Qassim University
جامعة القصيم
College Of Computer | كلية الحاسب

QUICKMART

WAREHOUSE-TO-CUSTOMER GROCERY DELIVERY APPLICATION

Subject: CS383

Instructor: Alaa Alsehail

prepared by:

Lama Alghofaili (group leader)

And another 8 group members

Table Of Contents

ABSTRACT

QuickMart is a warehouse - to - customer grocery delivery application designed to simplify the shopping process by allowing users to order daily necessities online. This report documents the software engineering process undertaken to design QuickMart, covering the project management, ethical considerations, requirements specification (SRS), and system design (SDD). The project utilizes the Waterfall model to ensure a structured approach to development, moving from detailed requirements gathering to comprehensive system modeling using UML diagrams. This document serves as a blueprint for the implementation phase, ensuring all functional and non-functional requirements are met.

INTRODUCTION

2.1 Purpose

The purpose of this document is to define the requirements and design for the "QuickMart" system. It details the intended features, user interfaces, and the underlying logical structure of the system.

2.2 Scope

QuickMart is a mobile application facilitating grocery delivery from warehouses to customers. The system aims to replace traditional time-consuming shopping with a digital solution offering:

- User authentication and profile management.
- Product browsing and shopping cart functionality.
- Secure payment integration.
- Real-time delivery tracking and warehouse management integration.

PART A: PROJECT MANAGEMENT

3.1.Group Meetings

First Meeting:

Meeting Agenda:

- Date & Time: October 1, 2025
- Time Spent: Two Hours
- Discussion: Project Idea Selection

Attendances:

- Lama Alghofaili
- Member 2
- Member 3
- Member 4
- Member 5
- Member 6
- Member 7
- Member 8
- Member 9

Notes:

- In this meeting, we discussed several project ideas for our software engineering course. Some of the ideas included: food delivery apps, restaurant delivery services, Uber- style human delivery, government apps like TAWAKKALNA and ABSHER, booking platforms like BOOKING.COM, shopping platforms like NOON and AMAZON, educational systems like NOOR, managing an online store, hotel reservations, and hospital appointment systems.
- After discussing all options, we agreed to choose a grocery delivery application as our project, focusing on quick and easy delivery of daily necessities.
- We decided to name the project QUICKMART, reflecting speed, convenience, and simplicity.

PART A: PROJECT MANAGEMENT

3.1.Group Meetings

Second Meeting:

Meeting Agenda:

- Date & Time: October 3, 2025
- Time Spent: One hour and 30 minutes
- Discussion: Project description completed. Lama chosen as group leader.Task distribution and development methodology will be decided next meeting.

Attendances:

- Lama Alghofaili
- Member 2
- Member 3
- Member 4
- Member 5
- Member 6
- Member 7
- Member 8
- Member 9

Notes

- We discussed and created a comprehensive description of the QuickMart project, including the project objectives and the core services the application will provide.
- We talked over the details of the idea and how to implement it practically.
- A team leader was appointed to coordinate tasks and ensure smooth workflow.
- It was agreed that task distribution and the choice of development methodology will be decided in the next meeting.
- All members actively participated and contributed their suggestions and ideas for the project.

PART A: PROJECT MANAGEMENT

3.1.Group Meetings

Third Meeting:

Meeting Agenda:

- Date & Time: October 20, 2025
- Time Spent: Two Hours
- Discussion: Software process model and SRS

Attendances:

- Lama Alghofaili
- Member 2
- Member 3
- Member 4
- Member 5
- Member 6
- Member 7
- Member 8
- Member 9

Notes:

- the team reviewed several process models (waterfall, agile, incremental) and decided to adopt the waterfall model due to its suitability for the project and ease of progress tracking.
- we discussed the use case scenario in detail and worked on developing it. this include defining the main success scenario for the "place order" activity, and identifying alternative flows such as items out of stock, delivery slot unavailability, payment failure, and order cancellation, also specifying postconditions, business rules, special requirements, and assumptions for the scenario.
- the srs were divided among the team members.

PART A: PROJECT MANAGEMENT

3.1.Group Meetings

Fourth Meeting:

Meeting Agenda:

- Date & Time: November 30, 2025
- Time Spent: Two Hours
- Discussion: SDD structure. requirements for architecture model .uml diagrams (activity - sequence - state). class diagram expectations

Attendances:

- Lama Alghofaili
- Member 2
- Member 3
- Member 4
- Member 5
- Member 6
- Member 7
- Member 8
- Member 9

Notes:

- the team reviewed the required sections of the software design document (sdd), including the architecture model and the uml diagram set. the members discussed how each diagram should be structured and the level of detail expected.
- we discussed the general flow for the activity, sequence, and state diagrams, ensuring that all diagrams align with the system behavior and the previous use cases. the group also referenced the existing materials in the shared files to maintain consistency.
- the expectations for the class diagram were clarified, including defining main classes, relationships, attributes, and methods based on the system requirements.
- the deadline for completing all sdd sections was confirmed, and the team agreed to submit everything by wednesday.

