# **[Project Logo (optional)]**

**Restaurant Reservation System**

**Software Requirements Specification**

**Version 2**

**Team Number: 18**

**Project Manager: Anthony Epps**

**Mentor: Rizzo**

**Team Members: Dakotah Campbell, Mangala Dadallage, Kevin Hernandez Gaspar, Jason Lee, Prince Varghese, Luke Wainwright**

**Revisions**

| **Version** | **Primary**  **Author(s)** | **Description of Version** | **Date Completed** |
| --- | --- | --- | --- |
| 1 | Group 18 | Initial Version (No High Level Design) | 10/29/2023 |
| 2 | Group 18 | Revised Version (with High Level Design | 11/5/2023 |

**Review History**

| **Reviewer** | **Version Reviewed** | **Date** |
| --- | --- | --- |
| Susan Rizzo | Version 1 | 10/19/23 |
| Susan Rizzo | Version 2 | 11/1/23 |

**Table of Contents (version 2)**

1. Introduction
   1. Project Objectives
   2. Project Scope
   3. Project Overview
2. Project Description
   1. Project Features / Functions
   2. User Stories
   3. Use Cases
3. Team Collaboration and Documentation
4. Project Management
5. Requirements Specification
   1. Business Requirements
   2. User Requirements
   3. Functional Requirements
   4. Non-Functional Requirements
   5. Implementation (Performance) Requirements (Optional)
6. High-level Design
   1. Security (Required)
   2. Hardware (Required)
   3. User Interface (Required)
   4. Architecture (Required)
   5. Database (Required)
   6. Top-level Classes (Required)
   7. Data Flow and States (Required)
   8. Reports (Required)
   9. Internal Interfaces (Optional)
   10. External Interfaces (Optional)
   11. Other Outputs (Optional)
   12. Configuration Data (Optional)
   13. Training (Optional)
7. Appendixes (Optional)
8. **Introduction**
   1. Project Objectives

The goal of this project is to develop a web application that a fine dining restaurant can use to allow its customers to make a reservation and start placing their order so when they arrive, there is minimal wait time.

* 1. Project Scope

This project is to allow a customer to create a reservation at a restaurant, as well as starting the order process. This includes the initial table reservation, and initial order placing. This effectively “removes” the initial pattern of walking in, sitting down, and looking at the menu and all waiting involved.

* 1. Project Overview

The goal of this project is to develop a web app quickly, efficiently and to meet the goals of the end user. The web app will allow customers to log in, see a model of the table and all available seating, and make a reservation. They then will be allowed to look at the menu and place an initial order, so that when they arrive, the food will be ready to be brought out, reducing wait times to essentially zero!

1. **Project Description**
   1. Project Features / Functions

* User Login: The user will be able to enter credentials and log into their account to start the process for creating a reservation.
* Table Reservation: The application will display a screen that shows all currently available tables within a selectable time frame, allowing them to accurately confirm where they will be seated during their requested time slot.
* Menu: Once the table is selected, the user will then be presented with a menu that has an image of the item and its price and description.
  1. User Stories
     1. As a customer, I want to place my order ahead of time so that when I arrive, I do not have to wait for my food.
     2. As a customer, I want to be able to make sure I have a table reserved so that when I arrive at the restaurant, I will not have to wait.
     3. As a staffer, I want to be able to see how many reservations I will have within a certain time window so that I may properly schedule the correct number of staff to accommodate guests.
     4. As a staffer, I would like to have a system in place to require customers to confirm their reservation to properly run and manage the flow of restaurant traffic.
     5. As a staffer, I would like to be able to edit and manipulate tables as needed, for example in case of large party reservations via other channels.
  2. Use Case
     1. Use Case 1:

| Description | Customer Creates reservation |
| --- | --- |
| Pre-Condition | Customer has no table reserved |
| Post Condition | Customer has table reserved |
| Basic Path | Pick table on app |
| Alternative paths |  |
| Exception | No tables left |

* + 1. Use Case 2:

| Description | Customer places order |
| --- | --- |
| Pre-Condition | Customer has a table, with no order |
| Post Condition | Customer has table and order |
| Basic Path | Place order through app |
| Alternative paths | Upon arrival, place another order |
| Exception | Item out of stock, Payment Declined |

