

Department of Computer Science

COMP4300 - Graduation Project

Fall 2025/2026

Title of Project: **Dolphin Stock System**

Group Members with IDs :

Ramadan Abo Syam 1212082

Ayman Al Asmar 1211043

Rami Al Rimawi 1211583

Supervisor: **Dr.Iyad Jaber**

**Section – B**

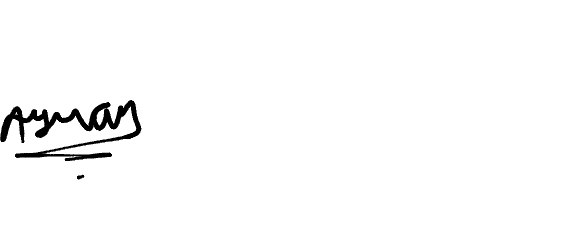
Title of Project: **Dolphin Stock System**

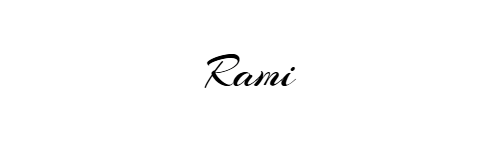
Project No: **1**

Supervisor: **Dr. Iyad Jaber**

Key Areas: **Website Course, Database course,**

**Mobile Application Course**



****

**Section – C**

**Student Signature:\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_**

**Date Submitted: 30 / 6 / 2025**

**First Supervisor Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**First Supervisor Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Date Approved: \_\_\_\_ / \_\_\_\_ / \_\_\_\_\_**

# Abstract

The **Dolphin Stock System** is a digital inventory and sales management solution designed to replace traditional paper-based operations within Dolphin company. The system integrates a web-based platform for accountants and a mobile application for Storage staff, delivery drivers, suppliers, customers, and sales representatives. It streamlines core business processes, including order creation, stock tracking, payment handling, and delivery coordination.

A key feature of the system is its role-based design, where each user accesses only the tools relevant to their role. The mobile application supports offline mode for Storage staff operations and offers real-time

GPS tracking for delivery drivers, enhancing transparency and operational continuity.

By automating workflows, improving coordination, and enabling real-time decision-making, the **Dolphin Stock System** enhances efficiency, reduces human errors, and provides a scalable foundation for business growth.

**Table of Contents**

[**Abstract 3**](#_Toc219572765)

[**Chapter 1: Introduction 8**](#_Toc219572766)

[**1.1 General overview & Motivation 8**](#_Toc219572767)

[**1.2 Aim 9**](#_Toc219572768)

[**1.3 Objectives 10**](#_Toc219572769)

[**1.4 Overview of the Technical Area 11**](#_Toc219572770)

[**1.5 Technology 11**](#_Toc219572771)

[** Dart 11**](#_Toc219572772)

[** Flutter 11**](#_Toc219572773)

[** Supabase 12**](#_Toc219572774)

[**Chapter 2: Background 13**](#_Toc219572775)

[**2.1 Details of relevant theory 13**](#_Toc219572776)

[**2.1.1 Inventory and Sales Management Systems: 13**](#_Toc219572777)

[**2.1.2 Google Maps (GPS) 14**](#_Toc219572778)

[**2.1.3 Offline Synchronization 15**](#_Toc219572779)

[**2.2 Review of past/reported work 17**](#_Toc219572780)

[**2.2.1 Sortly 17**](#_Toc219572781)

[**2.2.2 SMACC 17**](#_Toc219572782)

[**2.2.3 Zoho inventory 18**](#_Toc219572783)

[**2.3 Tools and Technology 19**](#_Toc219572784)

[**2.3.1 Sortly 19**](#_Toc219572785)

[**2.3.2 SMACC 20**](#_Toc219572786)

[**2.3.3 Zoho inventory 20**](#_Toc219572787)

[**Chapter 3: System Analysis and Design 21**](#_Toc219572788)

[**3.1 Product Description 21**](#_Toc219572789)

[**3.1.1 System Objectives 21**](#_Toc219572790)

[**3.1.2 System Main Features 22**](#_Toc219572791)

[**3.1.3 Operating Environments 23**](#_Toc219572792)

[**3.1.4 Constrains 24**](#_Toc219572793)

[**3.1.5 Functional Requirements 25**](#_Toc219572794)

[**3.1.6 Non-Functional Requirement 31**](#_Toc219572795)

[**3.2 Use Case Diagram 32**](#_Toc219572796)

[**3.2.1 Actors 32**](#_Toc219572797)

[**3.2.2 Use Cases Description 33**](#_Toc219572798)

[**3.2.3 Use Cases Diagram 40**](#_Toc219572799)

[**3.3 System Models 43**](#_Toc219572800)

[**3.3.1 Class Diagram 43**](#_Toc219572801)

[**3.3.2 Sequence Diagram 44**](#_Toc219572802)

[**3.3.3 Activity Diagram 48**](#_Toc219572803)

[**3.3.4 State Chart Diagram 52**](#_Toc219572804)

[**3.4 System Architecture 56**](#_Toc219572805)

[**3.4.1 Sub-System 56**](#_Toc219572806)

[**3.4.2 Software Architecture 57**](#_Toc219572807)

[**3.4.3 Deployment Diagram 58**](#_Toc219572808)

[**3.4.4 Component Diagram 59**](#_Toc219572809)

[**3.5 Data Management and ERD 60**](#_Toc219572810)

[**Chapter 4: Conclusion and Future works 61**](#_Toc219572811)

[**4.1 Review of the project 61**](#_Toc219572812)

[**4.2 Next Semester Plan 62**](#_Toc219572813)

[**Conclusion 63**](#_Toc219572814)

[**Bibliography 64**](#_Toc219572815)

[**Appendices 65**](#_Toc219572816)

[** Use Case Specifications 65**](#_Toc219572817)

**Table of Figures**

[**Figure 1 : Dolphin Company Logo 9**](#_Toc201965429)

[**Figure 2 : Dart Language's 11**](#_Toc201965430)

[**Figure 3 : Flutter Language's 11**](#_Toc201965431)

[**Figure 4 : MySQL 12**](#_Toc201965432)

[**Figure 5 : PHP Language's 12**](#_Toc201965433)

[**Figure 6 : Google Maps (GPS) 15**](#_Toc201965434)

[**Figure 7 : Offline Synchronization 16**](#_Toc201965435)

[**Figure 8 : Sortly System 17**](#_Toc201965436)

[**Figure 9 : SMACC System 17**](#_Toc201965437)

[**Figure 10 : Zoho System 18**](#_Toc201965438)

[**Figure 11 : Overall Use Cases Diagram 40**](#_Toc201965439)

[**Figure 12 : 1st Level Mange payment 41**](#_Toc201965440)

[**Figure 13 : 1st level Edit Product quantity and resend 41**](#_Toc201965441)

[**Figure 14: Discuses the changes 42**](#_Toc201965442)

[**Figure 15 : decision with the order sender 42**](#_Toc201965443)

[**Figure 16 : Class Diagram 43**](#_Toc201965444)

[**Figure 17 :Create and Send Order (Customer) Sequence Diagram 44**](#_Toc201965445)

[**Figure 18 : Create and Send Order (Sales Representative) Sequence Diagram 45**](#_Toc201965446)

[**Figure 19 : Supplier Receives and Responds Stock-In Order Sequence Diagram 46**](#_Toc201965447)

[**Figure 20 : Storage Staff Handles Stock-Out Order Sequence Diagram 47**](#_Toc201965448)

[**Figure 21 : Create and Send Order (Customer) Activity Diagram 48**](#_Toc201965449)

[**Figure 22 : Create and Send Order (Sales Representative) Activity Diagram 49**](#_Toc201965450)

[**Figure 23 : Supplier Receives and Responds Stock-In Order Activity Diagram 50**](#_Toc201965451)

[**Figure 24 : Storage Staff Handles Stock-Out Order Activity Diagram 51**](#_Toc201965452)