PART A: PROJECT MANAGEMENT

3.1.Group Meetings

Fifth Meeting:

Meeting Agenda:

- Date & Time: December 6, 2025
- Time Spent: Two Hours
- Discussion: how to present the project

Attendances:

- Lama Alghofaili
- Member 2
- Member 3
- Member 4
- Member 5
- Member 6
- Member 7
- Member 8
- Member 9

Notes:

• The team discussed how to organize the final presentation and agreed to make it more persuasive rather than purely technical.

• Different approaches were considered, and we chose a story-based pitch to grab the audience's attention.

• We agreed to start by presenting a real problem students face daily, then introduce QuickMart as the solution.

• Presentation roles were divided based on content: marketing slides for strong speakers, technical slides for members specialized in diagrams and system design.

• Final decision: present QuickMart as a scalable business, not just a class project.

PART A: PROJECT MANAGEMENT

3.2. Project Repository

The team uses Google Drive as the main repository for the QuickMart project. You can access the full repository via the link below:
Repository Link: CS383 Project

PART B: SOFTWARE ENGINEERING ETHICS AND RESPONSIBILITY

The QuickMart grocery delivery system was developed by following the main ideas from the ACM/IEEE Software Engineering Code of Ethics.
Our team made sure to act responsibly and ethically while building the system.
We focused on five important principles during the project:

1. User and Public Safety:
   - The system was designed to keep users' data and activities safe.
   - Personal and payment information is protected using encryption.
   - Delivery tracking was added carefully to respect users' privacy.
   - We avoided using any unsafe or untrusted tools that could harm users' security.

2. Privacy and Confidentiality:
   - The app stores sensitive data such as addresses, phone numbers, and payments.
   - We made sure that only the right people can access this data.
   - No information is shared with others without the user's permission.
   - Privacy rules were followed in both the design and development stages.

3. Professional Responsibility and Honesty:
   - Each team member worked honestly and took part in all project phases.
   - We gave credit to all sources and materials we used.
   - Decisions were made together and based on what was right and fair.
   - Our main goal was to create a reliable and useful system for users.

PART B: SOFTWARE ENGINEERING ETHICS AND RESPONSIBILITY

4. Quality and Competence:
   - We tried to make every part of the project clear, correct, and well-organized.
   - We reviewed each other's work to fix mistakes early.
   - The system was tested to make sure it meets the requirements.
   - Everyone in the team learned and improved their skills during the project.

5. Social and Environmental Responsibility:
   - We thought about how the system can help society in a positive way.
   - QuickMart helps save time and fuel by reducing trips to stores.
   - The project avoids any actions that could harm people or the environment.
   - It also makes grocery shopping easier for people who can't move around easily.

By following these five principles {safety, privacy, honesty, quality, and social responsibility} our team showed that we care about ethical and responsible software development.
This helps build trust with users and shows respect for the community.

PART C: SOFTWARE PROCESS ACTIVITIES AND MODELS

5.1 Selected Model: Waterfall Model

The Waterfall Model has been selected as the primary software development process for the QuickMart grocery delivery system project.

5.2 Model Description:

The Waterfall Model follows a linear and sequential approach to software development, where each phase must be fully completed before the next phase begins. The phases relevant to our project include:

- Requirements Analysis and Specification (SRS Document)
- System Design (SDD Document)
- Final Documentation and Presentation

PART C: SOFTWARE PROCESS ACTIVITIES AND MODELS

5.3 Justification for Selection:

The Waterfall Model was chosen for QuickMart based on the following key reasons:

1.Well-Defined and Stable Requirements:

- QuickMart operates as a standardized grocery delivery system with clearly defined business processes
- Core functionalities including order processing, inventory management, and delivery tracking are well-understood and unlikely to undergo significant changes
- The academic context provides fixed requirements through the project guidelines

2. Documentation-Driven Development:

- Comprehensive documentation (SRS and SDD) is a critical assessment requirement in this course
- Waterfall emphasizes thorough documentation at each phase, ensuring all requirements and design decisions are properly recorded
- This approach aligns perfectly with the project's evaluation criteria that prioritize documentation quality

3. Structured Sequential Nature:

- The logical progression of deliverables (SRS → SDD → Presentation) matches Waterfall's phased approach
- Each milestone can be completed fully before moving to the next, reducing integration risks
- This provides clear checkpoints for progress assessment throughout the project timeline

PART C: SOFTWARE PROCESS ACTIVITIES AND MODELS

5.3 Justification for Selection:

4. Predictable Project Management:
   - Fixed academic deadlines and deliverable schedules make Waterfall ideal for planning and execution
   - The model allows for accurate timeline estimation and resource allocation
   - Team members can focus on completing current phase deliverables without uncertainty about future changes

5. Project Type Compatibility:
   - As a grocery delivery system, QuickMart involves standardized operational workflows
   - The sequential nature of order processing (browse → select → pay → deliver) aligns with Waterfall's structured approach
   - Considerations for handling customer data and payment information benefit from Waterfall's rigorous documentation requirements

6. Academic Context Suitability:
   - The educational setting requires clear demonstration of software engineering principles
   - Waterfall's distinct phases allow for structured learning and assessment of each development stage
   - The model facilitates comprehensive coverage of all software engineering activities specified in the course guidelines