* + 1. Use Case 3:

| Description | Payment Verification - Failed |
| --- | --- |
| Pre-Condition | Tables is selected and payment is loaded |
| Post Condition | Payment is unverified and Table is not reserved – customer is sent a text to provide correct payment or table will not be reserved |
| Basic Path | Staff checks tables that are empty, updates them in app, customers pays and makes reservation |
| Alternative paths |  |
| Exception |  |

* + 1. Use Case 4:

| Description | Staff updates available tables |
| --- | --- |
| Pre-Condition | Tables are shown as all full |
| Post Condition | Available tables are now shown |
| Basic Path | Staff checks tables that are empty, updates them in app |
| Alternative paths |  |
| Exception |  |

* + 1. Use Case 5:

| Description | Payment Verification - Accepted |
| --- | --- |
| Pre-Condition | Tables is selected and payment is loaded |
| Post Condition | Payment is verified and Table is reserved |
| Basic Path | Staff checks tables that are empty, updates them in app, customers pays and makes reservation |
| Alternative paths |  |
| Exception |  |

1. **Project Collaboration and Documentation**

For project collaboration, we will be using Microsoft Teams and GitHub. Microsoft Teams will be used for team communication. We will use GitHub to host all repositories and documents related to the project. For diagrams, we will use LucidChart, Mermaid, Excalidraw and Google Docs.

1. **Project Management**

GitHub Projects will be used for project management. We will be using the Project Scrum board and GitHub Issues to track and manage all current and future tasks.

1. **Requirements Specification**
   1. **Business Requirements**

| Requirement ID | Requirement Description | MOSCOW |
| --- | --- | --- |
| BR1 | Allow customers to put in their order remotely | M |
| BR2 | Restaurant workers need to be able to access orders before customers arrive | M |
| BR3 | Customers can pay for their meals through the application | M |
| BR4 | Allow customers and staff be able to communicate through the application | S |
| BR5 | Enable customers to have food paid for and ordered before they arrive at restaurant | S |

* 1. **User Requirements**

| Requirement ID | Requirement Description | MOSCOW |
| --- | --- | --- |
| UR1 | Require ordering food and make payments online. | M |
| UR2 | Require including add on items to the main order. | S |
| UR3 | Screen to see whether the order is started preparation, in progress, or ready to pick up. | C |
| UR4 | The payment receipt requires a reference number to track the order. | C |
| UR5 | Require a user account to collect points for free meals, review previous orders, and write reviews. | S |

* 1. **Functional Requirements**

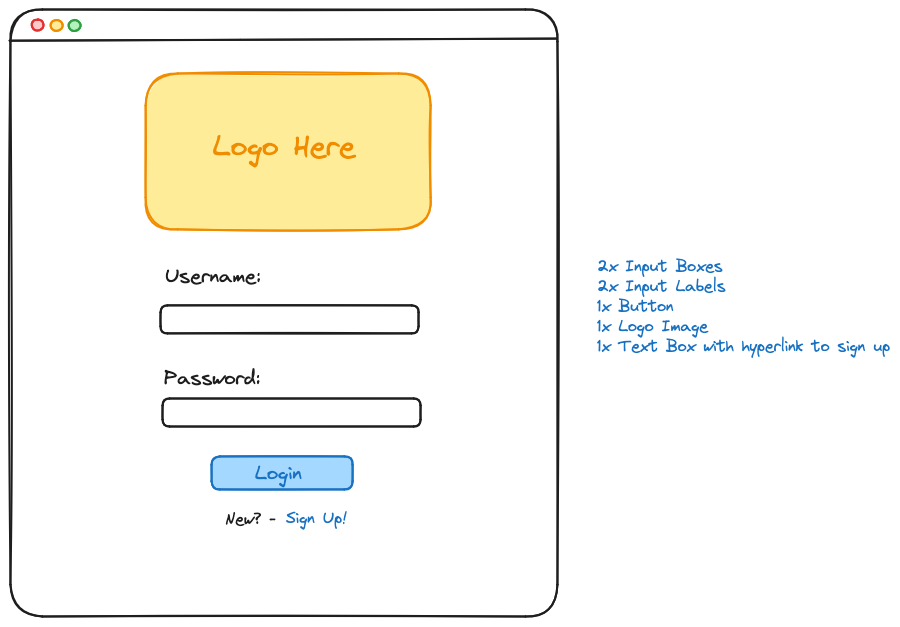
| Requirement ID | Requirement Description | MOSCOW |
| --- | --- | --- |
| FR1 | Must enable customers to make reservations online and place food orders in advance, allowing customization of orders and specifying dining times. | M |
| FR2 | Provide a user-friendly interface with detailed menu descriptions and images, allowing customers to easily customize their orders based on dietary preferences and portion sizes. | M |
| FR3 | Must be integrated with the restaurant's kitchen operations to provide real-time order updates, ensuring that food is prepared and ready for pickup at the specified reservation time. | M |
| FR4 | Send immediate order confirmations and periodic status updates to customers via email, SMS, or the app, keeping them informed about the status of their orders. | S |
| FR5 | Automatically assign tables to pre-ordering customers based on their reservation times and party size, as well as notify customers when their table is ready for seating. | C |

