user to insert all relevant data, such as – price, name, category and a URL for picture of the item.
- 5a. User presses on save button.
- 6a. System stores data, and adds the new item to the relevant section in the menu.

## Modify menu item

**Goals:** Allow the restaurant manager to change or delete menu items according to the restaurant's inventory. All this in order to prevent mistakes in customers' take-away orders if there is a lack of a certain item or its description needs to be changed.

**Participating actors:** Manager

**Basic workflow :**

- 1a. The user logs in to the system as manager.
- 2a. User navigates to the menu interface and presses options button.
- 3a. The user browses the menu, pick an item to modify
- 4a. System presents interface with item details, such as price, description and a delete option.
- 5a. User presses on relevant option, makes the required changes and presses on save button.
- 6a. System updates the database accordingly and updates the menu.

## Non-Functional Requirements

Requirement number	Requirement type	Requirement Description
1	Supportability	In order to run the system, a docker container must be used.

## System bonuses - Appendix no. 1



- **Sending emails** - the system will send emails to users when performing various actions:
  - When registering a new customer, the restaurant will send an email with the user's details.
  - When making a reservation, the restaurant will send a confirmation email with the details of the order.
  - When placing an order, an email will be sent detailing the ordered items and a receipt.
- **Saving location** - when placing a delivery or takeaway order, the system will save the landmarks from which the order was placed.
- **Client** – we build a client using android OS which supports multiplatform options that our system offers:
  - Making reservations
  - Browsing the menu
  - Dine-in/Takeaway orders
  - Modifying the menu
  - Viewing orders and reservations

## Technologies Document – Appendix no. 2

Name	System Part	Description and Role
Spring boot	Server	an open-source Java framework used for programming standalone, production-grade Spring-based applications with minimal effort. Handles superapp server side, rest API and DB crud.
Postgres SQL	Database	a free and open-source relational database management system (RDBMS) emphasizing Handles our DB .extensibility and SQL compliance and holds all our data.
Junit	QA	JUnit is a test automation framework for the Java programming language. We used JUnit for unit testing.
Gradle	Server	Gradle is a build automation tool for multi-language software development. It controls the development process in the tasks of compilation and packaging to testing, deployment, and publishing. Used for building libraries and import server and client.
Android Studio	Client	An Official integrated development environment (IDE) for Google's Android operating system. Used for building the client.
Trello	Development	A web-based project management and collaboration tool that uses boards, lists, and cards to organize tasks. Used for organization, assigning tasks and collaboration among team members.
Bit Bucket	Development	A web-based platform for hosting Git repositories and collaborating on code. Provides version control and repository hosting for source code, enabling collaboration on projects.
Slack	Development	Slack is a cloud-based team communication. Used for communication among team members.
Docker	Server	Docker is a platform that enables developers to package, deploy, and run applications in isolated and lightweight containers. Streamlines application deployment by creating lightweight, portable containers for execution.

## Project report – Appendix no. 3

### Team members

<i>Student's name</i>	<i>ID</i>	<i>Email</i>	<i>Kanban Avatar</i>	<i>Role</i>
Aviad Korakin	318383544	Aviad.korakin@s.afeka.ac.il		Team lead, Product owner, Team
Daniel Raby	314924358	Daniel.raby@s.afeka.ac.il		DBA, Team, DEVOPS
Tamar Yusufov	316525815	Tamar.yusufov@s.afeka.ac.il		QA, System Architect, Tech writer, Team
Itay Biton	315176859	Itay.biton@s.afeka.ac.il		UI/UX, Team
Oded Ginat	209372507	Oded.ginat@s.afeka.ac.il		Scrum master, Team

## General Summary of work on sprint 4

- **What went well for the team and should be continued the on next phases of work**
  - Tasks to be done were clear to everyone. Each team member knew what he had to do and met the times that set for him.
- **What should be improved in team work**
  - More teambuilding activities to make the atmosphere more friendly.
- **What problems did the team encountered through this phase of work**
  - We had to write the client from the beginning because what we did from the beginning didn't integrate with our server.
- **Why did we not complete all planned work**
  - We didn't complete the shift management app, due to lack of time.

