Team Y

DishIQ
Software Requirements Specification
For DishIQ

Version 1.0

Revision History

Date	Version	Description	Author
23/10/2025	1.0	Introduction and Supporting Information	Krista U. Singh
23/10/2025	1.0	Use-Case Model Survey	Lin Finnegan
23/10/2025	1.0	Assumptions and Dependencies	Marina Morcos
23/10/2025	1.0	Use-Case Reports	Brianna Hayles
23/10/2025	1.0	Supplementary Requirements	Jing Han Lin

Table of Contents

Software Requirements Specification

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to describe the goals, features, and requirements behind DishIQ — an AI-enabled restaurant order and delivery system. The main idea behind DishIQ is to make the restaurant experience smarter and more personalized for both customers and restaurant staff.

Through this system, customers can browse the menus from various restaurants, place and track orders, leave ratings, and chat with an AI assistant that helps answer questions about dishes, deliveries, and even the restaurant itself. Simultaneously, employees such as chefs, delivery drivers, and restaurant managers can handle their daily tasks more efficiently. For example, chefs can manage their own menus, delivery drivers can bid for deliveries, and the manager can monitor performance, handle customer complaints, and view customer satisfaction.

This document outlines how DishIQ's system will function, what each user group can do, and what technical and AI components will be integrated to make DishIQ seamless. The goal of this SRS is to serve as a roadmap for the development process, making sure the design, code, and testing stays consistent and aligns with the restaurants' needs.

1.2 Scope

DishIQ is a comprehensive platform that combines restaurant management, customer engagement, and AI-driven personalization. It focuses on improving both customer satisfaction and staff efficiency by using intelligent automation and data-based decision-making.

There are three main types of users within the system:

1. **Employees** – This group includes chefs, delivery staff, and managers.
    - Chefs can independently create and update menus, view feedback, and monitor how well their menu items perform based on customer ratings and complaints
    - Delivery staff can bid for orders, track deliveries, and view customer ratings
    - Managers oversee registration, handles finances, manages complaints and compliments, and makes staffing decisions like hiring or promotions

2. **Customers** – Registered users who can order food, leave reviews, and participate in discussions.
    - Customers can earn VIP status after spending over $100 or completing three successful orders without complaints.
    - VIPs receive exclusive discounts, free deliveries, and early access to new menu items. Their feedback also carries more weight in the system as well.

3. **Visitors** – Individuals can browse menus, interact with the AI assistant, and apply for registration as customers.

In addition to these core functionalities, DishIQ also introduces an AI Image-Based Dish Search feature. This creative component allows users to upload a photo of any dish and receive suggestions of visually similar menu items from the restaurant. By combining image recognition and AI, the system provides a more interactive and intuitive way to discover food based on personal cravings.

Overall, DishIQ's system scope covers restaurant operations, customer interactions, reputation management, financial processes, and AI-based personalization within a single integrated platform.

1.3 Definitions, Acronyms, and Abbreviations

This section defines key terms and roles used throughout the DishIQ Software Requirements Specification. These definitions ensure consistent understanding of the system's structure, functions, and AI features.

1. **Manager** – The employee responsible for overseeing registrations, finances, staff performance, and complaint resolution within the system.

2. **Chef** – A restaurant employee who creates and manages menu items, receives feedback on dishes, and can earn rewards or penalties based on customer ratings.

3. **Delivery Person** – An employee who bids on available deliveries, transports orders, and is evaluated by customers on delivery quality and service.

4. **Customer** – A registered user who can place orders, rate dishes and deliveries, participate in discussions, and file complaints or compliments.

5. **Visitor** – A non-registered user who can browse menus, ask questions through the AI chat, and apply for registration.

6. **VIP Customer** – "Very Important Person"; A customer who attains VIP status after completing three successful orders or spending over $100. VIPs receive discounts, free deliveries, and priority handling of feedback.

7. **Knowledge Base** – A locally stored collection of information about the restaurant, menu, and users that the AI assistant references to answer questions before consulting an external model.

8. **Complaint/Compliment System** – A reputation-tracking feature where customers, delivery staff, and chefs can submit and respond to feedback managed by the restaurant's managers.

9. **Warning System** – A disciplinary mechanism that issues warnings to users or staff for rule violations; repeated warnings may lead to demotion or account removal.

10. **Finance Module** – The component that manages customer deposits, order payments, and automated checks for sufficient account balance.