* 1. **Non-Functional Requirements**

| Requirement ID | Requirement Description | MOSCOW |
| --- | --- | --- |
| NFR1 | Processing the reservations must be completed as fast as possible with little error to avoid accidental duplicate reservations | M |
| NFR2 | Order Screen must be easy to understand on how to add or remove items | M |
| NFR3 | When food is added to the order, it should be immediately displayed, and total price should be updated dynamically | S |
| NFR4 | Menu should be in an easy to view manner with all necessary information visible | S |

* 1. **Implementation (Performance) Requirements (Optional)**

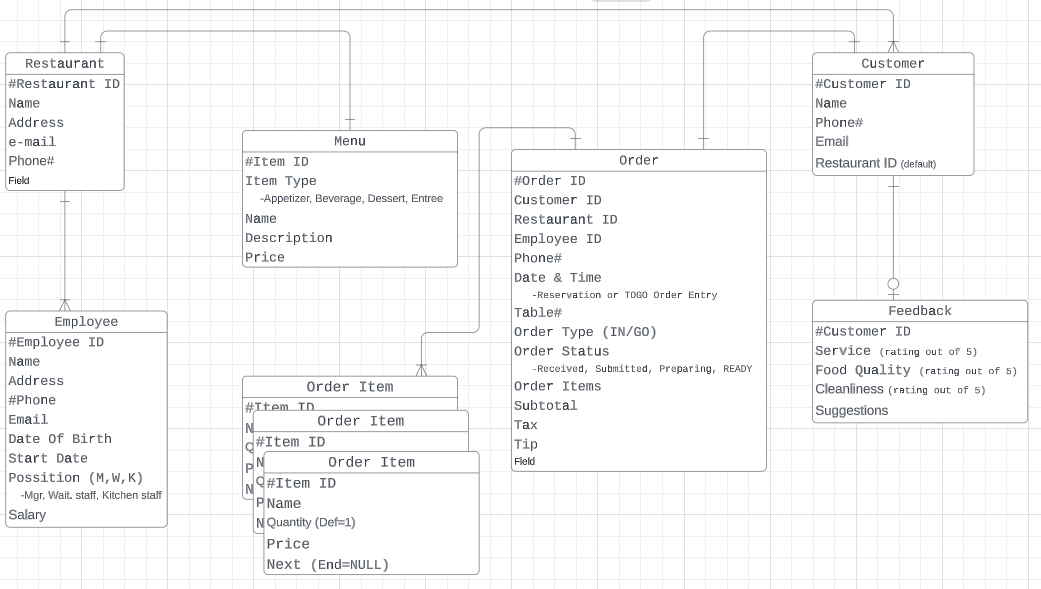
| Requirement ID | Requirement Description | MOSCOW |
| --- | --- | --- |
| IR1 | Integrate with a dependable payment gateway to securely process online payments. | M |
| IR2 | Real-time notifications to staff members when a new reservation is made, or an order is placed. | M |
| IR3 | Strong security measures, including encryption of sensitive data and protection against common web vulnerabilities. | M |
| IR4 | Role-based access control to ensure that only authorized staff members can perform administrative actions. | M |
| IR5 | Handle sudden fluctuations in traffic without experiencing significant slowdowns or downtime. | S |
| IR6 | Efficient search functionality to allow users to quickly find specific menu items or filter them based on categories, ingredients, or dietary preferences. | S |
| IR7 | Smooth scrolling and responsive design for seamless user experience across different devices. | S |
| IR8 | Caching mechanism to optimize the loading time of frequently accessed data, and to improve overall application performance | C |
| IR9 | Efficient data compression to minimize the size of data transferred over the network. | C |
| IR10 | Support for multiple languages to cater to international/tourist customers. | C |
| IR11 | Optimized image loading and rendering to ensure fast and seamless display of high-resolution images of menu items. | W |