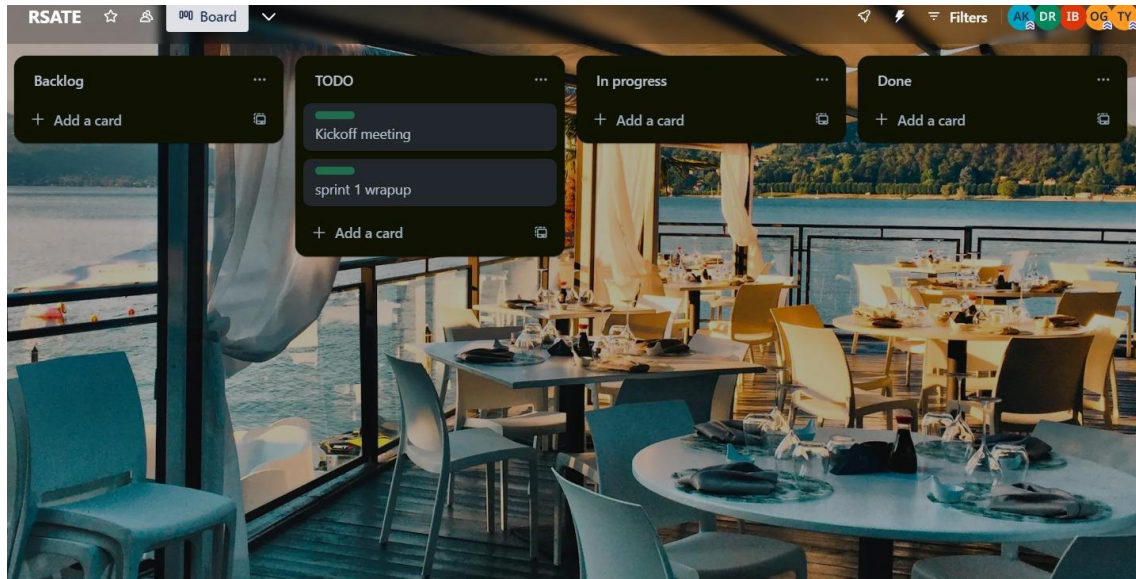
## General Summary of work on the project

- **What went well for our team throughout the semester and we would like to preserve in the following projects that we will participate n the industry?**
  - Clear division of tasks.
  - Everyone knew what they were supposed to do.
  - Everyone felt that they could turn to the other group members in order to ask, consult or get help.
- **How can we improve in future projects?**
  - We will try to communicate more between the team members in order to optimize the work on each part of the code.
  - We will work more on teambuilding activities between the team members in order to make the atmosphere lighter and more friendly.
- **What we enjoyed the most?**
  - The learning process.
  - The understanding that a complete software engineering project contains so many different departments, which require integration between them.
  - We enjoyed the fact that there are many options for expanding a project, and that the options are endless.
- **What would we do differently?**
  - We would pay more attention to the client from the beginning of the semester.
  - We would use wisely in kanban boards in order to divide the work between team members.
- **Remote work**
  - Most of the work and communication between us was done remotely.
  - Working remotely made it easier for us by saving a lot of time in scheduling physical meetings, getting to specific places and leaving more time for work.
  - We mainly used WhatsApp, Discord and Slack for remote work.