[**Figure 25 : Create and Send Order (Customer) State Chart Diagram 52**](#_Toc201965453)

[**Figure 26 : Create and Send Order (Sales Representative) State Chart Diagram 53**](#_Toc201965454)

[**Figure 27 : Supplier Receives and Responds Stock-In Order State Chart Diagram 54**](#_Toc201965455)

[**Figure 28 : Storage Staff Handles Stock-Out Order State Chart Diagram 55**](#_Toc201965456)

[**Figure 29 : Software Architecture Diagram 57**](#_Toc201965457)

[**Figure 30 : Deployment Diagram 58**](#_Toc201965458)

[**Figure 31 :Component Diagram 59**](#_Toc201965459)

[**Figure 32 : Data Management and ERD Diagram 60**](#_Toc201965460)

**Table of Tables**

[**Table 1 : Comparison Table 20**](#_Toc201965461)

[**Table 2 : Actor's 33**](#_Toc201965462)

[**Table 3 : Login Description 34**](#_Toc201965463)

[**Table 4 : Assign Storage Staff to Prepare Order Description 34**](#_Toc201965464)

[**Table 5 : Customer order and discuss with accountant 35**](#_Toc201965465)

[**Table 6 : Sales Rep Create and Send Order 36**](#_Toc201965466)

[**Table 7 : Edit Quantity of Product and Resend Description 37**](#_Toc201965467)

[**Table 8 : Manage Products Description 38**](#_Toc201965468)

[**Table 9 : Manage Mobile Accounts Description 38**](#_Toc201965469)

[**Table 10 : Create and Send Stock-Out Orders Description 39**](#_Toc201965470)

[**Table 11 : Assign and Send Delivery Driver to Order Description 39**](#_Toc201965471)

[**Table 12 : add Payment Description 40**](#_Toc201965472)

[**Table 13 : Next Semester Plan 63**](#_Toc201965473)

[**Table 14 : View Products 66**](#_Toc201965474)

[**Table 15 : View Sales Report 66**](#_Toc201965475)

[**Table 16 : Discuss the Changes 67**](#_Toc201965476)

[**Table 17 : View Location of Delivery Driver 67**](#_Toc201965477)

[**Table 18 : Resend Order 68**](#_Toc201965478)

[**Table 19 : Filter Products List 68**](#_Toc201965479)

[**Table 20 : Select Customer 69**](#_Toc201965480)

[**Table 21 : View Assigned Orders 69**](#_Toc201965481)

[**Table 22 : View Customer Path 70**](#_Toc201965482)

[**Table 23 : View a Receipt for the Signature 70**](#_Toc201965483)

[**Table 24 : Search for Product 71**](#_Toc201965484)

[**Table 25 : Receive Stock-In Orders 71**](#_Toc201965485)

# Chapter 1: Introduction

## General overview & Motivation

Inventory and sales management are essential activities in many businesses that depend on the continuous flow of goods between suppliers, storage facilities, and customers. In small to medium-sized enterprises, especially in wholesale and distribution environments, these processes are still commonly handled using traditional methods such as paper records, manual data entry, phone calls, and simple spreadsheets. Although these methods may work when the business size is limited, they become inefficient and difficult to control as the number of transactions and daily operations increases.

Relying on manual inventory and sales management creates several challenges. Human errors in record-keeping can result in inaccurate stock levels, duplicated orders, or missing transactions. In addition, the absence of real-time information makes it hard for management to know the actual availability of products, which often leads to overstocking or shortages. Communication delays between sales, storage, and delivery teams further reduce operational efficiency and may negatively affect customer satisfaction.

Another major limitation of traditional approaches is the lack of effective tools for monitoring order progress and delivery operations. Without digital tracking systems, it becomes difficult to follow the status of orders, coordinate deliveries, or confirm that products have been successfully delivered. These issues are especially critical in businesses where timely delivery, transparency, and coordination between departments are necessary for maintaining smooth operations and customer trust.

To address these challenges, many organizations are shifting toward digital inventory and sales management systems that automate manual processes and replace paper-based workflows. Such systems provide centralized control, real-time data access, and better coordination between different business units. The Dolphin Stock System is proposed in response to this need as a digital solution that automates traditional inventory and sales processes, including manual record-keeping, order handling, and delivery coordination. By replacing these conventional methods with integrated web and mobile platforms, the system aims to improve efficiency, reduce errors, and support more organized and scalable business operations within Dolphin Company.



Figure 1 : Dolphin Company Logo

## Aim

This project aims to design and develop a comprehensive Inventory and Sales Management System that digitizes and streamlines stock handling, order processing, and delivery tracking through integrated web and mobile platforms. The system connects all stakeholders, including accountants, storage managers, storage staff, delivery drivers, suppliers, customers, and sales representatives, to enhance real-time collaboration, reduce operational delays, and improve accuracy.

In addition, the system supports offline functionality for storage staff, live GPS tracking for delivery drivers, and role-based workflows. These features enable efficient, scalable, and well-coordinated business operations across all company departments.

## Objectives

The main objectives of this project are as follows:

* To digitize and organize inventory and sales operations by replacing traditional paper-based methods with a unified digital system.
* To improve operational efficiency by reducing manual work, minimizing human errors, and accelerating order processing across different departments.
* To enhance coordination and communication between all stakeholders involved in the business workflow, including management, storage, sales, suppliers, and delivery teams.
* To provide accurate and up-to-date information that supports better monitoring, control, and decision-making within the company.
* To support operational continuity in environments with limited or unstable internet connectivity by enabling the system to adapt to offline operation, ensuring that essential processes are not interrupted.
* To develop a structured and scalable system that can adapt to business growth and evolving operational requirements.

## ****Overview of the Technical Area****

This project falls under **business process automation**, with a focus on **inventory control, sales management, and delivery coordination**.

The technical stack used includes:

* **Flutter (Dart)** – For developing cross-platform mobile apps
* **Supabase** – a backend platform that provides a PostgreSQL database, authentication, and APIs, allowing applications to manage data and users without building a custom backend.
* **Firebase or local storage** – For push notifications and offline syncing This technical foundation allows the system to be responsive, accessible across platforms, and easy to maintain

## Technology

### Dart

Figure 2 : Dart Language's

Dart is an object-oriented client-optimized programming language used to build mobile and web applications on various platforms.[6]

### Flutter

Figure 3 : Flutter Language's

Flutter is an open-source UI software development kit created by Google. It is used to develop cross platform applications from a single codebase for any web browser, Android, IOS, Linux, macOS, and windows.[4]

### OIPSupabase

Figure 4 : Supabase

Supabase is an open-source backend-as-a-service (BaaS) platform built on PostgreSQL. It is used to manage application databases, authentication, and APIs, allowing web and mobile applications to interact with backend services efficiently.

Figure 3 : Supabase

# Chapter 2: Background

As businesses grow and customer demands increase, relying on manual, paper-based processes for managing inventory and sales is no longer a practical option. Many companies face challenges such as inconsistent stock levels, delays in order fulfillment, and difficulty tracking product movement. To address these issues, the development of a digital Inventory and Sales Management System offers a modern solution that enhances accuracy, reduces human error, and streamlines operations. These systems enable real-time tracking of inventory levels, simplify sales workflows, and provide valuable data insights that support faster and more stable decision-making. By transitioning from traditional methods to digital platforms, companies especially in sectors like sanitary ware can achieve greater efficiency, improved customer service, and enhanced scalability.