1. **High-level Design**
   1. **Security (Required)**
      1. Login Management: In today's world, security of personal data is at the forefront of everyone's minds. I feel like for this application, we would want to implement something like NextAuth.js or another authentication framework to handle authentication and data management so we do not have to worry about it. This will let us focus on building just the app and not the app and then a safe, secure authentication system.
      2. Payment Management: Similar to the login system, we do not want to “own” responsibility of storing, securing and accessing customer financial data. This can be offloaded to a service like Stripe.
   2. **Hardware (Required)**
      1. Computer (Restaurant Use)
      2. Monitor (Restaurant Use)
      3. Tablet (Restaurant and End User)
      4. Mobile Phone (End User)
      5. Server in Cloud (App)
      6. Database API (App)
   3. **User Interface (Required)**
      1. Login Page: The login page will feature a simple page with app logo front and center. It will feature 2 input boxes for username and password. Below these there will be a login button to start the login process. Underneath that there will be a hyperlink to take to the sign-up page.
      2. Sign Up Page: The Sign-up page will feature the logo in the top left corner of the app. There will be 4 input boxes (with corresponding labels) and 3 dropdown menus. There will be a submit button on the bottom of the page, followed by a hyperlink to the sign in page.
      3. Login Page Wireframe:
      4. Sign Up Page Wireframe:A screenshot of a login form

         Description automatically generated
   4. **Architecture (Required)**

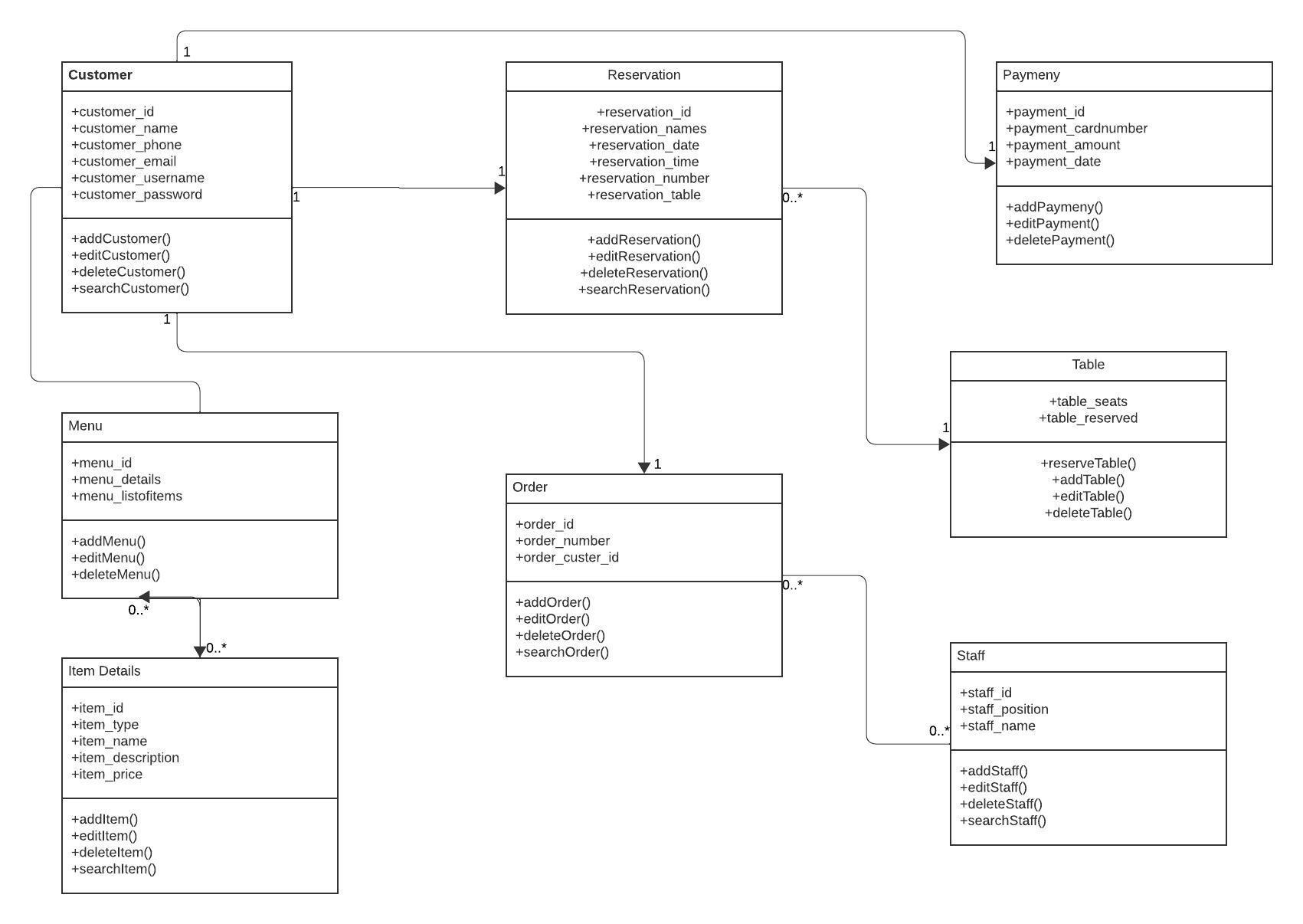
For the fine dining restaurant web application, a multi-tier architecture would be suitable to meet the project objectives and requirements. This architecture divides the application into distinct layers, each responsible for a specific set of functionalities.

