# Restaurant Management System Documentation

## Problem Statement

For this project, our group is tasked with designing a restaurant management system that allows for different roles to be executed by the respective staff. Those roles include: host, waiter, chef, manager, and owner. To perform an action which requires a higher role a user with the role or higher must allow it with authentication. For example, a waiter needs to correct a price for a special but can't as a manager role is required. This means a manager or higher needs to give permission to do so through the system. Each role has their own user interface that allows them to do their job. This system needs to allow switching between other roles. (Group Project)

The roles for each page of the system are as follows:

**Host**
• Logs into system at front of house
• Greets customers
• Checks for open tables
• Takes customer to table
• Marks table is occupied
• When customer leaves marks table as open

**Waiter**
• Logs in to system at front of house
• Should be able to easily put in order
• This can include modifying the order to customer's preferences
• Should be able to add orders after initial order is taken for late walk-ins or dessert
• Needs to know what table to take order to
• Drinks should be separate from meal so refills are kept track of
• Take payment from customer
• Assigned a table/tables to wait

**Chef**
• Logs into system at back of house (kitchen)
• See orders as they come in
• Should be allowed to prioritize orders per table
• Mark orders as in-progress or done

**Manager**

- Can do what waiter does in addition to (is not assigned tables to wait by default)
- Add/Remove workers to system for new hires/fires
- Assigns tables to waiters
- Mark prices for specials
- Modify Menu
    - Add new items
    - Remove Items
    - Change prices

**Owner**

- Can do everything a manager does in addition to
- Add/Remove managers
- Has access to reports
- Reports
    - Show how much of each menu item was ordered during a period of time
    - Shows the amount of money earned from payments

**Admin/Maintenance**

- Has access to underlying server to fix issues
- Is not a worker for the restaurant but for your "Software Firm"
- Should be a separate login from the restaurant so as not to be removed by accident

# Our Overall Goal

For this project, our overall goal was to meet the criteria and develop a working restaurant management system

# Programmers Documentation

Below, is the project documentation that was developed by the programmers by the use of the A.I. *Gemini cli* using the prompt **[“insert AI prompt here”]**. This documentation was then edited by Chatgpt using the prompt, “Can you convert the documentation below into plain text for a google doc. I mean remove the unnecessary spaces and numbers that denote the lines.”

# Restaurant Management System Documentation

This document provides a brief overview of the Restaurant Management System's frontend pages and backend functionality.

## Backend

The backend is a Node.js application using the Express framework. It connects to a Supabase instance for database operations. The main backend file is `backend/src/index.js`.

### API Endpoints

The backend exposes a RESTful API for managing the restaurant's data. The main resources are:

- **`/api/roles`**: Manages user roles (e.g., Chef, Waiter, Manager).
- **`/api/users`**: Manages user accounts.
- **`/api/tables`**: Manages the restaurant's tables, including their status and assigned waiters.
- **`/api/menu_items`**: Manages the menu items.
- **`/api/orders`**: Manages customer orders.
- **`/api/order_items`**: Manages the items within each order.
- **`/api/payments`**: Manages payments for orders.

The API supports standard CRUD (Create, Read, Update, Delete) operations for these resources.

## Frontend

The frontend is a React application that provides the user interface for the system. The main pages are located in the `frontend/src/pages` directory.

**Pages**

- `Home.jsx`: The main landing page of the application.
- `CreateAccount.jsx`: Allows new users to create an account. It includes a form to enter user details and select a role.
- `LoginForm.jsx`: Provides a login form for existing users. Upon successful login, it redirects users to their respective dashboards based on their roles.
- `CreateOrder.jsx`: Allows waiters to create new orders for a specific table. It displays the menu and allows adding items to an order.
- `PageAdmin.jsx`: The admin dashboard, which provides a comprehensive overview of the system. It includes tabs for managing users, tables, orders, and menu items.
- `PageChef.jsx`: The chef's dashboard. It displays a list of orders with their items and status, allowing the chef to track and update the progress of each order.
- `PageHost.jsx`: The host's dashboard. It shows the status of all tables (e.g., available, occupied, needs cleaning) and allows the host to manage table assignments.
- `PageManager.jsx`: The manager's dashboard. It provides tools for managing employees, tables, and the menu.
- `PageOwner.jsx`: The owner’s dashboard. It provides access to high-level reports and employee management features.
- `PageWaiter.jsx`: The waiter’s dashboard. It displays the tables assigned to the waiter and allows them to create and manage orders.