11. **AI Assistant** – The chat interface that answers user questions using information from the local knowledge base and, if needed, a language model.

12. **LLM** – "Large Language Model"; a text-based AI used by the chat assistant to generate or refine responses when local data is insufficient.

13. **Image-Based Dish Search** – The creative AI feature that analyzes an uploaded food photo and recommends visually similar dishes from the restaurant's menu.

14. **CLIP** – "Contrastive Language-Image Pretraining"; an open-source AI model used to compare uploaded food images with menu item photos.

15. **YOLO** – "You Only Look Once"; an object detection model used for food recognition.

16. **VIP Promotion Rules** – The conditions that determine when a customer becomes a VIP and how their privileges and warning limits differ from regular customers.

17. **Bid System** – A competitive process where delivery personnel propose delivery prices, and the manager assigns the order based on the lowest or most justified bid.

1.4 References

- Zhang, Y., Chen, L., & Wang, S. (2021). *AI-based Restaurant Recommendation Systems: A Survey.* Journal of Intelligent Information Systems, 57(3), 475-497.
- Luo, Z., & Xu, H. (2022). *The Impact of AI Chatbots on Customer Experience in Online Food Ordering Platforms.* International Journal of Human-Computer Studies, 163, 102798.
- McDowell, E. (2023). *AI in Hospitality: How Automation and Chatbots Are Transforming Restaurants.* Hospitality Technology Magazine.
- Hugging Face CLIP and YOLO model documentation.
- Python, Flask, and Streamlit official documentation.
- OpenCV, Pillow, and NumPy for image processing.
- SQLite and MySQL documentation for database integration

1.5 Overview

The remainder of this SRS document provides a detailed description of the DishIQ system and its requirements.

Section 2 presents the overall system structure, user roles, and the relationships between different components. Section 3 defines the system's specific requirements, including functional and supplementary details. Lastly, Section 4 provides supporting materials, such as diagrams, mock-ups, and references that guide implementation and future updates.

Together, these sections define the complete software framework of DishIQ, clarifying how it functions and the features that make it an intelligent and efficient restaurant management platform.

2. Overall Description

2.1 Use-Case Model Survey

The Use-Case Model below illustrates the primary interactions between DishIQ's users and system functionalities.

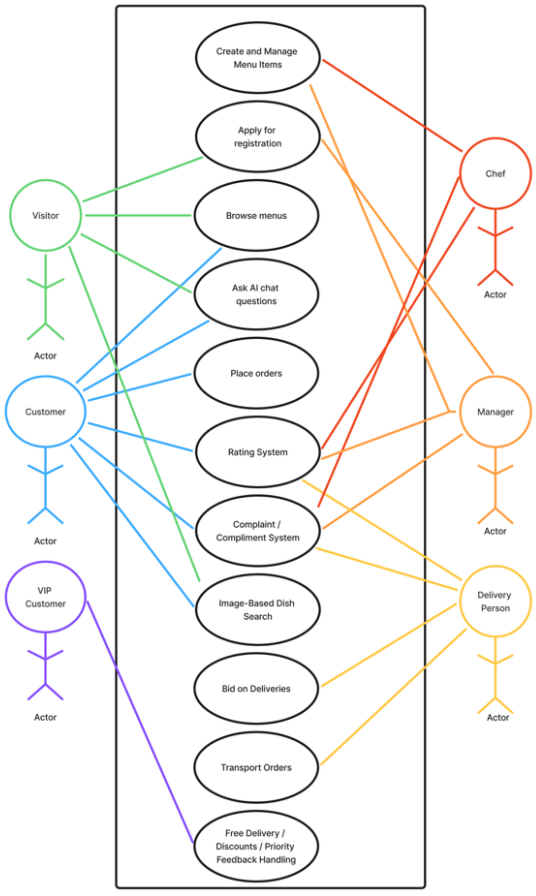

Figure A-1

Actors:

- **Visitor:** Can browse menus, ask questions through the AI chat, and apply for registration.
- **Customer:** Can log in, place orders, rate dishes and deliveries, submit compliments or complaints, and use the Image-Based Dish Search feature.
- **VIP Customer:** Has all Customer privileges plus additional benefits — attains VIP status after completing three successful orders or spending over $100 without complaints; receives discounts, free deliveries, and higher feedback weight.
- **Chef:** Can create and manage menu items, view customer feedback, and respond to complaints or compliments.
- **Delivery Person:** Can bid on deliveries, transport orders, and view delivery ratings or complaints.
- **Manager:** Handles user registrations, manages complaints and compliments, monitors ratings, and oversees chef and delivery staff performance.

