Software Requirements Specification (SRS) Document:
Restaurant Ordering Web Application
Team Ctrl + Alt + Elite (Jared Scheurer, Jackson Grocki, Zak Knudson)
CSET 120 – Software Development
11/13/2025

SRS Document for Ambrosia

"The Food of Gods"

Table of Contents

1. Introduction

Purpose of the Document

This SRS outlines the features, requirements, and system behavior for a simplified restaurant ordering web app. It serves as a roadmap for development and keeps the team's vision and goals aligned.

Project Overview

The system has three types of users:

Guest: Can browse the menu, add items to the cart, and checkout immediately. Cannot schedule orders or view past orders. Ability to Log-In/Sign-Up.

Customer (signed-up): Can do everything a Guest can, plus schedule orders ahead of time, and view past orders.

Manager: Log in with hardcoded credentials to manage menu items and delete scheduled orders.

All data will be stored in JavaScript arrays/objects. No database or backend is used; this is a demo of a restaurant ordering platform. Guests can use the app without signing up, while Customers get added functionality like scheduling ahead and viewing order history.

Intended Audience

Instructor - will approve the SRS and grade the project.

Team members will use this as a development roadmap.

Any potential "investors."

Use Cases

1. A guest browses the menu and places an order.
2. A guest signs up and logs into a customer account.
3. A customer schedules an order ahead of time.
4. A customer views past orders.
5. The manager logs in and adds/removes menu items.
6. The manager deletes scheduled orders.
7. The system generates a receipt after checkout.

2. Project Scope

In-Scope Functionality

Guests can browse the menu, add/remove items in the cart, and check out.

Customers can sign up, log in, schedule orders, and view past orders.

The manager can log in, add/delete menu items, and delete scheduled orders.

Menu display with at least 20 items across categories.

Check out with cash or card and an optional tip.

Receipt generation showing items, total, customer name, and estimated prep time.

Out-of-Scope

No database or backend/server.

No real payment processing.

No permanent order history after refresh for Guests (Customer history stored temporarily in localStorage).

Goals & Objectives

Build a simple web app that feels like a real restaurant ordering system.

Make it easy to browse, order, and manage items for all users.

Allow Customers and Managers extra functionality compared to Guests.

Constraints

Must use HTML, CSS, and JavaScript.

Runs entirely in the browser.

Menu and users hardcoded (for Manager and default Customers).

Three user types: Guest, Customer, Manager.

3. Requirements

Functional Requirements

Guest Requirements

1. Browse the menu.
2. Select/unselect items.
3. Add/remove items from cart.
4. Check out with cash or card.
5. Receive receipt.

Customer Requirements (all Guest features +)

1. Sign up for an account.
2. Log in/log out.
3. Schedule order ahead.
4. View past orders.

Manager Requirements

1. Log in with hardcoded credentials.
2. Add menu items.
3. Remove menu items.
4. Remove scheduled orders.

Behavioral Requirements

Guests cannot schedule orders or view past orders.

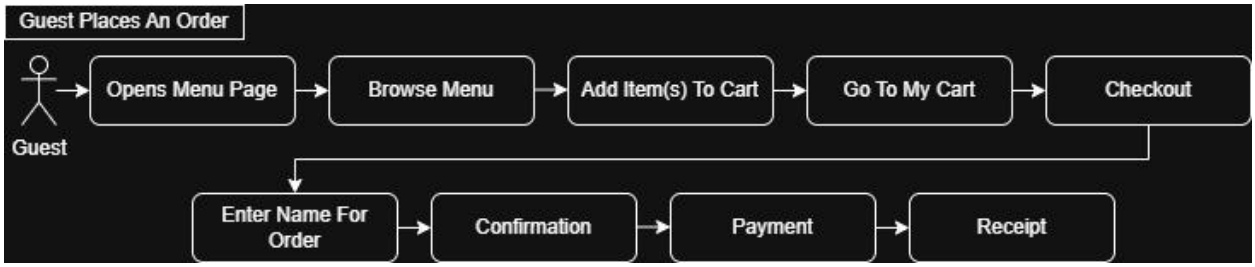
Prevent checkout with an empty cart.