## Details of relevant theory

### Inventory and Sales Management Systems:

Inventory and sales management Systems are key tools that help businesses manage product quantities, order processing, and sales monitoring more efficiently. These systems play an important role in improving day-to-day operations and ensuring that products are always available — which is especially crucial in fast-paced environments involving both retail and wholesale transactions, where stock movement is dynamic and time-sensitive.[3]

As retail and wholesale operations grow more complex there's a growing need for systems that integrate all key functions into one platform. For example, having a centralized web dashboard for accountants to manage transactions and generate reports, alongside a dedicated mobile application for Storage staff and delivery personnel, creates a structured workflow that increases accountability and speeds up execution.[2]

In terms of business resilience, an effective inventory system allows companies to respond quickly to supply chain disruptions or sudden shifts in customer demand. With real-time tracking and reporting tools, managers can detect issues early, optimize restocking processes, and ensure products are available when needed.[1]

These systems also help improve internal communication. Real-time updates and live order tracking reduce delays and miscommunication between departments. Features like the ability to assign, accept, reject, or comment on orders keep everyone aligned and informed at every stage of the workflow. [1]

Lastly, built-in alerts and access control features ensure that each team member has visibility and permissions tailored to their role. This minimizes confusion, enhances security, and supports a smooth and efficient business process — whether in wholesale distribution or direct-to-consumer retail operations

### Google Maps (GPS)

Google Maps is a powerful web service that provides detailed geographic data about locations and regions worldwide. In addition to traditional road maps, Google Maps offers satellite imagery, aerial views, and, in some areas, street-level visuals captured by moving vehicles. [9]

Within the Dolphin Stock System, Google Maps is integrated with GPS technology to provide accurate, real-time tracking of delivery drivers. This enables the system to display live driver locations as they move along delivery routes, offering precise updates for the accountant interface. Thanks to GPS integration, delivery status and driver location information shown on the system are highly reliable, allowing management to make informed decisions and respond promptly to any delivery issues. [9]

The inclusion of Google Maps in Dolphin Stock System significantly improves logistics and delivery transparency. It allows the accountant and administrative staff to monitor routes, estimate arrival times, and optimize delivery operations. This integration ultimately enhances customer satisfaction by ensuring timely deliveries and providing visibility throughout the delivery process.[9]

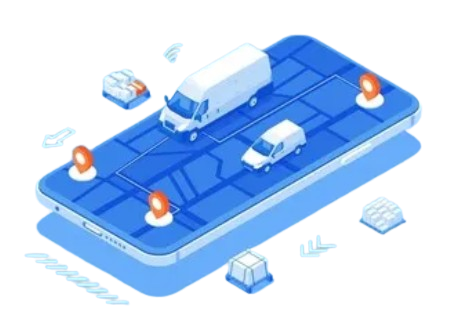


Figure 5 : Google Maps (GPS)

### Offline Synchronization

In the Dolphin Stock System, offline synchronization is designed specifically for storage staff, enabling them to continue performing essential warehouse operations even when an internet connection is unavailable. This feature is particularly important in environments where network connectivity may be weak or unstable, as it ensures that daily operations are not disrupted. [8]

When storage staff use the mobile application, the system stores operational data locally on the device using SQLite through the sqflite package. This local data storage allows storage staff to view assigned tasks and record inventory movements without relying on real-time communication with the central server. The data is retained locally until an internet connection becomes available. [8]

Once a network connection is detected, the application automatically initiates the synchronization process by sending the locally stored data to the central database. At the same time, the local database is updated with any new information retrieved from the server, ensuring data consistency between the mobile application and the central system. [8]

By limiting offline functionality to storage staff and using SQLite for local data storage, this approach ensures uninterrupted warehouse operations, maintains data accuracy, and improves the overall efficiency of the Dolphin Stock System. [8]

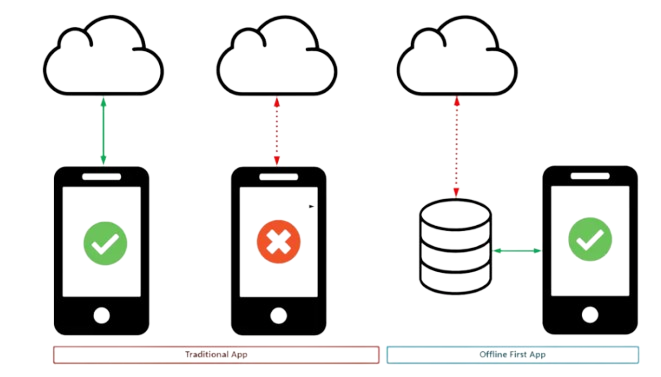


Figure 6 : Offline Synchronization

## Review of past/reported work

### Sortly

Sortly is a visual inventory management application that allows users to organize inventory using folders and photos. It supports QR/barcode scanning, offline syncing, and is ideal for businesses that want simple stock tracking with mobile access.[Sortly](https://www.sortly.com/)

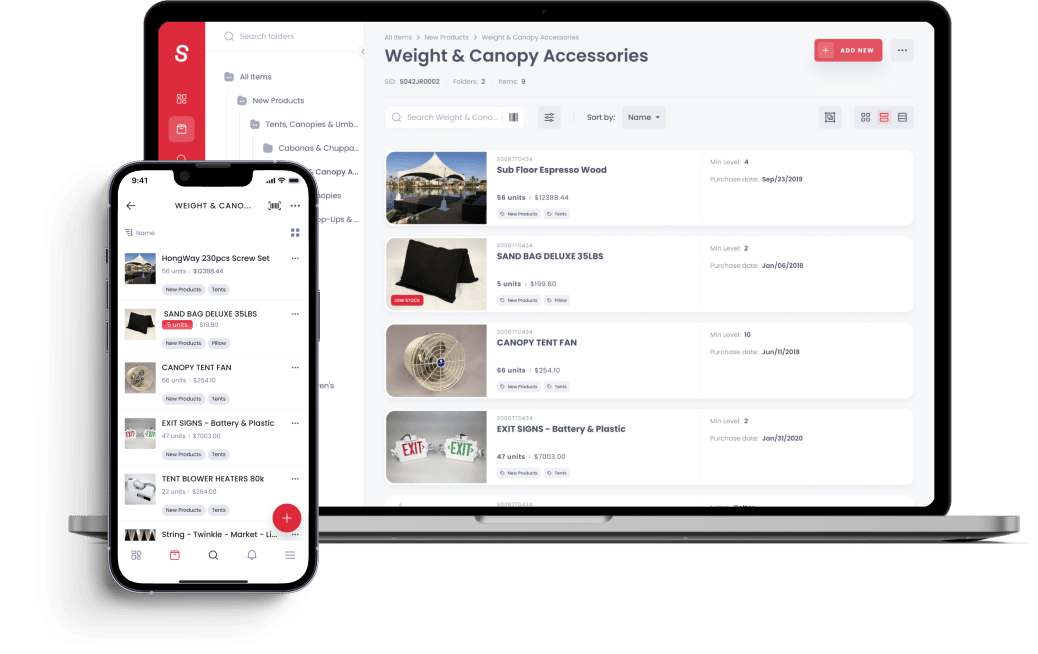


Figure 7 : Sortly System

### SMACC

SMACC is an ERP software with modules for accounting, inventory, HR, and point of sale (POS). It is mostly used for managing finances, automating invoices, and controlling product records in medium to large businesses.[SMACC](https://www.smacc.com/en/accounting-software/inventory-management-software/)



Figure 8 : SMACC System

### Zoho inventory

Zoho Inventory is a cloud-based inventory and order management system that allows businesses to track stock, manage orders, process shipments, and generate reports. It’s suitable for multi-channel sales and offers automation features for small to mid-sized businesses. [Zoho](https://www.zoho.com/us/inventory/)