# Instructions for Setting up Software

To set up a local host for our restaurant management system, you will need to pay for our service. After a successful payment, our company will send you a link to our management system's github. From there, you will follow the requisite instructions, installing the specific [free] software and then cloning the git repository onto your device.

It should be noted that this management system does work on all operating systems, and all types of devices that support the aforementioned softwares.

# Common Contingencies for the Software

In case of any type of malfunction or bug, reach out to Alpha Team Technologies at the toll free number (309) 123-4567.

# Project Meeting Notes

Below are the meeting notes that were catalogued throughout the development process of the restaurant management system. Eah week includes tasks, what was discussed during the meeting, and who was present during those meetings.

Week 2

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, Andrew, and Kishan

9/3/25

We decided that our team name to be “Alpha Team”

During our meeting we discussed the beginning stages of our project, the roles that each of us will take part in. Our team leader Josh was assigned roles.

Our group also set up a line of communication and a time for us to meet and work on the project. We will use Whatsapp and Discord to communicate

We will be using Windows as our operating system

The next step will be us for us will be for us to mock up our UI

We were told to avoid the waterfall, agile, and Scrum style. He (Mr. Rogers) said it’s best for us to use an iterative process

We plan on using Vite for the display the frontend of our local host website

Josh will set up a Readme file for our repository

We also exchanged our GitHub Account

We got out of class

Use backend api
Front end react
Dealing with SQL tables will be the most labor intensive part of the project

Week 3 (9/10/25)

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

9/8/25

We had a team meeting and we set up our GitHub Repository

During this discussion we talked about expectations for the project we have planned

We discussed our expectations for progress for our meetings. We plan to have our specific sections of our code for the Monday

As the project lead, Max plans on assigning tasks for each of the members

We will meet At 9:00 PM through Discord every single Monday

Max sent out a mock up draft for the Scope Statement

The team spent the majority of the meeting setting up the proper software for us to begin working on our project

9/10/2025

During class today we discussed our project status report

Max discussed what our responsibilities will be for next week. He said that he will submit the Scope Statement Assignment that's due tonight

We plan to start coding for next week

For our meeting for the following Monday Josh and Max will have our coding responsibilities for the week

Development will begin on that coming Monday

Week 4

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

9/15/25

In our meeting, Max assigned each of the tasks for each of the members of the team

We will be using the Tailwindcss for designing our specific pages.

Max wants us to use reusable components when coding so that we have a consistent style throughout the system

Week 5

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

9/22/25

We had our meeting at 9:00 PM. We did not assign any tasks as we are waiting for Max, Josh C., and Pablo to set up the backend database for our project. Max said that he would assign tasks for the week coming this Wednesday in class.

9/24/25

In class we went over wire-framing in software development for the lecture

Once Josh puts mock data into the database, then we can make queries to the database and remove the hard coded data from the frontend

We got the login and registrar functionality working. Meaning that each of the pages are tailored to the specific roles that we are tasked with

Josh recommended that we learn how to do SQL queries and API requests

The main functionality on the backend is completed

If we keep at the pace that we're going we can finish within 3 weeks

Max briefly discussed our presentation roles

For next week, Max wants to work on my page (Chef Page) to query data instead of having hard coded data. We plan on meeting via discord sometime next week.

Week 6

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

10/1/25

In class we went over how to test our software to increase its security from malicious entities who would want to break our code.

Week 7

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

10/8/25

Max informed us that he finished the chef page, the manager page, and the host page. The later two pages need some security fixing

The basic functionality for the pages is working though

The plan moving forward is to work on the frontend to make a more cohesive final product

Moving forward we will plan to meet and discuss how we want the management system to look and function

Week 8

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

10/15/25

Today in class we learned about how to do automated testing for our web application. The Specific applications “owasp Zap” and “SQL Map”

He also recommended Portswigger as a free means to learn about the security side of web-development

He told us that he will have a menu for us next week, as result of his computer dying

Week 9

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

10/22/2025

Today in class, Mr. Rogers displayed the menu that we will need to implement into our restaurant management system

Max created the functionality that makes a receipt. Allowing for full functionality for the website

The front end styling is something that we still need to work on

Week 10

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

10/29/2025

During class the lecture consisted of announcements. We have 4 weeks left to work on our project

Josh plans on doing an audit of each of the pages to make sure that everything is in order

Max made a tab menu that has multiple buttons that you can click through

The one thing that needs to be fixed is the payments on the system

For the functionality, needs to ensure that when someone leaves a table we clear all of their previous data, and then add new data for a new table

Bug:
The nav bar should render differently for users that aren't admins

On the Reports page, the reports aren't showing up when they should be

Our payments table is hardcoded

Week 11

11/5/2025

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

During class Mr. Rogers announced that we have 3 weeks before our presentation and project are due. That is not counting Thanksgiving break. For the next 2 weeks lectures are optional. Not the presentation day which is December 3d.