Menu updates instantly after the manager edits.

Receipt calculations are accurate.

Use Case Scenarios

UC1 – Guest Places an Order:
Guest browses menu > select item(s) > my cart > checkout > guest info (name for order) > confirmation > payment > receipt.



Primary Actor: Guest

Preconditions: Homepage is open

Trigger: Guest navigates to Menu Page

Main Flow:

1. Open Menu Page
2. Browse Menu
3. Select a MenuItem to view
4. Add item to cart
5. Repeat 2-4 until finished with Order
6. Go to My Cart
7. Checkout
8. Enter name for the Order
9. Order and Name confirmation
10. Payment page displayed w/ payment options
11. Receipt Page displayed

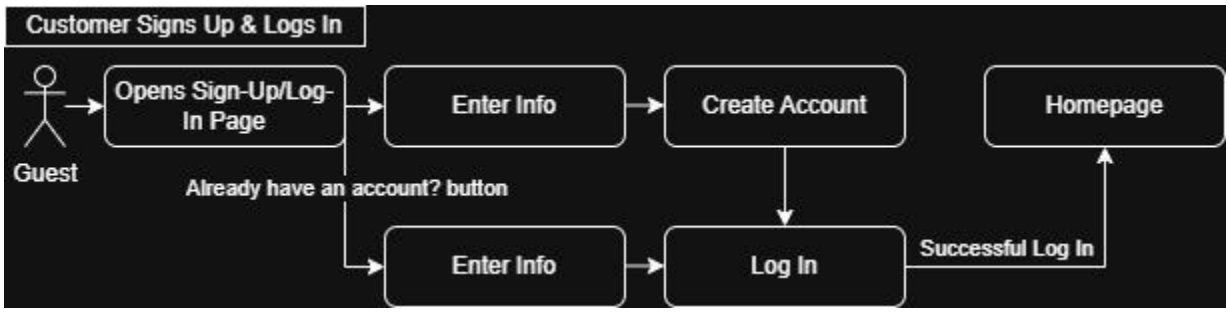
Exceptions:

- Guest opts not to place their Order
- Guest signs in to their Customer account

Postconditions: Receipt Page displayed and Cart is cleared

Special Requirements: Must not be logged in to Customer account

<u>UC2 – Guest Signs Up & Logs In</u>:
Open sign-up > enter info > create account > log in > access full features.
account already? button > enter info > log in > access full features.



Primary Actor: Guest

Preconditions: Homepage is open and not signed in

Trigger: Guest

Main Flow:

1. Open Sign-Up Page
2. Enter desired name, username, and password
3. Create Account
4. System validates information
5. Account created successfully
6. Redirected to Log-In Page
7. Guest logs into their Customer account
8. Customer redirects back to Homepage

Alternate Flow (Account already exists):

1. Open Sign-Up Page
2. Click the “Already have an Account?” button
3. Redirected to Log-In Page
4. Guest logs into their Customer account
5. Customer redirected back to the Homepage

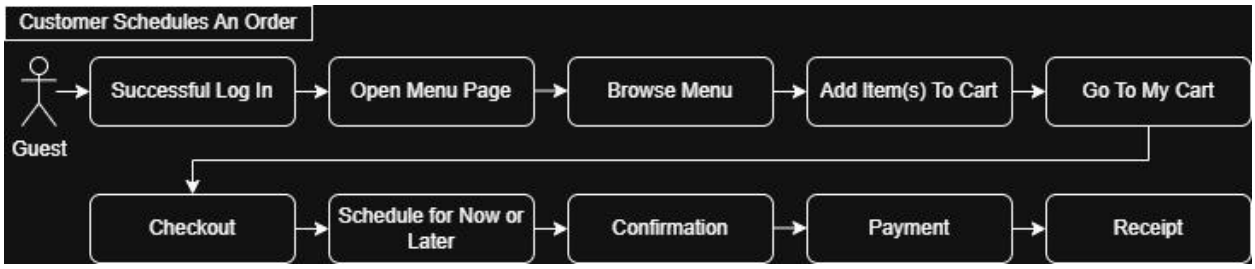
Exceptions:

- Account already exists (Signing up) > Prompts Guest to go to Log-In Page
- Bad password (falls outside allowable passwords) won’t allow account Sign-Up
- Username already in use won’t allow account Sign-Up

Postconditions: Account is logged in and Homepage is displayed

Special Requirements: Username must be unique and password must meet minimum requirements (at least 8 characters and no more than 15)

<u>UC3 – Customer Schedules an Order</u>:
Log in > open menu page > browse menu > select item(s) > my cart > checkout > choose time (now/later) > confirmation > payment > receipt.



Primary Actor: Customer

Preconditions: The Customer has added items to their cart already and is ready to schedule a time

Trigger: Customer pays and confirms their order

Main Flow:

1. Log in
2. Open menu page
3. Select item
4. My cart
5. Checkout
6. Choose time (now/later)
7. Confirmation and payment

Exceptions:

- Customer ops to end the session

Postconditions: The order will be sent to the restaurant at the specified time selected by the customer

Special Requirements: They must be signed in as a customer; as a guest will not have this feature available to them.

<u>UC4 – Customer Views Past Orders</u>:
Log in > my cart > open past orders > system displays order history > view order details.

Primary Actor: Customer

Preconditions: Customer is logged in with at least one previous Order

Trigger: Customer selects “Past Orders” from My Cart Page

Main Flow:

1. Open My Cart Page
2. Select the “Past Orders” option
3. Past Orders Page displayed

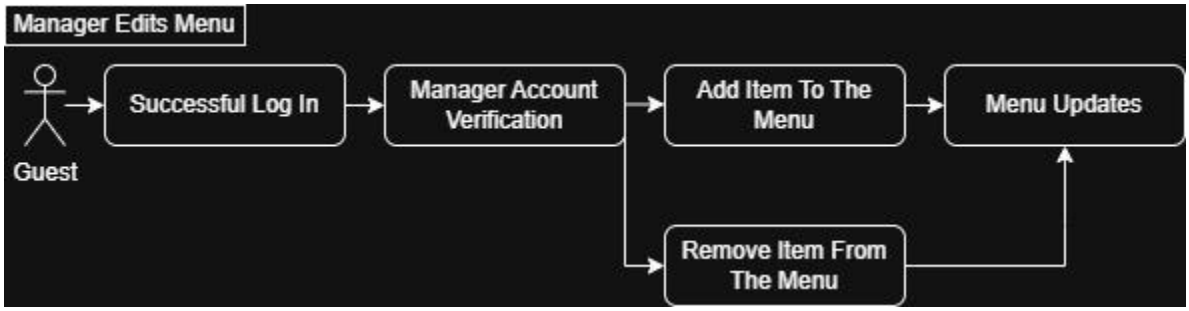
Exceptions:

- No past orders on Customer’s account > “No Previous Orders” message displayed instead

Postconditions: Customer views list and details of Past Orders

Special Requirements: Must be logged into the Customer Account

<u>UC5 – Manager Edits Menu</u>:
Log in > verification > add/remove items > menu updates live.



Primary Actor: Manager

Preconditions: Manager logs in with appropriately hardcoded information

Trigger: Manager opens Menu ADMIN Page

Main Flow:

1. Open Menu ADMIN Page
2. Select Add Item
3. Enter Item information
4. Confirm information
5. Menu updates

Alternate Flow:

1. Open Menu ADMIN Page
2. Select Remove Item
3. Select appropriate ItemID
4. Confirm deletion
5. Menu updates

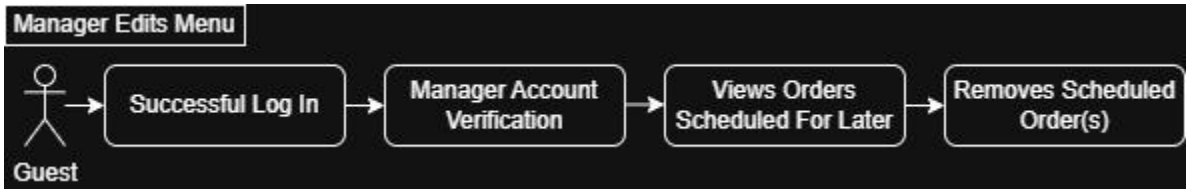
Exceptions:

- Adding an ItemID that already exists prompts

Postconditions: Menu updated for Customers and Guests

Special Requirements: Must be logged into Manager / ADMIN account

<u>UC6 – Manager Removes Scheduled Orders</u>:
Log in > scheduled orders list > select order > delete > list updates.