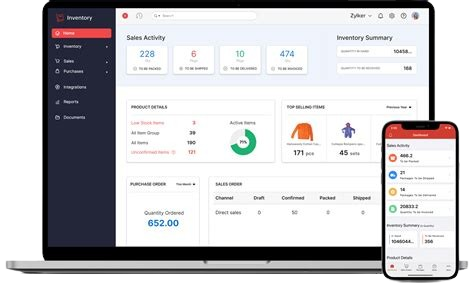
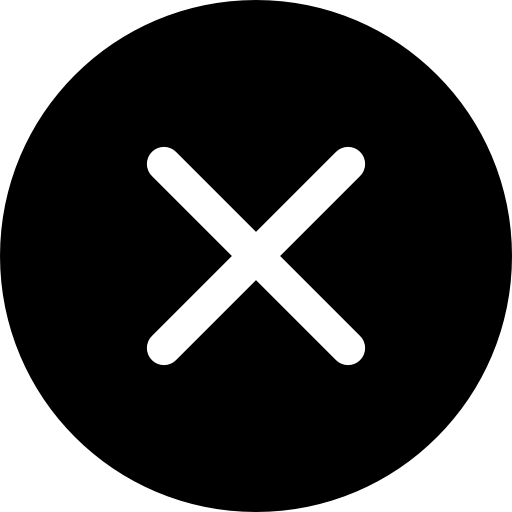


Figure 9 : Zoho System

**Comparison**

Table 1 : Comparison Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Zoho Inventory** | **SMACC** | **Sortly** | **Dolphin Stock system ( our project)** |
| **Offline syncing** |  |  |  |  |
| **Tracking delivery driver** |  |  |  |  |
| **Generate Reports** |  |  |  |  |
| **User Permissions Control** |  |  |  |  |
| **Role-Based Interfaces** |  |  |  |  |
| **Signed Delivery Confirmation** |  |  |  |  |
| **Customer Access & Interaction** |  |  |  |  |
| **Supplier Portal** |  |  |  |  |
| **Sales Representative Access & Interaction** |  |  |  |  |
| **Check Management** |  |  |  |  |
| **Alerts & Notifications** |  |  |  |  |
| **Multi-Inventory**  **Support** |  |  |  |  |
| **Storage Location Management** |  |  |  |  |
| **Batch Tracking** |  |  |  |  |
| **Order Splitting Across Inventories** |  |  |  |  |
| **Damaged products Management** |  |  |  |  |

**Partial =** **Yes =** **No =**

## Tools and Technology

### Sortly

* **Languages:** Java, Python, JavaScript
* **Tools:** Babel, Jupyter, Adobe XD, Apache
* **Integrations:** QuickBooks, Slack, Teams, API

### SMACC

Specific details about SMACC's technology stack are **not publicly available.**

### Zoho inventory

* **Languages:** Java (Backend), JavaScript (Frontend)
* **Tools:** Zoho proprietary frameworks, Zoho Cloud Infrastructure

# Chapter 3: System Analysis and Design

## Product Description

### System Objectives

* **Implement Role-Based Access and Interfaces**

Provide distinct system interfaces tailored to each role: Accountant, Storage Staff, Delivery Driver, Supplier, Customer, and Sales Representative ensuring each user only accesses the features relevant to their responsibilities.

* **Enable Real-Time Inventory Management**  
  Track product quantities across multiple Storage s in real time, with support for both stock-in and stock-out operations.
* **Support Offline Functionality**  
  Allow Storage staff to work without internet connectivity by storing actions locally and synchronizing them automatically once reconnected.
* **Enhance Delivery Tracking**  
  Integrate GPS-based tracking via Google Maps to monitor delivery driver locations and routes in real time from the accountant interface.
* **Order Lifecycle Automation**

Digitize the full order process: creation, review, preparation, delivery, confirmation including support for quantity edits and multi-party approvals.

### System Main Features

The Dolphin Stock System offers a comprehensive suite of features designed to improve inventory tracking, streamline order processing, and enhance coordination between company departments. Below are the main features provided by the system:

* **Role-Based Access and Interfaces:**

Each user (Accountant, Storage Staff, Supplier, Customer, Sales Representative, and Delivery Driver) has a dedicated interface tailored to their responsibilities and permissions.

* **Inventory Management:**

Real-time tracking of product quantities across Storage s with support for stock-in and stock-out operations, alerts for low stock levels, and inventory movement history.

* **Order Management Workflow:**

Orders can be placed by customers or sales representatives, reviewed by accountants, and updated if necessary. A multi-level approval flow allows for quantity adjustments and customer/supplier confirmations.

* **Offline Syncing for Storage Staff:**

Storage users can work without internet access. All actions are saved locally and automatically synchronized with the server once connectivity is restored.

* **Delivery Tracking with Google Maps Integration:**

Delivery drivers are tracked live via GPS, and the accountant can monitor their routes through an interactive map, allowing better route planning and delivery coordination.

* **Digital Delivery Confirmation:**

Customers sign directly on the delivery driver mobile app upon receiving their orders, ensuring a secure and verifiable delivery process.

* **Supplier Interaction:**

Suppliers receive order requests from the accountant and can either approve, reject, or suggest quantity changes. Their response is sent back to the accountant for confirmation.

* **Customer and Sales Rep Features:**

Customers can place and confirm orders directly, while sales reps can submit orders on behalf of their assigned customers, improving field-level sales efficiency.

* **Reporting and Logs:**

The system provides monthly reports, transaction logs, and audit trails for all key actions, ensuring transparency and accountability.

* **Check Management and Payment Tracking:**

Accountants can record and monitor check payments, due dates, and associated check owner.

### Operating Environments

Our project is an integrated system consisting of two main interfaces: a mobile application and a web platform. The mobile application was developed using the **Flutter** framework and the Dart programming language, making it compatible with both Android and iOS platforms. It is used by Storage staff and delivery drivers. The web platform, designed for accountants, was developed using **HTML**, **CSS**, and **PHP**, and can be accessed through desktop or mobile web browsers. The system relies on a **MySQL** database, with data exchanged between its components through **REST APIs** to ensure smooth and secure synchronization. Additionally, the mobile application supports offline mode, allowing users to continue performing their tasks seamlessly even in the absence of an internet connection.

### Constrains

**Internet Dependency (Web Platform):** The accountant's web interface requires an active internet connection to function, as it interacts with the database and API in real time.

**Offline Syncing Delay:** Orders created or updated by Storage staff in offline mode will not be visible to the accountant until the mobile device reconnects to the internet and the data is successfully synced.

**GPS Dependency for Delivery Devices:** Delivery drivers' devices must support GPS functionality to enable live location tracking and route navigation features within the mobile application.