PART D: SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

6.1. Requirements Engineering Plan

6.1.1. Requirements Gathering Methods

1- Stakeholder Interviews: Conducted with potential users (students, faculty) to understand their grocery shopping challenges

2- Questionnaire/Survey: Distributed to collect data on preferred delivery times, payment methods, and product categories

3- Domain Research: Analyzed existing grocery delivery apps to identify best practices and common features

4- Brainstorming Sessions: Regular team meetings to generate and refine requirements

6.1.2. Requirements Analysis Approach

1- Categorization: Grouping requirements into functional, non-functional, and external interfaces

2- Prioritization: Using MoSCoW method (Must-have, Should-have, Could-have) to focus on essential features first

3- Validation: Cross-referencing requirements with project guidelines and stakeholder needs

PART D: SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

6.1. Requirements Engineering Plan

6.1.3.Requirements Validation

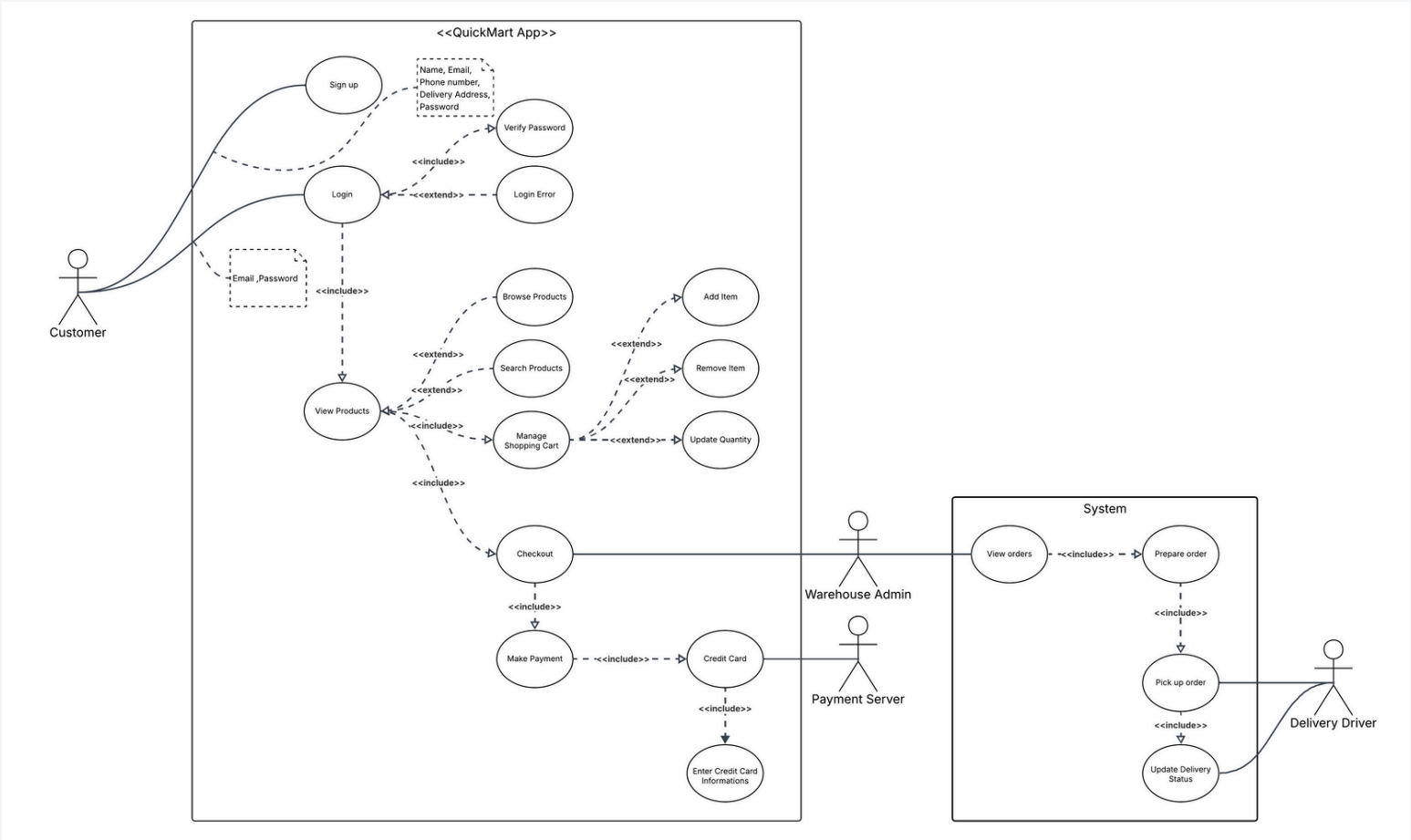
1- Peer Reviews: Team members review each other's requirements for clarity and completeness

2- Prototype Walkthroughs: Using wireframes to validate user workflow understanding

3- Checklist Verification: Ensuring all project guideline requirements are addressed

PART D: SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

6.1.4.UML use case



Use Case Diagram illustrating the core interactions between the system's actors (Customer, Driver, and Admin) and the primary functionalities of the QuickMart application.