Primary Actor: Manager

Preconditions: the manager must be logged in as a manager

Trigger: Manager selects an item to remove from the menu

Main Flow:

1. Log in
2. Scheduled orders list
3. Select order
4. Delete
5. List updates

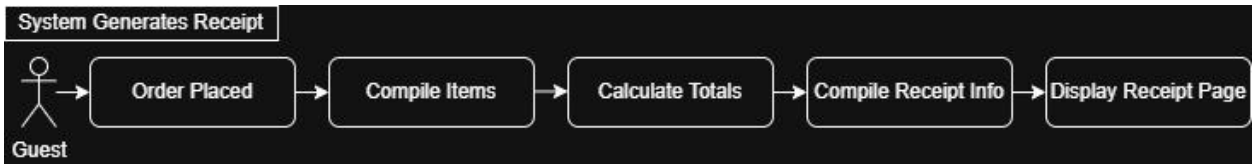
Exceptions:

- No Scheduled Orders > “No Scheduled Orders” displayed

Postconditions: The Scheduled Orders list will be updated

Special Requirements: Must be logged in as Manager

UC7 – System Generates Receipt:
Order placed > compile items > calculate totals > display receipt with name, items, total, and date.



Primary Actor: System

Preconditions: Order successfully placed

Trigger: Customer completes checkout or scheduled ordered successfully placed

Main Flow:

1. Compile ordered items, quantities, and prices
2. Calculate subtotal, tax, tip, and total
3. Display total amount and payment option
4. Generate receipt with customer name, items, total, date/time
5. Receipt Page displayed

Exceptions:

- Error in calculation > system displays error and prompts a new attempt
- Receipt page fails to display > prompt user to refresh page

Postconditions: Receipt successfully generated and displayed

Special Requirements: Accurate calculations and clear / readable display

Non-Functional Requirements

Performance: Pages should load instantly and the menu should update immediately.

Usability: Clean UI, simple navigation, clear buttons.

Reliability: Hardcoded data consistent; totals accurate.

4. System Design

Entities

Guest –

- attributes: cart, no userID
- methods: viewMenu(), viewDetails(itemID), addToCart(item), removeFromCart(item), checkout()

Customer –

- attributes: userID, name, username, password, cart, orderHistory[]
- methods: viewMenu(), viewDetails(itemID), addToCart(item), removeFromCart(item), checkout(), scheduleOrder(), viewPastOrders(), logOut()

Manager –

- attributes: hardcoded username & password
- methods: addMenuItem(itemID), deleteMenuItem(itemID), removeScheduledOrder(orderID), viewMenu()

MenuItem –

- attributes: itemID, name, price, category
- methods: viewDetails()

Order –

- attributes: orderID, userID, itemList[], total, scheduledTime
- methods: calculateTotal(), printReceipt(), addScheduledOrder(), removeScheduledOrder(orderID)

System –

- methods: signUp(userInfo), logIn(account)