### Functional Requirements

**User Requirements & System Requirements**

**Web Browser (Accountant account)**

* **UR-1:** The system shall allow Accountant to register, and create accounts using needed information and managing all the accounts.
* **SR-1.1:** The system shall allow the accountant to create new user accounts by selecting a user role (e.g., Delivery Driver, Storage Staff, Customer, or Sales Representative), and prompt the accountant to enter the required information based on each role with core registration information such as identity number and password.
* **SR-1.2:** The system shall allow the accountant to assign specific roles and permissions to each user such as Delivery Drivers, Storage staff, Customer and Sales Representative.
* **SR-1.3:** The system shall log all user account activities for auditing purposes.
* **UR-2:** The system shall allow the accountant to add, delete, view, and edit product.
* **SR-2.1:** The system shall enable the accountant to add a new product by providing essential information, including name, category, wholesale price, selling price, weight, dimensions (height and width), batch number, and the associated supplier.
* **SR-2.2:** The system shall allow the accountant to view and modify existing product information as needed.
* **SR-2.3:** The system shall implement a soft delete mechanism for products, allowing the accountant to mark a product as deleted by setting an is\_active as no attribute, ensuring data integrity and preserving historical associations.
* **UR-3:** The system shall allow the accountant to create or receive new orders and assign them to designated Storage staff for preparation.
* **SR-3.1:** The system shall allow the accountant to create new orders by selecting from available products, specifying the required quantity, choosing the customer, entering the applicable order vat percent, and assigning the order to a designated storage staff member for preparation.
* **SR-3.2:** The system shall allow the accountant to receive orders submitted by customers or Sales Representatives, review their details, and assign them to the appropriate Storage staff member for preparation.
* **SR-3.3:** The system shall allow the accountant to receive the prepared order from the storage staff, review its contents, and assign it to a designated delivery driver for dispatch.
* **UR-4:** The system shall allow the accountant to review and approve or reject modifications made to orders by the Storage staff.
* **SR-4.1:** The system shall display a list of order modification requests submitted by Storage staff, including the reason for each requested change.
* **SR-4.2:** The system shall allow the accountant to approve or reject each request with optional comments.
* **SR-4.3:** The system shall allow the accountant to forward order modifications to the customer for approval. Based on the customer's response, the system shall either send the order to Storage staff or notify the accountant of its cancellation.
* **UR-5:** The system should allow the accountant to filter and search products within the inventory.
* **SR-5.1:** The system shall allow the accountant to perform product searches using the product name or unique identifier.
* **SR-5.2:** The system shall allow the accountant to filter products by category, stock level, supplier, or by selecting a specific date range during which the product appeared in stock-in or stock-out transactions.
* **UR-6:** The system should display timely alerts and notifications to the accountant.
* **SR-6.1:** The system shall generate real-time alerts and notifications for the accountant regarding key events, including low stock levels, new or updated incoming orders, and upcoming check due dates.
* **UR-7:** The system shall auto-update order statuses in real time based on user actions.
* **SR-7.1:** The system shall automatically update the status of each order based on actions performed by mobile app users and display the real-time status updates to the accountant.
* **UR-8:** The system shall track the real-time locations of delivery drivers and display them to the accountant through an interactive map interface.
* **SR-8.1:** The system shall integrate with GPS services to capture and process the live location data of delivery drivers.
* **SR-8.2:** The system shall continuously receive periodic location updates from the drivers’ mobile applications and refresh their positions on the interactive map in near real time.
* **SR-8.3:** The system shall allow the accountant to select and focus on the live location of any individual delivery driver.
* **UR-9:** The system shall support the management and tracking of payment transactions (cash or check).
* **SR-9.1:** The system shall allow the accountant to record payment transactions by specifying the payee type (Customer, Supplier), payment type (Cash or Check), and related payment details.
* **SR-9.2:** If the payment type is a check, the system shall require entering additional information, including bank name, check image, check amount, issue date, and exchange date.
* **SR-9.3:** The system shall store all payment records and allow the accountant to filter and search by payee type, payment type, date, or status (e.g., pending, exchange, transferred).
* **UR-10:** The system shall generate monthly sales and inventory reports.
* **SR-10.1:** The system shall allow selecting reporting periods and metrics.
* **SR-10.3:** The system shall allow scheduling automated report generation.
* **SR-10.4:** The system shall provide visual summaries (charts, graphs) in the report.
* **UR-11:** The system shall support the management of stock-in orders with suppliers.
* **SR-11.1:** The system shall allow the accountant to create stock-in orders by selecting products, specifying quantities, and send the order to a supplier.
* **SR-11.2:** The system shall display a summary of discrepancies (if any) between ordered and received quantities to the accountant for review and approval.

**For Mobile App**

* **UR-1:** The system shall allow the user to log in.
* **SR-1.1:** The system shall authenticate users using their identity number and password, and require them to select their role upon login.
* **SR-1.2:** The system shall support session management.
* **SR-1.3:** The system shall provide a password recovery mechanism via registered email address.

* **UR-2:** The system shall send real-time push notifications to all mobile app users, regardless of their role, whenever a new order is assigned or an existing order is modified by the accountant.
* **SR-2.1:** The system shall identify the role of the mobile app user and send the relevant push notification based on the assigned or modified order.
* **SR-2.2:** The system shall ensure that offline users receive queued notifications once they reconnect to the internet.

**For Storage staff Account**

* **UR-3:** The system shall allow Storage staff to view and manage orders assigned by the accountant.
* **SR-3.1:** The system shall display a list of all incoming orders assigned to the Storage staff for preparation.
* **SR-3.2:** The system shall store order data locally on the mobile device to ensure accessibility in offline mode and automatically synchronize actions performed offline once an internet connection is restored.
* **SR-3.3:** The system shall allow Storage staff to accept or reject assigned orders, and require a reason to be provided in case of rejection.
* **SR-3.4:** The system shall allow Storage staff to edit the quantity of some product in order and write the reason for modified.
* **UR-4:** The system shall allow the Storage staff to create new stock-in orders from suppliers and submit them to the accountant for confirmation.
* **SR-4.1:** The system shall provide an interface for Storage staff to initiate a new stock-in order by selecting a supplier and product, including quantity.
* **SR-4.2:** The system shall allow the Storage staff to attach optional comments to the order.

**For Delivery Driver Account**

* **UR-5:** The system shall allow the Delivery Driver to view delivery orders assigned to them by the accountant.
* **SR-5.1:** The system shall display a list of assigned delivery orders.
* **SR-5.2:** The system shall show order details including customer info.
* **UR-6:** The system shall allow the Delivery Driver to view the customer’s delivery route via an interactive map interface.
* **SR-6.1:** The system shall show order details including customer info.
* **SR-6.2:** The system shall calculate and display the optimized route.
* **SR-6.3:** The system shall update the route in real-time based on GPS.
* **UR-7:** The system shall allow the Delivery Driver to present a receipt to the customer for signature on the mobile device.
* **SR-7.1:** The system shall display a digital receipt after delivery.
* **SR-7.2:** The system shall allow the customer to sign on the device screen.
* **UR-8:** The system shall send the real-time GPS location of the Delivery Driver to the Accountant.
* **SR-8.1:** The system shall capture GPS coordinates periodically.
* **SR-8.2:** The system shall send GPS data to the server using a secure API.
* **SR-8.3:** The system shall update the accountant’s map dashboard.
* **SR-8.4:** The system shall allow tracking the delivery history.

**For Supplier Account**

* **UR-9:** The system shall allow the Supplier to view all incoming stock-in orders assigned by the accountant.
* **SR-9.1:** The system shall display a list of all pending stock-in orders assigned to the Supplier.
* **SR-9.2:** The system shall allow the Supplier to open and view order details, including product names and quantities.
* **UR-10:** The system shall allow the Supplier to update the quantities of products within a stock-in order prior to submission for Accountant approval.
* **SR-10.1:** The system shall provide editable quantity fields for each product in the stock-in order.
* **SR-10.2:** The system shall allow the Supplier to submit the modified order back to the accountant for review.

**For Customer Account**

* **UR-11:** The system shall allow the customer to place an order and send it to the accountant for processing.
* **SR-11.1:** The system shall provide an interface for the customer to initiate a new order by selecting products and specifying quantities.
* **SR-11.2:** The system shall notify the customer of any modifications made by the accountant and allow the customer to approve or reject

**For sales Representative Account:**

* **UR-12:** The system shall allow the Sales Representative to create and submit new orders to the accountant on behalf of assigned customers.
* **SR-12.1:** The system shall enable the Sales Representative to select a customer from their assigned list and specify the products and quantities for the order.
* **SR-12.2:** The system shall transmit the created order to the accountant for review and processing.
* **SR-12.3:** The system shall display the accountant’s response to the Sales Representative, indicating whether the order was approved or rejected.