System Overview:
DishIQ supports AI-enhanced restaurant ordering and management. The Use Case Diagram (shown in Figure 2-1) shows how these actors interact with major system functions such as browsing menus, placing orders, managing menus, bidding on deliveries, using the AI chatbot, and searching for dishes by image.

2.2 Assumptions and Dependencies

These assumptions and dependencies define the technical foundation required for DishIQ's successful operation.

1. **Chatbot and Knowledge Base Dependency**
   One of the features of DishIQ is the chatbot, which allows customers to ask questions relating to the restaurant. To allow for fast and accurate responses, a local knowledge base will store structured FAQs and restaurant information for answering common questions. However, if the question is outside the scope of the local knowledge base, the system will need to use a free or open-source LLM for chatbot fallback.

2. **Database Reliance**
   DishIQ depends on a relational database for storing all menu data, customer information, staff profiles, and order records. This database will handle data retrieval, updates, and daily transactions, ensuring that menu items, prices, order statuses, and user accounts are always accurate and accessible. The system relies on the database being secure, reliable, connected, and consistently available.

3. **Internal Delivery Operations**
   DishIQ will support food delivery using internal delivery staff employed by the restaurant. Since no third-party delivery services will be integrated, it is assumed that the restaurant will manage its own scheduling and availability. This reduces the complexity of external dependencies, but places full responsibility for logistics on the restaurant.

4. **GUI and Image Support**
   The system will feature a GUI that can be developed as either a desktop or web-based application. The GUI must support images so that customers can view pictures of each dish before ordering. This improves the user experience and assumes stable image storage and display capabilities within the system.

5. **Authentication Requirements**
   To ensure secure access, DishIQ will use password-based authentication for both customers and staff. The system assumes that proper encryption or hashing techniques will be in place, so user credentials are safely stored and protected from unauthorized access.

6. **Chatbot Rating System**
   Since the chatbot is deeply incorporated into the platform, DishIQ will include a rating system allowing users to score the accuracy and helpfulness of LLM-generated responses. These ratings will be used to improve the quality of future chatbot answers by refining the knowledge base and identifying areas where additional information is needed.

7. **Creative Feature Dependency**
   The creative feature allows users to submit images of food, and the system suggests food options that are visually similar. External APIs, libraries, or additional models will be required to support this functionality.

3. Specific Requirements

3.1 Use-Case Reports

The following use cases (UC-01 through UC-12) describe interactions between users and the **DishIQ Web System**. Unless stated otherwise, each use case operates at the **User-Goal level** within the system scope.

UC-01: Visitor Browsing

Primary Actor: Website Visitor

Goal: To allow visitors to browse a restaurant menu and ask the AI chatbot questions without registering for an account.

Preconditions:

- The system is accessible online
- Menu data is available
- Chatbot is up and running

Main Success Scenario:

1. Visitor opens the DishIQ homepage
2. System loads displaying restaurants and a search bar
3. Visitor selects a restaurant or searches for a specific restaurant and/or dish
4. Visitor views menu (dish descriptions, prices, and ratings)
5. Visitor can ask questions in the chatbot
6. Chatbot answers using knowledge from the base or LLM fallback
7. Visitor continues browsing or exits

Postconditions:

- Visitor can leave the platform or proceed to registration

Alternative Flows:

- 3a. Menu fails to load ---> System displays an error message
- 6a. Chatbot cannot answer ---> System suggests contacting support

UC-02: Customer Registration

Primary Actor: Website Visitor

Goal: To create a customer account and gain access to full ordering features.

Main Success Scenario:

1. Visitor clicks “Register”
2. System displays registration form
3. Visitor enters relevant information (name, email, password, and payment details)
4. System validates input and stores the new customer’s information in the record
5. System sends confirmation message
6. Customer can successfully log in

Postconditions: New customer account is created

Alternative Flows:

- 4a. Invalid Information ---> System prompts for connection
- 5a. An account has already been made with said email ---> System displays “Account already registered”

UC-03: Place Order

Goal: To select dishes, submit payment, and place an order.