PART D: SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

6.2. Functional Requirements

1- User Authentication & Access Control:

- The system shall allow customers to register, log in, and authenticate before accessing checkout and order placement features.

2- Shopping Cart & Item Validation:

- The system shall allow customers to manage cart items (add, remove, update) and verify that the cart contains valid, in-stock items before checkout.

3- Product & Order Summary Display:

- The system shall display detailed product information and a complete order summary including items, prices, subtotal, delivery fee, total cost, and delivery address.

4- Delivery Slot Management:

- The system shall allow customers to select an available delivery time slot and notify them if the selected slot becomes unavailable.

5- Payment Method Selection & Processing:

- The system shall allow customers to choose a payment method and process credit card payments through a secure external payment gateway.

PART D: SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

6.2. Functional Requirements

6- Order Validation & Confirmation:

- The system shall validate stock availability, delivery slot availability, and payment confirmation before creating the final order.

7- Order Creation & Inventory Update:

- The system shall create a confirmed order and automatically update inventory levels for the purchased items.

8- Notifications for Customers & Warehouse:

- The system shall send order confirmation notifications to the customer and notify warehouse staff to begin preparing the order.

PART D: SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

6.3. Non-functional Requirements

1- Performance Requirements

- The system shall load the main interface and product listings within 2 seconds under normal network conditions.
- The system shall process order checkout within a maximum of 3 seconds.

2- Reliability Requirements

- The system shall maintain an uptime of at least 99% to ensure continuous service availability.
- The system shall guarantee that no order data is lost during communication between components (app, warehouse, delivery drivers).

3- Usability Requirements

- The system shall allow users to place an order in a simple and intuitive workflow requiring minimal interactions.
- The user interface shall be easy to navigate and optimized for quick actions, supporting users of all technical backgrounds.

4- Security Requirements

- The system shall encrypt all sensitive user and payment data using industry-standard encryption methods.
- The system shall require secure authentication to prevent unauthorized access to user accounts and personal data.

PART D: SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

6.3. Non-functional Requirements

5- Availability Requirements

- The system shall be available 24/7, except during scheduled maintenance.
- Maintenance downtime shall not exceed 2 hours per month.
- A backup server shall be available to restore service in case of server failure.

6- Compatibility Requirements

- The mobile application shall support major operating systems, including iOS 14+ and Android 10+.
- The web version shall work on modern browsers (Chrome, Safari, Edge, Firefox).
- The system shall adjust automatically to different screen sizes (mobile, tablet, desktop).

PART D: SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

6.4. External Interface Requirements

1. Payment Gateway Integration:
   - The system shall interface with external payment gateways (e.g., Visa, MasterCard, Mada) to process online payments securely. Payment data must be encrypted and comply with PCI DSS standards.

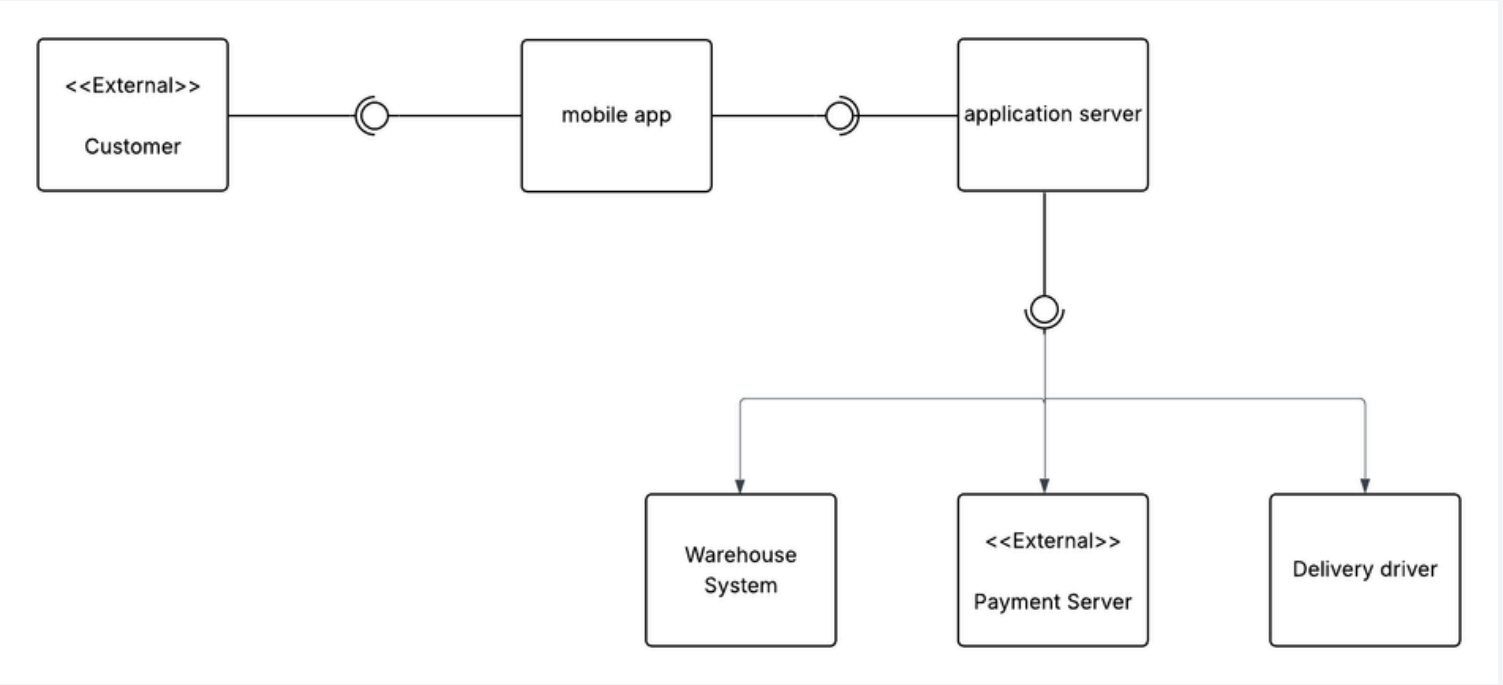
2. Warehouse Management System (WMS) Integration:
   - QuickMart shall communicate with the warehouse system to check real-time inventory levels and update stock after orders are placed. Inventory updates must happen immediately to prevent overselling.