### Non-Functional Requirement

* **Security**  
  The system uses REST APIs combined with strong authentication and access control. Every user has to log in to access the system, and based on their role like accountant, storage staff, or supplier they only see and use the features they’re authorized to access. This keeps the system safe and prevents unauthorized actions.
* **Usability**  
  The interface was designed to be simple and straightforward, so users can quickly learn how to use the system without needing much training. Whether on desktop or mobile, screens are clean and focused on showing only what the user needs at the moment.
* **Availability**  
  The system is designed to be accessible and functional whenever users need it. By supporting offline mode for storage staff, Dolphin Stock System ensures that critical tasks such as preparing orders and updating inventory can continue even in areas with poor or no internet connection. Once the device reconnects, all pending actions are automatically synchronized with the central database. This high level of availability reduces downtime, improves operational continuity, and ensures the system remains usable in a wide range of working environments.
* **Performance**  
  Whether you're viewing reports or syncing large orders, the system is designed to respond quickly. Behind the scenes, smart caching and optimized database queries help keep everything running smoothly even when many users are active

## Use Case Diagram

### Actors

|  |  |
| --- | --- |
| **Actor** | **Description** |
| **Accountant** | This actor represents someone who is uses the web application to manage and track stock-in and stock-out orders, handle payments, generate reports, and manage user accounts. |
| **Storage staff** | This actor represents someone who is uses the mobile app to receive and prepare stock-out orders and to create stock-in orders upon receiving it from suppliers. |
| **Delivery driver** | This actor represents someone who is uses the mobile app to view assigned deliveries, navigate to customer locations, and confirm deliveries through digital signatures. |
| **Customer** | This actor represents someone who is uses the mobile app to place new orders, and approve any modifications made by the accountant. |
| **Supplier** | This actor represents someone who is uses the mobile app to view stock-in orders submitted by the accountant, review any modifications, and proceed with fulfillment based on the Accountant's response. |
| **Sales Representative** | This actor represents someone who is uses the mobile app to create and submit orders on behalf of assigned customers and receive feedback from the accountant. |

Table 2 : Actor's

### Use Cases Description

1. *Use Case Name : Login*

|  |  |
| --- | --- |
| Actors | Sales Rep, Customer, Storage Staff, Supplier, Delivery Driver |
| Description | This use case allows different system users to log in and access their respective dashboards. |
| Pre-Conditions | User must have a valid account and credentials. |
| Flow of Events | 1- User opens the login screen. 2- User enters username and password. 3- System verifies the credentials. 4- System grants access and redirects the user to their respective interface. |
| Alternative Flows | - If the information is invalid, an error message is displayed, and the user is asked to retry. |
| Post-Conditions | User is logged in and redirected to their dashboard. |
| Data Used | Username, password, role information. |
| Trigger | User wants to access the system. |

Table 3 : Login Description

1. *Use Case Name : Assign Storage Staff to Prepare Order*

|  |  |
| --- | --- |
| Actors | Accountant |
| Description | This use case allows the accountant to assign a storage staff member to prepare the order. |
| Preconditions | Accountant must have received a valid customer order. |
| Flow of Events | 1- Accountant opens a received order. 2- Accountant selects available storage staff. 3- System assigns the staff member and notifies them. |
| Alternative Flows | - If no staff are available, the system alerts the accountant and queues the task. |
| Postconditions | Storage staff is notified and task is scheduled. |
| Data Used | Order ID, staff ID, task assignment info. |
| Trigger | Accountant wants to assign staff to fulfill an order. |

Table 4 : Assign Storage Staff to Prepare Order Description

1. *Use Case Name : Customer order and discuss with accountant*

|  |  |
| --- | --- |
| Actors | Customer, accountant |
| Description | This use case allows the Customer to create a new order and send it to the accountant for review. The accountant checks the order details and, if any modifications are needed, edits the order and resends it back to the Customer. The Customer then either accepts the modified order or rejects it. |
| Preconditions | Customer and accountant must be logged in and have access to order creation. |
| Flow of Events | 1. Customer selects the products to order.  2. Customer enters order details including quantity.  3. Customer submits the order.  4. Accountant reviews and accepts the order. |
| Alternative Flows | 1. Customer selects the products to order.  2. Customer enters order details including quantity.  3. Customer submits the order.  4. Accountant modifies the order and sends it back.  5. Customer accepts or rejects the modified order |
| Postconditions | the order is either accepted by the accountant and confirmed, or sent back to the Customer for confirmation after modifications. |
| Data Used | Product details, customer ID, quantity, order ID. |
| Trigger | User wants to place an order. |

Table 5 : Customer order and discuss with accountant

1. *Use Case Name : Sales Rep Create and Send Order*

|  |  |
| --- | --- |
| Actors | Sales Rep, accountant |
| Description | This use case allows the Sales Rep to create a new order for his customer and send it to the accountant for review. The accountant checks the order details and, if any modifications are needed, edits the order and resend it back to the Sales Rep. The Sales Rep then either accepts the modified order or rejects it. |
| Preconditions | Sales Rep and accountant must be logged in and have access to order creation. |
| Flow of Events | 1. Sales Rep selects the Customer. 2. Sales Rep selects the products to order. 3. Sales Rep enters order details including quantity. 4. Sales Rep submits the order. 5. Accountant reviews and accepts the order. |
| Alternative Flows | 1. Sales Rep selects the Customer. 2. Sales Rep selects the products to order. 3. Sales Rep enters order details including quantity. 4. Sales Rep submits the order. 5. Accountant modifies the order and sends it back. 6. Sales Rep accepts or rejects the modified order |
| Postconditions | the order is either accepted by the accountant and confirmed, or sent back to the Sales Rep for confirmation after modifications. |
| Data Used | Product details, customer ID, quantity, order ID. |
| Trigger | User wants to place an order. |

Table 6 : Sales Rep Create and Send Order

1. *Use Case Name : Edit Quantity of Product and Resend*

|  |  |
| --- | --- |
| Actors | Storage Staff |
| Description | This use case allows the storage staff to update the quantity of products in an order and resend it for approval. |
| Preconditions | There must be a previously received order with quantity issues. |
| Flow of Events | 1- storage staff opens the received order. 2- storage staff updates the quantity for one or more products. 3- storage staff resend the updated order to the accountant. |
| Alternative Flows | 1- storage staff opens the received order in offline mode. 2- storage staff updates the quantity for one or more products. 3- System save the modified order in mobile storge.  4- System resends the updated order to the accountant after be connected to WIFI. |
| Postconditions | Updated order is submitted for review. |
| Data Used | Order ID, product ID, updated quantity. |
| Trigger | storage staff wants to update and resend product quantities. |

Table 7 : Edit Quantity of Product and Resend Description

1. *Use Case Name: Manage Products*

|  |  |
| --- | --- |
| Actors | Accountant |
| Description | This use case allows the accountant to add, update, or delete product records in the system. |
| Preconditions | Accountant must be logged in with appropriate access rights. |
| Flow of Events | 1- Accountant opens the product management section. 2- Accountant chooses to add, update, or delete a product. 3- Accountant fills in or updates the product details. 4- System saves the changes and updates the inventory. |
| Alternative Flows | - If a required field is missing, the system prompts the accountant to fill it in. |
| Postconditions | Product records are updated in the system. |
| Data Used | Product ID, name, price, supplier ID, inventory data. |
| Trigger | Accountant needs to manage product data. |

Table 8 : Manage Products Description

1. *Use Case Name: Manage Mobile Accounts*

|  |  |
| --- | --- |
| Actors | Accountant |
| Description | This use case allows the accountant to manage user accounts on the mobile app, including adding, editing, or removing drivers and storage staff. |
| Preconditions | Accountant must be logged in with admin privileges. |
| Flow of Events | 1- Accountant accesses the user management section. 2- Accountant views list of mobile users. 3- Accountant adds, edits, or removes user accounts. 4- System saves changes and updates user access. |
| Alternative Flows | - If required fields are missing or invalid, system prompts the user to correct them. |
| Postconditions | Mobile account changes are saved and applied. |
| Data Used | User ID, role, username, password, status. |
| Trigger | Accountant wants to manage access for mobile users. |

Table 9 : Manage Mobile Accounts Description

1. *Use Case Name: Create and Send Stock-Out Orders*

|  |  |
| --- | --- |
| Actors | Accountant |
| Description | This use case allows the accountant to create and send a stock-out order to the storage staff for product delivery. |
| Pre-Conditions | Accountant must be logged in and the order must be approved. |
| Flow of Events | 1- Accountant opens the order management module. 2- Accountant creates a stock-out order and selects products and quantities. 3- Accountant assigns the order to storage staff. 4- System records and notifies the assigned staff. |
| Alternative Flows | - If product quantity is insufficient, the system alerts the accountant. |
| Post-Conditions | Stock-out order is created and assigned to storage staff. |
| Data Used | Order ID, product details, storage staff ID. |
| Trigger | Accountant wants to prepare products for delivery. |