Main Success Scenario:

1. Customer logs in

2. System displays menu
3. Customer adds menu items to their cart
4. Customer reviews and confirms their order
5. System verifies necessary funds
6. System sends the order to the kitchen
7. System provides customer with an estimated delivery time
8. System checks deposit balance via Finance Module

Postconditions: Order is recorded and waiting to be prepared

Alternative Flows:

- 5a. Insufficient balance ---> Prompt to add funds
- 6a. Kitchen offline ---> System saves the order in the queue to be submitted later

UC-04: Kitchen Receives Order

Primary Actor: Kitchen/Chef

Goal: To receive, prepare, and update the status of the order

Main Success Scenario:

1. Kitchen receives a notification of the order
2. System displays dish details and notes made by customer
3. Kitchen sets the order status to “In Progress”
4. Kitchen prepares the dish
5. Kitchen changes the order status to “Complete”
6. System notifies delivery staff

Postconditions: The order is ready to be picked up by the delivery person

Alternative Flows:

- 2a. Ingredient missing ---> Chef flags order to manager for resolution

UC-05: Delivery Person Receives Order

Primary Actor: Delivery Person

Goal: To pick up the order from the restaurant and deliver it to the customer

Main Success Scenario:

1. Delivery person views available orders
2. System assigns the order to the delivery person based on bids or availability
3. Delivery person sets the status of the delivery to “In Progress”
4. When the order is delivered, the status is updated to “Delivered”
5. Customer receives confirmation and provides rating

Postconditions: Delivery marked complete, and rating recorded.

Alternative Flows:

- 3a. Delivery delay ---> Customer receives a notification

UC-06: Customer Rating and Feedback

Primary Actor: Customer

Goal: To rate their food and delivery service after receiving their order

Main Success Scenario:

1. Customer logs in after delivery
2. System shows orders awaiting feedback
3. Customer rates food
4. Customer rates delivery
5. Customer adds any positive or negative comments
6. System stores feedback and notifies manager

Postconditions: Feedback is saved for manager of restaurant to review

Alternative Flows:

- 3a. Customer skips feedback ---> System reminds them later

UC-07: Reputation Handling

Primary Actor: Manager

Goal: To process customer reviews

Main Success Scenario:

1. Manager logs into admin dashboard
2. Systems lists recent complaints and compliments
3. Manager reviews details
4. Manager issues warnings or commendation
5. System updates staff reputation record
6. If a complaint is upheld, a warning is issued; otherwise, it is dismissed.

Postconditions: Reputation database updated, and user notified.

Alternative Flows:

- 4a. Complaint invalid ---> Marked as dismissed

UC-08: HR Actions

Primary Actor: Manager

Goal: To promote, demote, or terminate staff based on performance.

Main Success Scenario:

1. Manager opens staff performance dashboard
2. System displays warning counts and ratings
3. Manager selects staff member to either promote, demote, or fire
4. System updates the status of the employee
5. Affected staff member receives notification

Postconditions: HR records updated and visible in manager reports.

UC-09: Deposit and Finance Management

Primary Actor: Customer

Goal: To verify sufficient funds before orders are processed.

Main Success Scenario:

1. Customer initiates payment for an order
2. System checks deposit balance
3. If sufficient, payment is processed
4. System deducts amount and logs transaction

Postconditions: Order and financial record updated

Alternative Flows:

- 2a. Insufficient funds --> Prompts the customer to add funds.

UC-10: Account Closure

Primary Actor: Manager

Goal: To close a customer or employee account

Main Success Scenario:

1. Manager searches for the account to close
2. System flags any outstanding balances or orders
3. Manager confirms the closure of the account
4. System clears deposits and blacklists user

Postconditions: Account removed from active records.

Alternative Flows:

- 2a. Pending order ---> System prevents closure until order is completed

UC-11: AI Chatbot Interaction

Primary Actor: Website Visitor, Customer, VIP Customer

Goal: To interact with the AI Chatbot for assistance

Main Success Scenario:

1. User opens chatbot interface
2. User asks a question
3. System searches local knowledge base
4. Chatbot answers questions if relevant data found
5. If the answer is not found, the system queries the external LLM
6. Answer is given to user

Postconditions: User receives relevant information

Alternative Flows:

- 5a. LLM unavailable ---> System falls back to the knowledge base---> System returns a fallback message.

UC-12: Creative Feature

Primary Actor: Customer, Visitor

Goal: To allow users to upload a photo of food and receive visually similar menu suggestions.