* + 1. Specifications
    2. Multi-Tier Architecture:
* Presentation Layer
* Application Layer
* Data Layer
* Service Layer (Optional)
  + 1. Presentation Layer:
* This is the user interface layer where customers and staff interact with the application.
* It includes the web-based user interface, mobile apps, and other client-side components.
* Users can log in, view available tables, access the menu, place orders, and communicate through the application.
* The user interface should be user-friendly, allowing customers to easily navigate the reservation and ordering process.
  + 1. Application Layer:
* This layer acts as an intermediary between the presentation layer and the data layer.
* It contains the business logic of the application, ensuring that user requests are processed correctly.
* User stories such as creating reservations, placing orders, and updating table availability are handled here.
* It's responsible for communication with the database, external payment gateways, and notification services.
  + 1. Data Layer:
* This layer stores and manages data related to reservations, menu items, user accounts, and more.
* It includes a database to store user information, reservations, menu items, and order details.
* Data-centric operations such as storing reservations, managing menu items, and updating table availability are carried out here.
  + 1. Service Layer (Optional):
* This layer can provide additional services like real-time notifications and external integrations.
* For instance, it can handle notifications to staff members when reservations or orders are placed, integrating with a dependable payment gateway for processing online payments, and integrating with email/SMS services for customer updates.
  + 1. Overview:
* Users (customers and staff) access the application through the Presentation Layer. Customers can log in, make reservations, place orders, and interact with the menu.
* The Presentation Layer communicates with the Application Layer, which contains the core business logic. For example, it handles table reservation, order placement, and staff updates regarding table availability.
* The Application Layer interacts with the Data Layer to store and retrieve data related to reservations, menu items, user accounts, and order information.
* Additional services such as real-time notifications and payment processing are handled by the Service Layer.
* Collaborative tools such as Microsoft Teams and GitHub are used for project collaboration and documentation.
* Project management is facilitated using GitHub Projects, which includes Scrum boards and GitHub Issues to track and manage tasks.
  + 1. This multi-tier architecture aligns with the project's objectives and requirements, ensuring efficient management of reservations and orders, a user-friendly interface, and seamless communication between users and the application. It also provides the necessary flexibility to accommodate potential future requirements and scalability as the restaurant's business grows.
  1. **Database (Required)**
     1. The application will utilize the PostgreSQL database.
     2. The database has seven entities: Restaurant, Employee, Customer, Menu, Feedback, Order Item & Order. Many relationships are one to one or one to many relationships, and there is one number of one to one or one to zero relationship between the Customer and the Feedback.
     3. All the reservations, dine in & TOGO orders are placed in ‘Order’ entity. Moreover, the customer can see whether the status of the order is received, submitted, prepared or ready.
     4. Entity Relationship Diagram