3. Delivery Tracking System Interface:
   - The system shall integrate with the delivery staff's mobile tracking app to monitor and update the delivery status in real-time. Customers should be able to view delivery status and estimated arrival time.

4. Notification Services Integration:
   - The system shall interface with external email and SMS services to send order confirmations, delivery updates, and alerts to customers. Notifications must be sent within 2 minutes of the triggering event.

PART E: SYSTEM AND SOFTWARE MODELING (SDD)

7.1. System Architecture Model

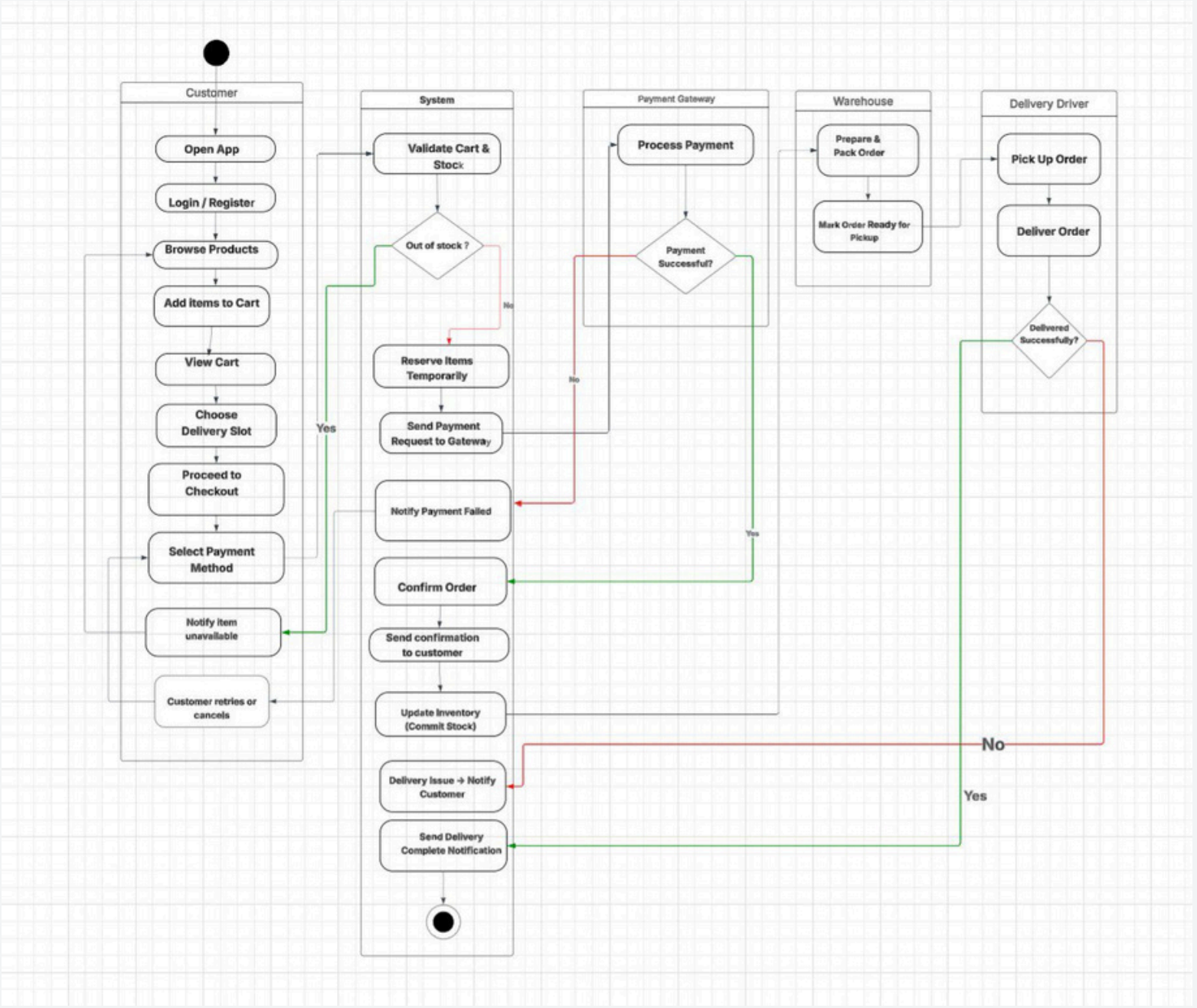


A high-level overview of the QuickMart system architecture, illustrating the layered interaction between the User Interface (UI), Application Logic, and the Central Database, ensuring a scalable and organized system flow.

PART E: SYSTEM AND SOFTWARE MODELING (SDD)

7.2. UML Diagrams

7.2.1. Activity Diagram

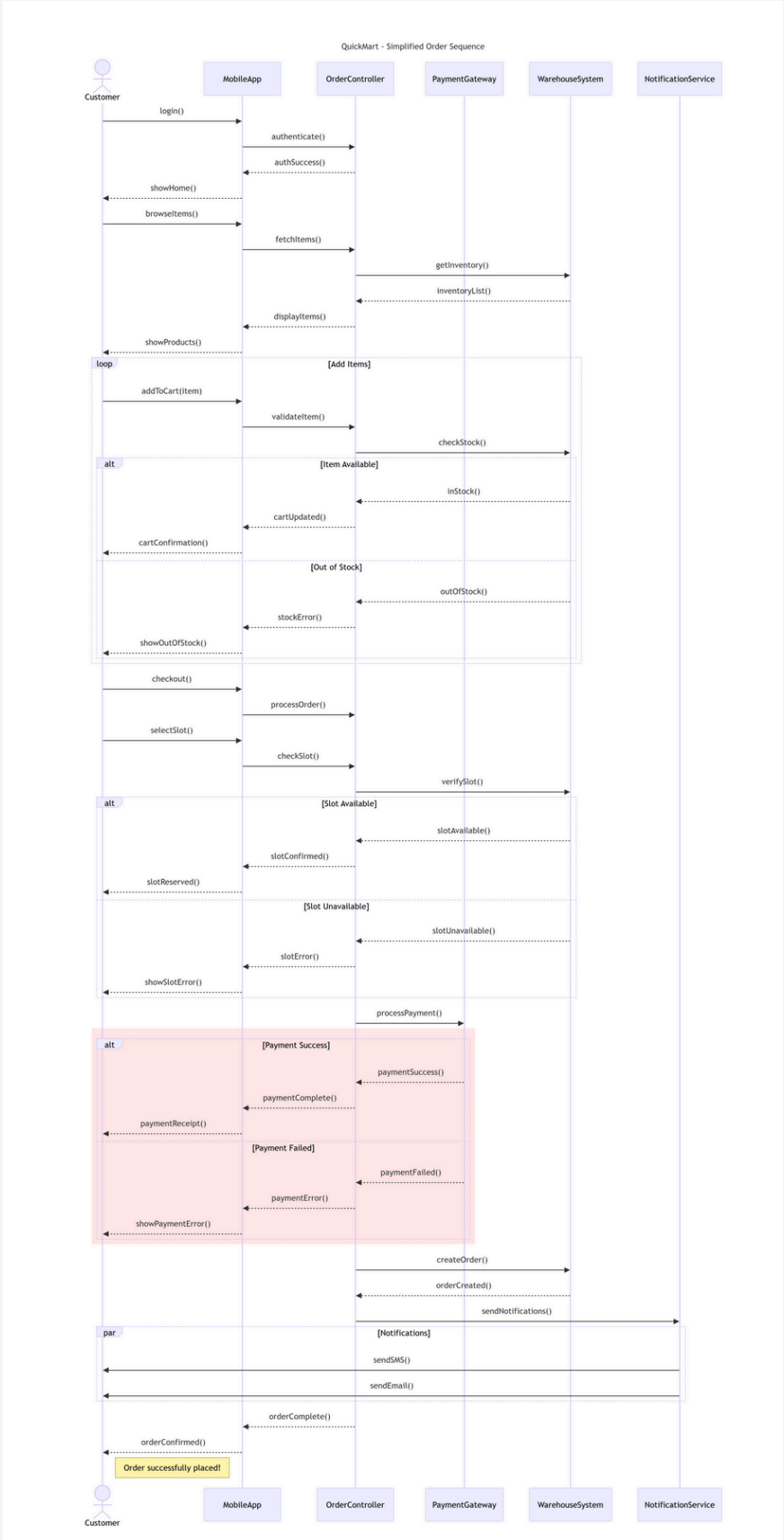


Activity Diagram depicting the sequential flow of the 'Ordering Process,' showing the decision paths from product selection to payment and order confirmation.