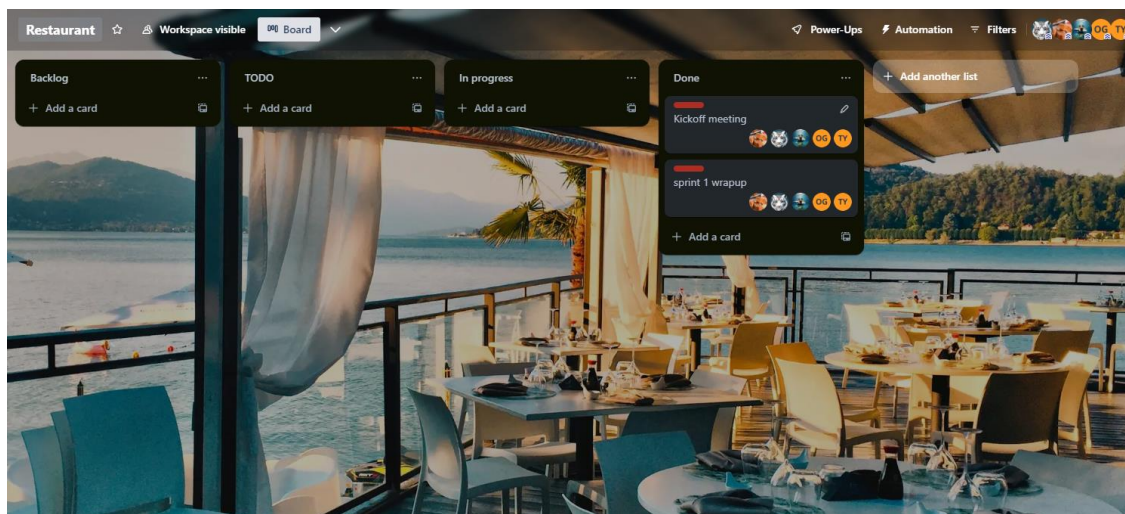
## Kanban Boards

### Sprint 1

01/05/2024



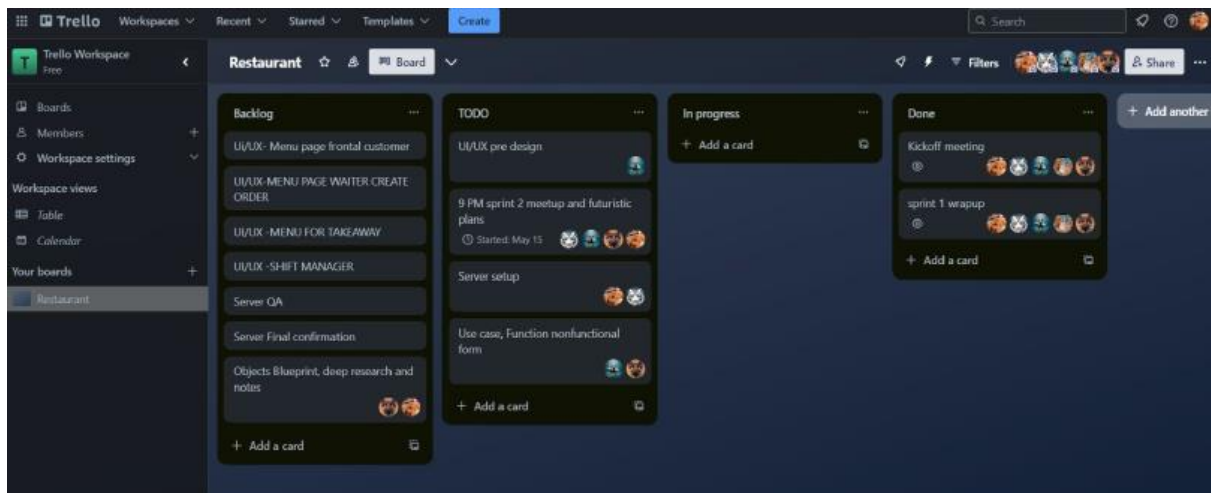
11/05/2024



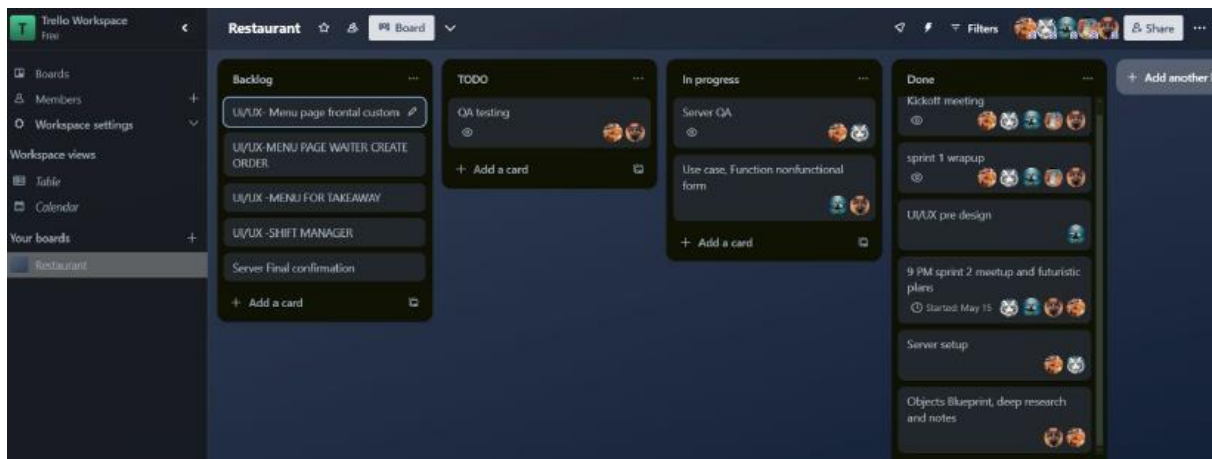


## Sprint 2

16/05/2024



04/06/2024



## Sprint 3

06/06/2024

**Restaurant** ☆ Workspace visible Board

**Backlog**

- UI/UX-MENU PAGE WAITER CREATE ORDER
- UI/UX -SHIFT MANAGER
- SPRINT3-SERVER QA
- SPRINT3-FINAL SERVER CONFIRMATION
- SPRINT3-QA
- + Add a card

**TODO**

- Connection between our objects to project
- Use-case diagrams
- Write Non-Functional Requirements
- SPRINT3-UI/UX RESEARCH
- Ui/UX- Menu page frontal customer
- Flutter - rest api connection
- QA testing
- SPRINT3- SERVER UPDATE
- + Add a card

**In progress**

- UI/UX -MENU FOR TAKEAWAY
- Server QA
- Functional requirements
- + Add a card

**Done**

- Kickoff meeting
- sprint 1 wrapup
- UI/UX pre design
- 9 PM sprint 2 meetup and futuristic plans
- Server setup
- Objects Blueprint, deep research and notes
- Use case, Function nonfunctional form
- Server Final confirmation
- + Add a card

25/06/2024

**Restaurant** ☆ Workspace visible Board

**Backlog**

- UI/UX-MENU PAGE WAITER CREATE ORDER
- UI/UX -SHIFT MANAGER
- + Add a card

**TODO**

- Connection between our objects to project
- + Add a card

**In progress**

- UI/UX -MENU FOR TAKEAWAY
- + Add a card

**Done**

- Kickoff meeting
- SPRINT3-UI/UX RESEARCH
- Write Non-Functional Requirements