Table 10 : Create and Send Stock-Out Orders Description

1. *Use Case Name: Assign and Send Delivery Driver to Order*

|  |  |
| --- | --- |
| Actors | Accountant |
| Description | This use case allows the accountant to assign a delivery driver to a prepared order and send them delivery instructions. |
| Preconditions | There must be a prepared order ready for delivery. |
| Flow of Events | 1- Accountant opens the list of prepared orders. 2- Accountant selects an available delivery driver. 3- System sends the delivery assignment and order details to the driver. |
| Alternative Flows | - If no drivers are available, the system alerts the accountant and places the order in a waiting queue. |
| Postconditions | Driver is assigned to the order and notified. |
| Data Used | Order ID, driver ID, delivery details. |
| Trigger | Accountant wants to dispatch an order for delivery. |

Table 11 : Assign and Send Delivery Driver to Order Description

1. *Use Case Name: add Payment*

|  |  |
| --- | --- |
| Actors | Accountant |
| Description | This use case allows the accountant to manage incoming and outgoing payments related to customer orders and supplier purchases. |
| Preconditions | Accountant must be logged in . |
| Flow of Events | 1- Accountant navigates to the payments section.  2- Accountant navigates to add new payment.  3- Accountant select the payer  3- Accountant add the payment information.  4- System saves the updates and adjusts payer balances. |
| Alternative Flows | 1- Accountant navigates to the payments section.  2- Accountant navigates to add new payment.  3- Accountant select the payer  3- Accountant add the check information.  4- System saves the updates and adjusts payer balances after the exchange check date is come. |
| Postconditions | Payments are recorded or updated in the system. |
| Data Used | Payment ID, order ID, amount, date, payer/payee info, check ID, bank name, bank branches, check image, exchange date, exchange rate. |
| Trigger | Accountant selects "Add New Payment" from the payments section. |

Table 12 : add Payment Description

### Use Cases Diagram

**Figure 11:** represents the overall use‐case diagram that captures how seven actors—the Accountant, Customer, Sales Rep, Storage Staff, Supplier, Delivery Driver, and GPS System—engage with the Dolphin Stock System to carry out product management, order placement, stock-in/out workflows, payment processing, and delivery routing from end to end.

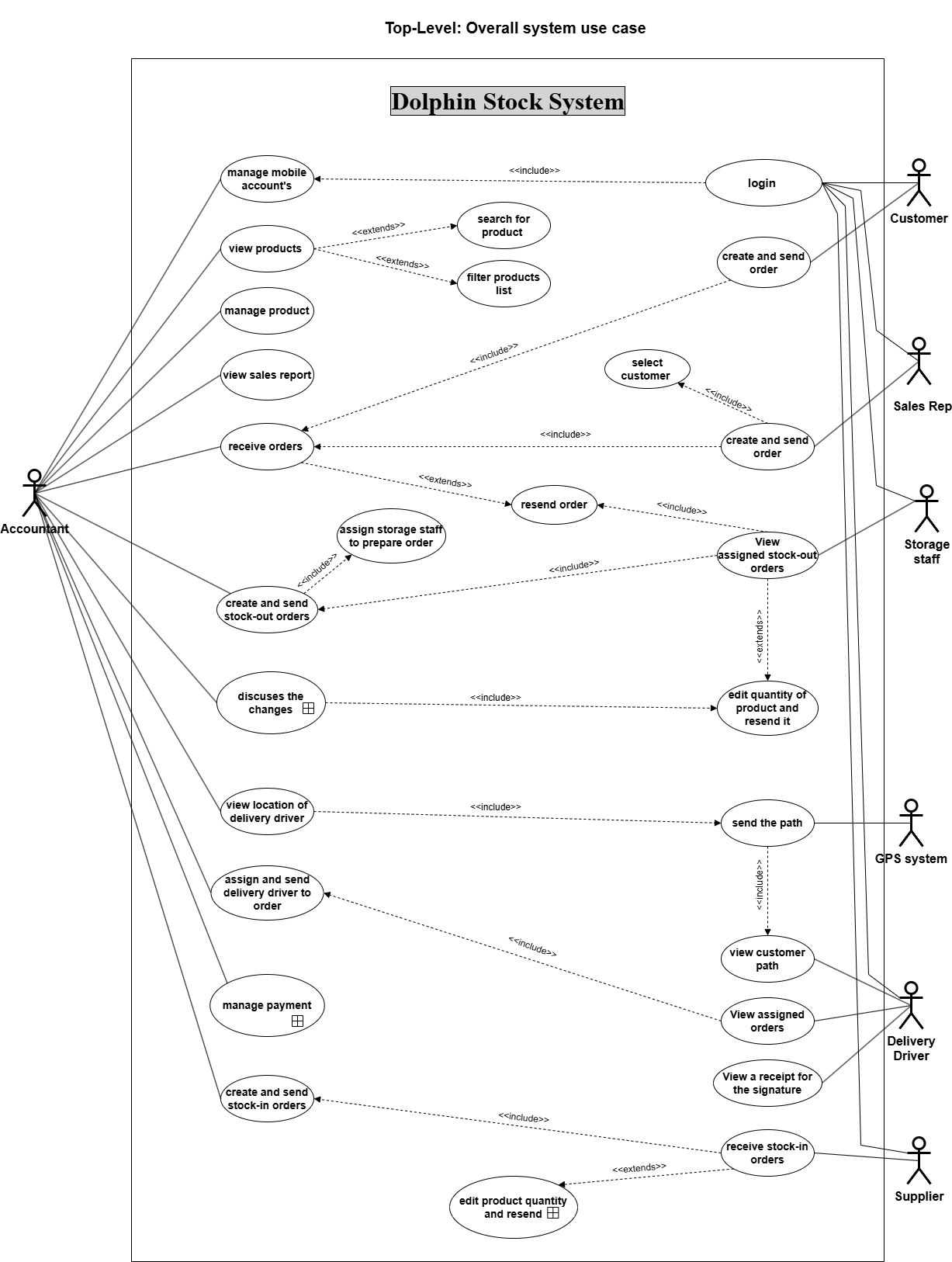


Figure 10 : Overall Use Cases Diagram

**Figure 12:** represents the 1st-level “Manage Payment” use case that captures how the accountant inserts receipt or voucher payments, selects the paying Customer or payee Supplier, chooses the payment method (cash or check), and updates check status.

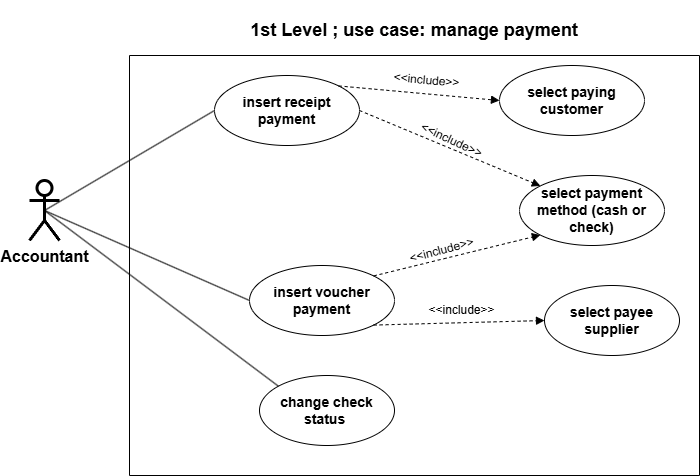


Figure 11 : 1st Level Mange payment

**Figure 13:** represents the 1st-level “Edit Product Quantity and Resend” use case that captures how the accountant views stock-out order changes and then either extends the flow to the Supplier to edit quantities and resend or rejects the order.