PART E: SYSTEM AND SOFTWARE MODELING (SDD)

7.2. UML Diagrams

7.2.2. Sequence Diagram

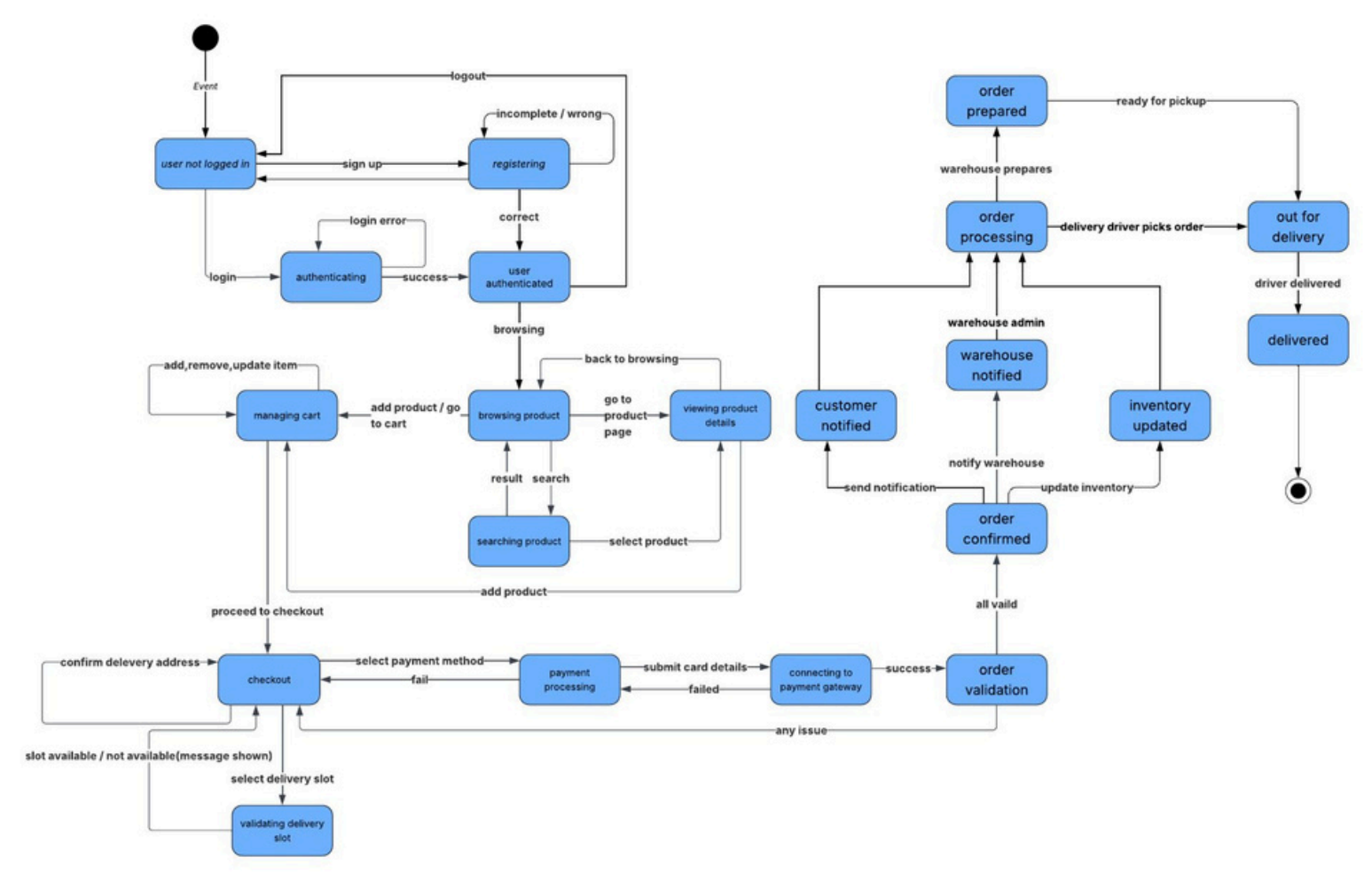


Sequence Diagram showing the object interactions and message exchange logic required to complete a successful 'User Login' and authentication process.

PART E: SYSTEM AND SOFTWARE MODELING (SDD)