Relationships

Manager > MenuItem
Manager can add/delete many MenuItems

Manager > Order
Manager can cancel scheduled Orders

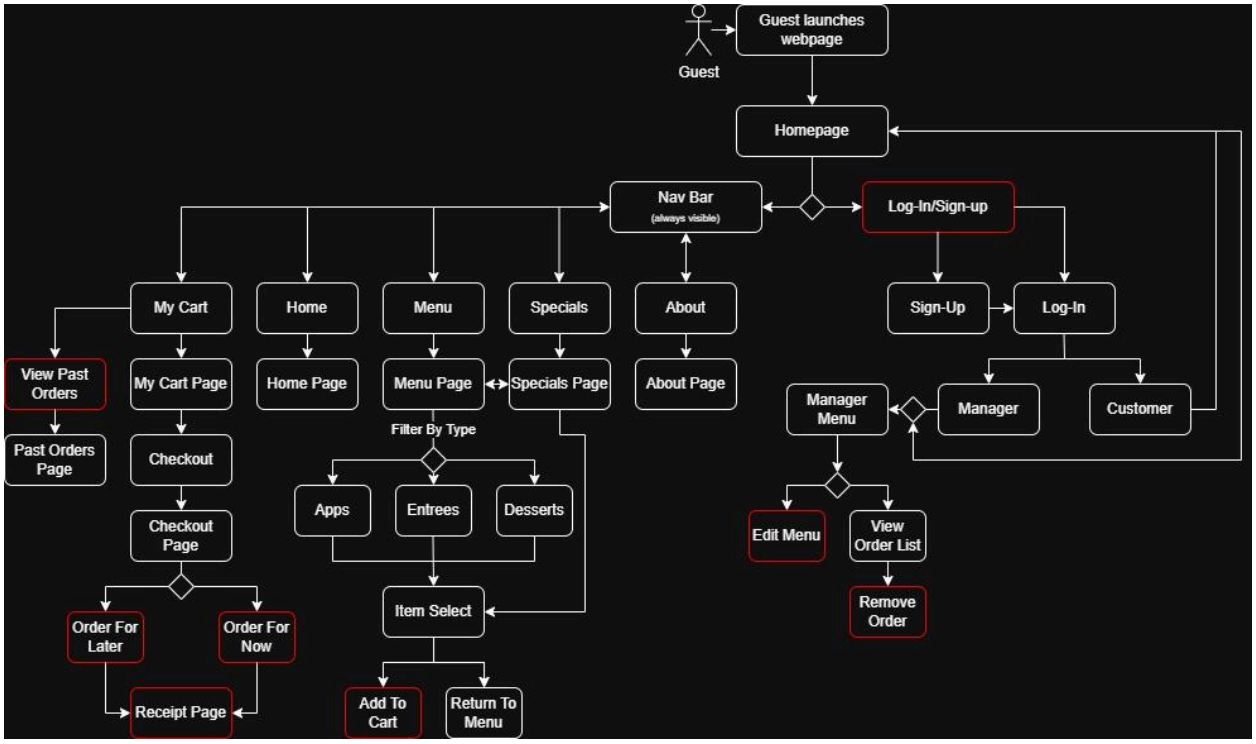
Customer > Order
Customer can place many Orders over time
Each Order belongs to an individual Customer

Order > MenuItem
Each Order consists of one or more MenuItem(s)

Guest > Order
Guest can make an Order without signing up
Each Guest Order is linked to a temporary session

System > Customer / Guest / Manager
System handles signUp() and logIn() for Customers / Guests
System validates Manager (admin) credentials for admin log in

UML Class Diagram - Ambrosia Webpage Workflow



5. System Architecture

Architecture Overview

Browser-based.

Data stored in JS arrays/objects.

Pages linked via navigation.

localStorage optionally used for Guest cart/order.

Core Components

Menu Display Module

Cart Management Module

Ordering Modules

Checkout Module

Receipt Module

Manager / ADMIN Panel

6. Testing & Validation

Test Cases

Guests can browse, add/remove items, checkout.

Customers can sign up, log in, schedule orders, view past orders.

Manager login rejects wrong credentials.

The manager can add/delete menu items and scheduled orders.

Receipts show correct totals.

Acceptance Criteria

All features work smoothly.

UI is intuitive.

Orders, scheduled orders, and receipts display correctly.

7. Best Practices

Keep requirements simple.

Prioritize core features.

Update SRS with major changes.

Track tasks in Trello.

Use GitHub for version control.

8. Conclusion & Approval

Summary

This SRS defines the restaurant ordering app with Guests, Customers, and Manager roles. It ensures clear functionality and alignment for development.

Sign-Off

Instructor: _____

Team Members:_____