We are pretty much done

There is an issue with the API requests for the payments

The security tester should have notes on everything he found. The programmers should have notes

Could have a separate document for issues that we faced in the development process. I should title the document, Below is the section I removed from the final documentation document

Leon was able to set up the testing environment. He used Selenium through a Python file called “Salenium.py” and tested the login functionality

Issues That We Faced

On the development end of this project there were no major roadblocks that we ran into. For the members that did contribute, there weren’t any major issues that derailed the development process. With that being said we did run into bugs as we finished our project.

One such bug was found in the functionality of payments. According to Max, the payments table had hard coded data within it. [EXPAND UPON THIS MORE]

Another bug that we ran into was on the Reports Page. The reports aren’t showing up when they should be. [EXPAND UPON THIS MORE].

Outside of the coding aspect. We had members of our group that didn’t actually contribute to the software development process.

How We Fixed These Issues

Max addressed the functionality issue by…

The reports page issues were addressed by…

The payments issues was addressed

I should include:

1. How to set up the project
2. Dependencies needed… i.e. if a rando were to install our project what software would they need to make this project. The sub-bullets are things that you should include below
    a. What operating system is allied. Can it be on Linux, Mac, Windows
    b. Do we send documentation?
    c. Do we send someone from our company to install it or can the customer set it up themselves
3. Contingency Plans
    a. Include what to do if System shuts down
    b. Who do you call if there's a database issue

Week 12

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

11/11/25

Today Max informed us via Whatsapp that he finished the payments and reports sections.

He said that he believes that “clearing” of the tables needs to be done when payment is submitted. To further elaborate when one pays, the table should be automatically cleared in the system and “open.”

Max also said that we will need to fully test the application and make sure all features are properly implemented

11/12/2025

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

We did not have an in-person class, so we decided to have a meeting on discord at 5:30 PM

Someone needs to take a look at payments API calls.

There needs to be editing on the chef page. The chef should be submitting a completed order.

Little fixes that need to be made to the system to help increase the flow of the management system. Max left comments for stuff to be done in the code.

According to Max we are 98% of the way done with the system.

For the presentation, Max thinks that each of us should have a chance to speak, but we

Include meetings, who was there, what was discussed, what bugs we had, and how we addressed those

Week 13

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

11/17/2025

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

I had a meeting with Max on Discord. During that meeting, we worked on the documentation of code for the frontend and backend of our restaurant management system.

Max wants me to have a rough draft of our finalized version of our project documentation by this Sunday

Week 14

Meeting Attendants: Josh, Joshua, Max, Pablo, Leon, and Andrew

11/20/2025

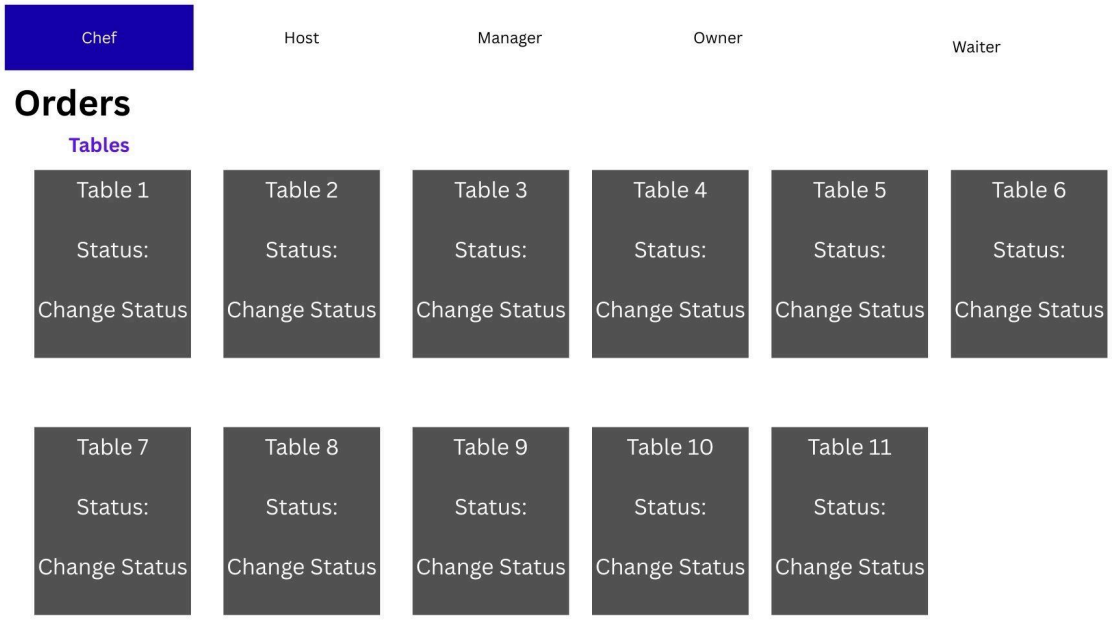
Our meeting was held on Discord voice call at 6:30 PM. We discussed the finishing touches we need to implement for our restaurant management system. Max tasked each member to test our respective pages.