7.2. UML Diagrams

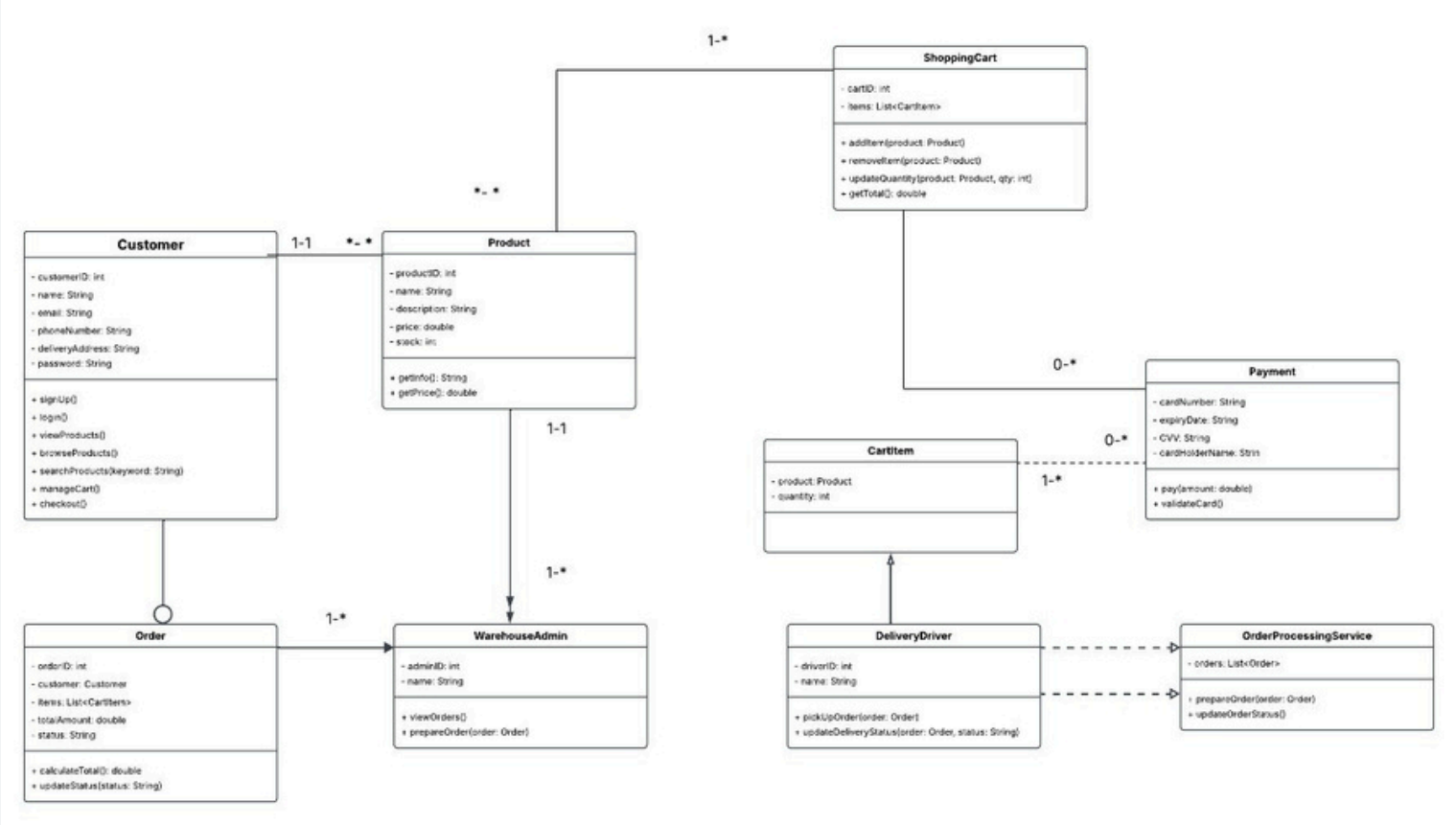
7.2.3. State Diagram



State Machine Diagram representing the various lifecycle states of a grocery order, from 'Pending' through 'Out for Delivery' to 'Completed'.

PART E: SYSTEM AND SOFTWARE MODELING (SDD)

7.3. Class Diagram



Class Diagram defining the static structure of the system, including class attributes, methods, and the relationships (associations) between entities like User, Order, and Product.

CONTRIBUTION TABLE (TASKS PER MEMBER)

Member	Tasks
Lama Alghofaili	**Part C** **Part D: 4.1. Requirements Engineering Plan** **Part D: 6.1.4.UML use case**
Member 2	**Third Meeting** **Part D: 4.2. Functional Requirements** **Part E: 5.1. System Architecture Model**
Member 3	**Fourth Meeting** **Part D: 4.4. External Interface Requirements** **Part E: 5.2.2. Sequence Diagram**
Member 4	**Part B** **Part D: 4.3. Non-functional Requirements** **Part E: 5.3. Class Diagram**
Member 5	**Part B** **Part D: 4.4. External Interface Requirements** **Part E: 5.2.3. State Diagram**
Member 6	**Fifth Meeting** **Part D: 4.2. Functional Requirements** **Part E: 5.2.1. Activity Diagram**
Member 7	**Part B** **Part D: 4.2. Functional Requirements** **Part E: 5.2.3. State Diagram**
Member 8	**Second Meeting** **Part D: 4.3. Non-functional Requirements** **Part E: 5.3. Class Diagram**
Member 9	**First Meeting** **Part D: 4.2. Functional Requirements** **Part E: 5.2.1. Activity Diagram**

CONCLUSION

In conclusion, this report has outlined the comprehensive software engineering process for the QuickMart application. By adhering to the Waterfall model, the team successfully defined stable requirements and produced a robust System Design Document (SDD). The ethical analysis ensures the system respects user privacy and safety, while the detailed UML diagrams provide a clear roadmap for the development phase. The structural foundation laid out in this report confirms that QuickMart is ready for implementation, aiming to deliver a reliable and efficient grocery shopping experience.