**Top-level Classes (Required)**

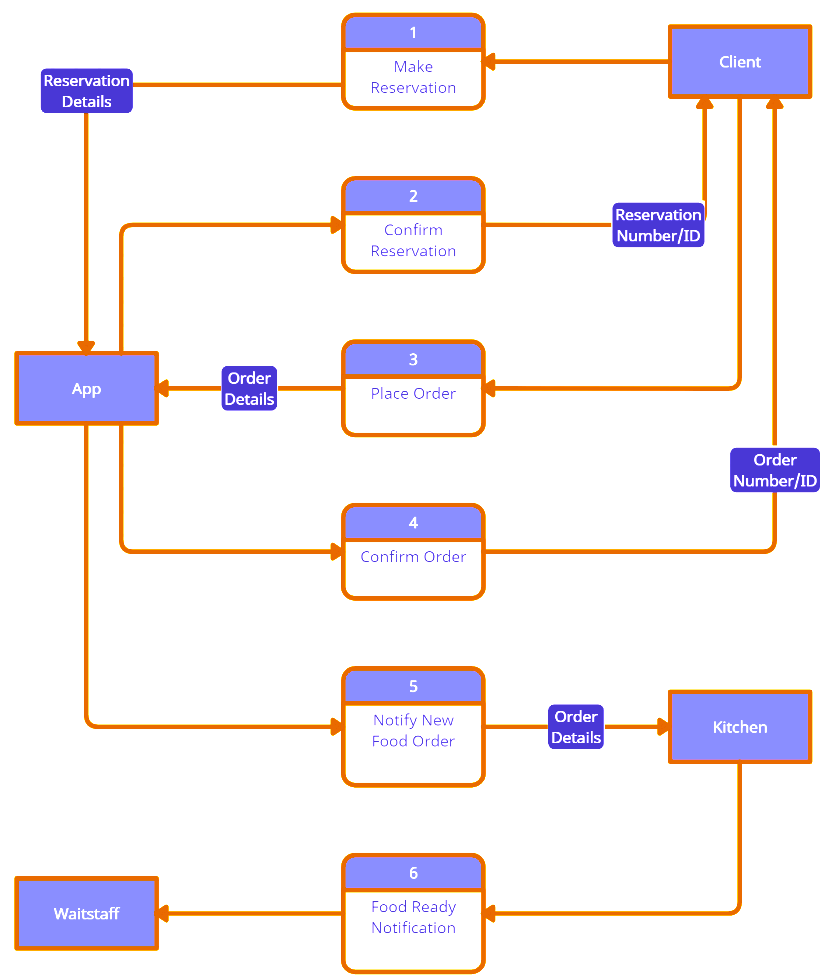
* + 1. Class Design



* 1. **Data Flow and States (Required)**
     1. Data Flow:

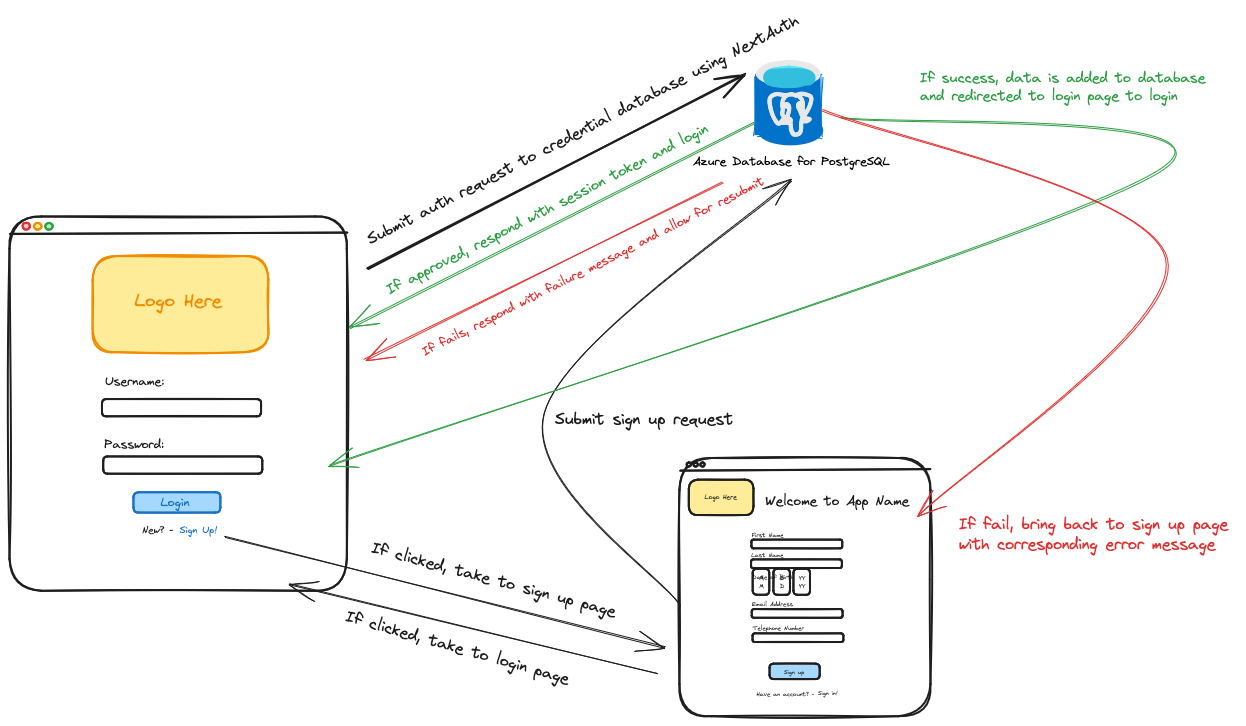
1. Client enters reservation details, which flows to the App.
2. App provides reservation confirmation back to the Client.
3. Client enters order details, which flows to the App.
4. App provides order confirmation back to Client.
5. App notifies the Kitchen of the new food order.
6. The Kitchen notifies the Waitstaff when the food is ready.
   * 1. Reservation:
7. Client requests a reservation by providing their information and reservation details.
8. The App checks the availability of tables for the specified date and time.
9. If a table is available, the reservation is confirmed and the reservation details are stored in the system.
10. If a table is NOT available, the client is informed and given other options.
    * 1. Orders:
11. Client places a food order.
12. The App shows the menu options to the client.
13. The Client selects the desired food items and specifies quantities.
14. The order details are stored in the system, and the clients receive an order confirmation.
    * 1. Food Preparation:
15. The Kitchen is notified of the food order.
16. The Kitchen prepares the ordered food items.
17. Once the food preparation is complete, the order is marked as ready.
    * 1. Food Service:
18. The Waitstaff is notified when the food order is ready.
19. The Waitstaff retrieves the prepared food items from the Kitchen.
20. The food is served to the client at their table.
    * 1. Data Storage:

* Reservation data is stored in a database, including Client information, reservation details, and status.
* Order data is stored in a database, including client information, order details, and status.
* Menu data, including available food items and their descriptions, is stored in a database.
  + 1. Data Flow Diagram:



* 1. **Reports (Required)**
     1. User Reports
     2. The app will be able to create user reports for the staff at the restaurant. The reports will offer information such as total number of app reservations, the most book tables, the most ordered food, the day of most reservations per week.
     3. Customer Reports
     4. The app will offer a report to the customers. The reports will offer information such as total time saved by using reservation service, and receipt history.
  2. **Internal Interfaces (Optional)**

**<** Specify how pieces of the application will interact.**>**

****

* 1. **External Interfaces (Optional)**

**<** Specify how the application will interact with other applications and resources.**>**

**A diagram of a ship

Description automatically generated**

* 1. **Other Outputs (Optional)**

**<** Specify the possible outputs generated by the project>

* 1. **Configuration Data (Optional)**

**<** Specify what data and parameters will be configurable in the configuration page or file.**>**

* Reservation Expiration Time
* Number of Tables
* Number of Seats per table
* Available Menu
  1. **Training (Optional)**

A User's Guide will be provided to customers demonstrating the application functions. Each page of the app will also feature a help button, denoted by a [ ? ] button.

1. **Appendixes (Optional)**