- Ui/UX- Menu page frontal customer
- sprint 1 wrapup
- Use-case diagrams
- SPRINT3-SERVER QA
- SPRINT3-FINAL SERVER CONFIRMATION
- SPRINT3-QA
- SPRINT3- SERVER UPDATE
- + Add a card

## Sprint 4

28/06/2024

Restaurant Workspace visible Board

Power-Ups Automation Filters

Backlog

- Last meeting - final confirmation
- Presentation planning
- Presentation practice
- + Add a card

TODO

- Server Final confirmation
- SPRINT4- SERVER UPDATE
- UI/UX- Menu page frontal customer
- UI/UX-MENU PAGE WAITER CREATE ORDER
- Fix Non-Functional Requirements
- Fixing documents - notes from last sprint
- SPRING3 - SERVER FIXES
- UI/UX --reservations
- Final document - writing report
- UI/UX - Takeaway order
- UI/UX --MENU FOR TAKEAWAY
- + Add a card

In progress

- Complete usecases
- QA testing
- + Add a card

Done

- + Add a card

+ Add another list

16/07/2024

Restaurant Workspace visible Board

Power-Ups Automation Filters

Backlog

- + Add a card

TODO

- + Add a card

In progress

- + Add a card

Done

- Last meeting - final confirmation
- Presentation planning
- Presentation practice
- Complete usecases
- Server Final confirmation
- Final document - writing report
- UI/UX - Takeaway order
- SPRINT4- SERVER UPDATE
- UI/UX --MENU FOR TAKEAWAY
- UI/UX --reservations
- Fix Non-Functional Requirements
- UI/UX- Menu page frontal customer
- UI/UX-MENU PAGE WAITER
- + Add a card

+ Add another list