Main Success Scenario:

- User opens the image search tool.
- User uploads or captures an image of a dish.
- System analyzes the image using CLIP and YOLO models.
- System compares the uploaded image with menu images.
- System displays similar menu items with names, ratings, and prices.
- User can click a suggested dish to view details or order it.

Postconditions:

- Matching dishes are displayed.
- User interaction is logged for personalization.

Alternative Flows:

- 3a. Image upload fails → System prompts user to retry.
- 4a. No visual matches found → System displays “No similar dishes found” message.

3.2 Supplementary Requirements

The following supplementary requirements describes interactions between users and the DishIQ Web System. Unless stated otherwise, each use case operates at the User-Goal level within the system's scope.

SR-01: AI responsiveness

Primary concern: System responsiveness

Goal: To ensure the chatbot is providing smooth and timely response to all users

Preconditions:

- The system server is online
- Network connection is stable

Main Success Scenario:

- The Chatbot provides a response within 3 seconds
- If chatbot response exceeds 3 seconds, display a “processing, please wait” message
- If chatbot cannot respond, the system should suggest contacting support or providing fallback response from local knowledge base (UC-01 alternativeernative flow 6a, UC-11 alternative flow 5a) and notify the user “processing error, please try again later”
- Users interact with chatbot without delays
- Users are informed if chatbot is temporarily unavailable

Postcondition: Users can interact with the chatbot and receive frequent and fast responses

SR-02: System Reliability

Primary concern: Service availability

Goal: To maintain stable operation and minimize downtime for users

Preconditions: Core systems service must be running

Main Success Scenario:

- The system runs continuously during user active hours
- Orders, chat, and menu requests are be processed successfully
- Scheduled maintenance does not disrupt functionality
- If menu fails to load, system retries automatically and displays an error message if still unavailable (UC-01 alternative flow 3a)
- If kitchen is offline, orders are queued and processed after the system recovers (UC-03 alternative flow 6a)

Postcondition: System achieves uptime and recovers efficiently through minor failure

SR-03: Data Security and Input Validation

Primary concern: Protection of user data and transactional data

Goal: To ensure that all customers, restaurants, and transactional data are securely stored and that no data is leaked

Preconditions: User account and payment method are valid

Main Success Scenario:

- Input validation prevents invalid data and duplicate emails from being made. System should display “Account already registered” (UC-02 alternative flow 4a and alternative flow 5a)
- The system encrypts and safely stores all user and transactional data
- Payment transaction is processed securely through a trusted gateway
- Access to sensitive data is restricted for all users, except for the manager

Postcondition: All sensitive data is safely stored and processed, and is unable to be accessed by unauthorized users

SR-04: Data Scalability

Primary concern: Load management and data growth

Goal: To maintain performance as user data and traffic increases

Preconditions: System has scalable resources for database and servers

Main Success Scenario:

- System must handle high load without menu or chatbot failures. If menu fails, display error message (UC-01 alternative flow 3a)
- All system functions like chatbot, orders, and menus function efficiently
- New restaurants and menus can be added without causing issues

Postcondition: System remains consistent under growing user data

SR-05: Usability and Accessibility

Primary concern: User experience and accessibility

Goal: To ensure that the system is easy to navigate and is accessible to all users

Preconditions: User interface is operational

Main Success Scenario:

- Website interface is easy to navigate
- All buttons are responsive
- Users receive clear error messages if menu fails, chatbot cannot answer, and if system is down (UC-01 alternative flow 3a and UC-11 alternative flow 5a)

Postcondition: User can navigate the system easily and be informed of errors

SR-06: AI Chatbot Fallback

Primary concern: AI availability and fallback handling

Goal: To ensure chatbot can provide answer even if the main LLM is unavailable

Preconditions: Chatbot interface is available, and knowledge base is updated

Main Success Scenario:

- User opens chatbot interface and can ask questions
- System searches local knowledge base to find answers
- If no local knowledge is found, system queries external LLM (UC-11 alternative flow 5a)
- If LLM is unavailable, system falls back to the knowledge base, if neither are available system will let users know to contact support or try again later (UC-11 alternative flow 5a)
- Users are notified of any delays or temporary unavailability of the chatbot

Postcondition: Users always receives a response or guidance on how to proceed if an issue arises

SR-07: Payment and Order Continuity

Primary concern: Financial validation and order processing reliability

Goal: To ensure orders are only processed if sufficient funds are available and that queued orders are managed correctly

Preconditions: Users have linked payment methods and system must have access to the account balance or deposit information (if any)

Main Success Scenario:

- System must verify funds are sufficient before order submission
- If funds are insufficient, server prompts the user to add money before proceeding or a prompt telling the user insufficient balance (UC-03 alternative flow 5a and UC-09 alternative flow 2a)
- If kitchen is offline, order is queued and automatically submitted when the kitchen is back online (UC-03 alternative flow 6a)
- Users are notified of any payment delays or issues.