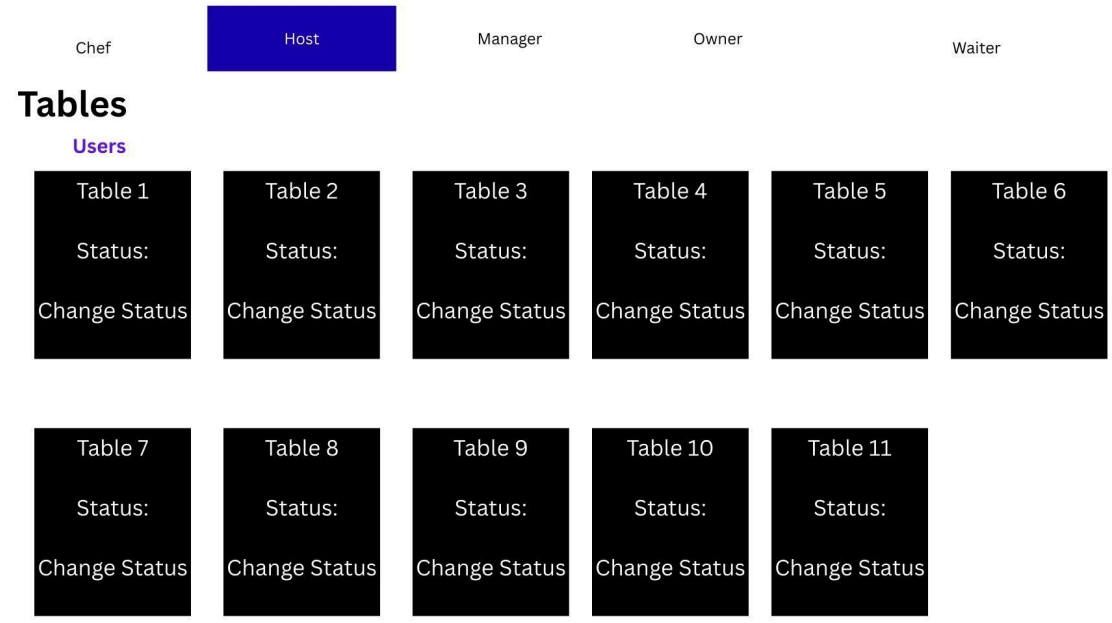
For the final documentation write up, Max wants me to incorporate the meeting notes into the final documentation document. After I do that, and make some final implementations, he and I will meet this Friday to upload the document as a pdf to our restaurant management github

11/21/2025

Max and I met via Discord voice chat at 12:00 PM. During this meeting, Max and I reviewedd our meeting notes and our software documentation document

# Page Mock Up Designs



Chef Host Manager **Owner** Waiter

# Welcome, Insert User

**Owner Dashboard**

## Workers

| | | |
|---|---|---|
| Name | Role | Remove |
| Name | Role | Remove |
| Name | Role | Remove |
| Name | Role | Remove |
| Name | Role | Remove |
| Name | Role | Remove |
| Name | Role | Remove |

Chef Host Manager Owner **Waiter**

# Waiter Dashboard

**Tables**

| | | | | | |
|---|---|---|---|---|---|
| Table 1<br>Status:<br>Change Status | Table 2<br>Status:<br>Change Status | Table 3<br>Status:<br>Change Status | Table 4<br>Status:<br>Change Status | Table 5<br>Status:<br>Change Status | Table 6<br>Status:<br>Change Status |
| Table 7<br>Status:<br>Change Status | Table 8<br>Status:<br>Change Status | Table 9<br>Status:<br>Change Status | Table 10<br>Status:<br>Change Status | Table 11<br>Status:<br>Change Status | |