Postcondition: Orders are recorded, and payments are secure. The user will be informed of any issues if arises

SR-08: Kitchen and Inventory Management

Primary concern: Order fulfillment reliability

Goal: To ensure kitchen staff receive the correct information and ingredient shortages are managed

Precondition: Kitchen interface is online, and inventory database is updated

Main Success Scenario:

- System sends orders with notes to kitchen
- Kitchen should update order status as “In progress” or “Complete”
- If ingredients are missing, system should flag the order to manager and staff (UC-04 alternative flow 2a)
- Orders are held until resolution

Postcondition: Orders and inventory remain consistent after update

SR-09: Delivery monitoring and notifications

Primary concern: Timely delivery and customer communication

Goal: To ensure that all customers are informed about delivery status and delays

Preconditions: Delivery personnel have access to assigned orders (after bidding), tracking must be available

Main success scenario:

- System should assign the delivery and update status to “In Progress” once the order is picked up
- System should update status to “Delivered” and notify the customers upon delivery
- If delivery is delayed, the system should automatically notify the user about it and display the new estimated time (UC-05 alternative flow 3a)

Postcondition: Delivery is tracked and user is informed with any changes that may have occurred

SR-10: Feedback and Reputation Management

Primary concern: Collecting and processing feedback along with managing complaints

Goal: To ensure that all customer ratings and complaints are captured and processed accordingly

Preconditions: Use users has completed an order and the feedback interface is opened

Main success scenario:

- System should prompt customers to rate food and delivery after order completion
- If customers skip feedback, system sends reminder notifications later (UC-06 alternative flow 3a)
- System will then process the feedback and updates manager dashboard
- Invalid complaints are flagged, marked as dismissed, and logged for reviewing (UC-07 alternative flow 4a)

Postcondition: Feedback and complaints are recorded accurately, and system should alert false complaints

SR-11: Account management and closure

Primary concern: Account integrity and compliance

Goal: To ensure accounts can be safely closed while respecting pending obligations

Preconditions: The account should exist in the system, and no outages should be affecting account operations

Main success scenario:

- The system should check for pending orders and outstanding balance before closing the account. If detected, the system should prevent the account from being closed (UC-10 alternative flow 2a)
- If no pending issues arise, account is closed, deposit is cleared, and user should be blacklisted from registering if required

Postcondition: Accounts closure should occur when all conditions are met which maintains operation and financial integrity

SR-12: Image Recognition Reliability

Primary concern: Visual upload and processing stability

Goal: To ensure that the image upload and visual recognition features function reliably

Preconditions: User device should support image upload and system image recognition should be active

Main success scenario

- User uploads a dish image
- System should successfully process the image and returns the results within 3 seconds
- System should notify the user if upload fails and should prompt the user to retry (UC-12 alternative flow 3a)
- System should display a “No similar dishes found” message to the user if no visual matches are found (UC-12 alternative flow 4a)

Postcondition: Users should receive consistent and informative feedback whether the image failed or succeeded to upload

4. Supporting Information

Index

Section	Title	Description
1.0	Introduction	Overview of DishIQ and purpose of this document
2.0	Use-Case Model Survey	Overview of key actors, system interactions, and dependencies (figure A-1)
3.0	Specific Requirements	Functional requirements defined through detailed use-case reports
3.1	Use-Case Reports	Detailed descriptions of system use cases (UC-01 through UC-12)
3.2	Supplementary Requirements	Non-functional requirements and additional system constraints (SR-01 through SR-12)
4.0	Supporting Information	Index and Appendices

Appendices:

Appendix A: Use Case Diagram

Figure A-1 illustrates the interactions between DishIQ's actors and system functionalities.

Ordinary users (Visitors and Customers) are positioned on the left, while privileged users (Chefs, Managers, and Delivery Personnel) are on the right.

Each oval represents a key system feature such as menu browsing, order placement, delivery bidding, reputation management, and AI-powered image search.