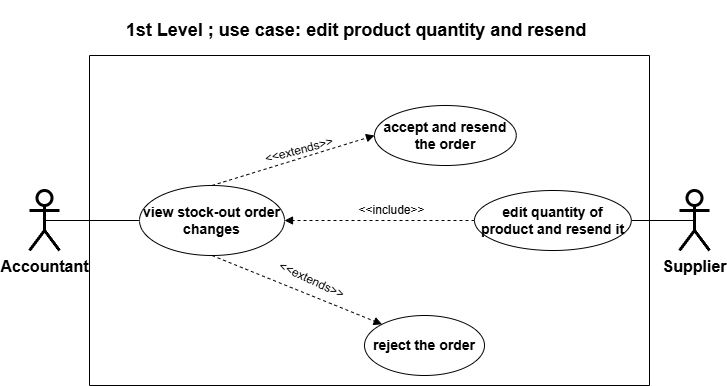


Figure 12 : 1st level Edit Product quantity and resend

**Figure 14:** represents the 1st-level “Discusses the Changes” use case that captures how the accountant inspects order modifications and, after consulting with the order sender (Storage Staff), decides to accept and resend the order or to reject it.

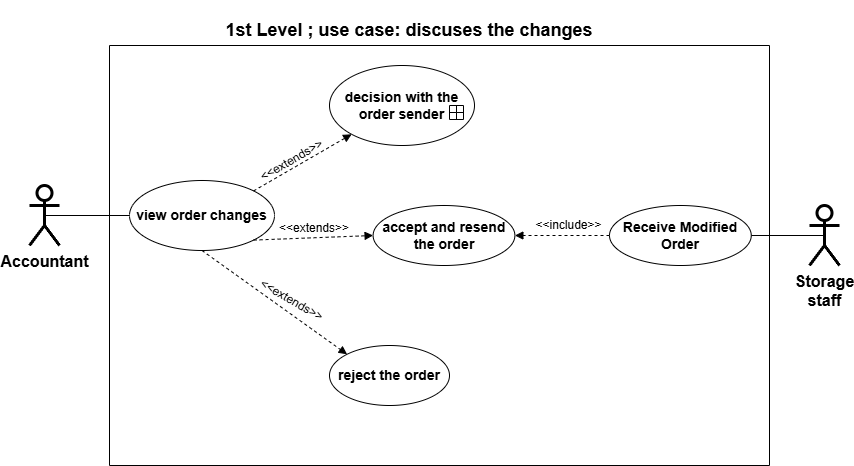


Figure 13: Discuses the changes

**Figure 15:** represents the 2nd-level “Decision with the Order Sender” use case that captures how the Accountant and Storage Staff interact with the Customer or Sales Rep to receive a modified order and then accept/resend or reject it.

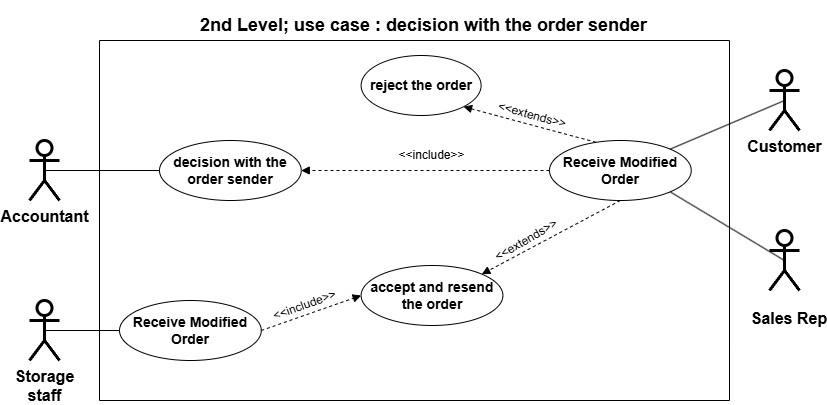


Figure 14 : decision with the order sender

## System Models

### Class Diagram

**Figure 16:** represents the static class diagram of the Dolphin Stock System It shows how domain objects, transnational records, organizational roles, and user accounts are organized and interconnected to support inventory management, order processing, payment workflows, and user interactions throughout the system.

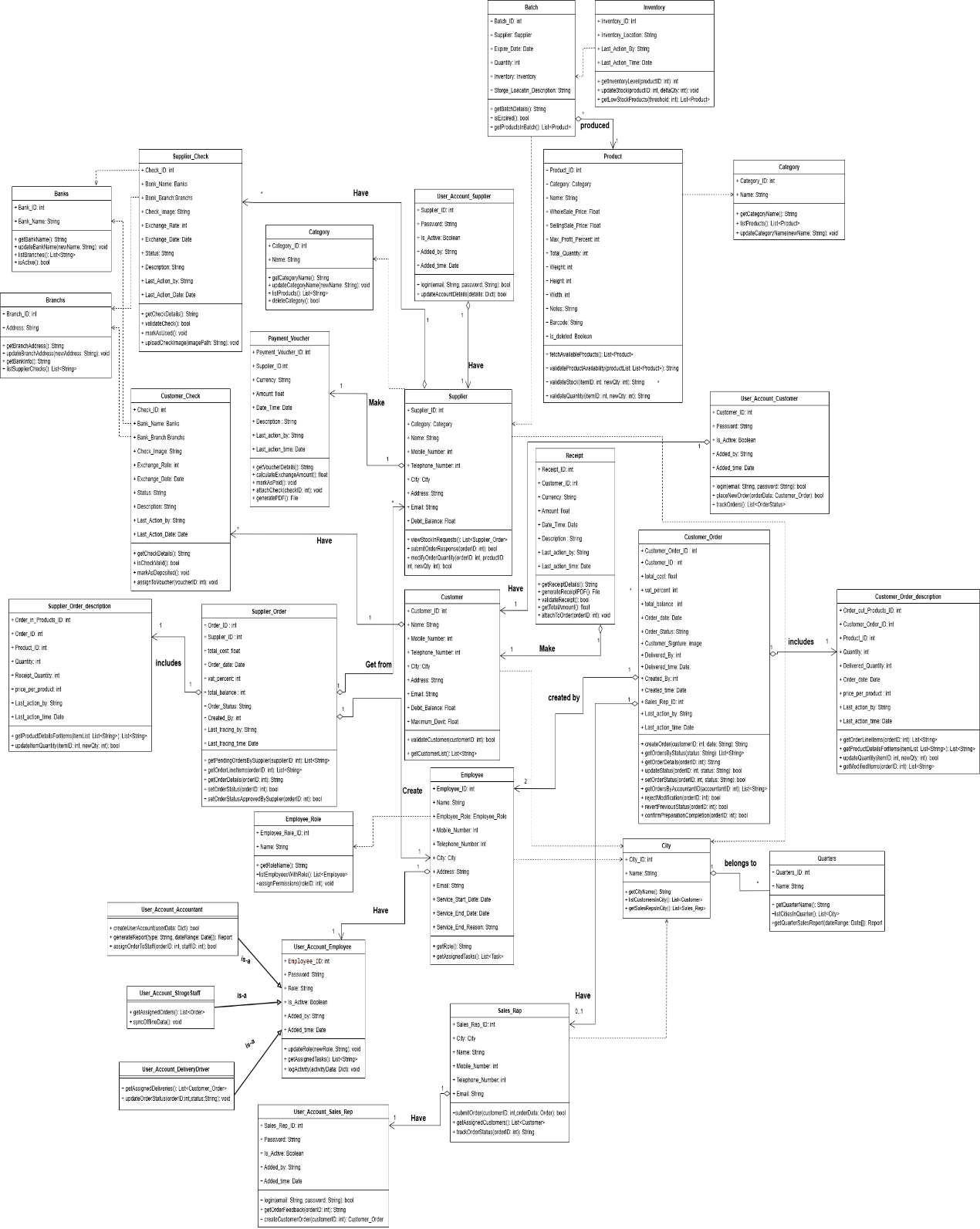


Figure 15 : Class Diagram

### Sequence Diagram

#### Create and Send Order (Customer) Sequence

**Figure 17:** is a sequence diagram for “Create and Send Order (Customer)”, that shows the order-handling parts of the system, and the accountant work together to submit an order and then either approve it, change it, or reject it.

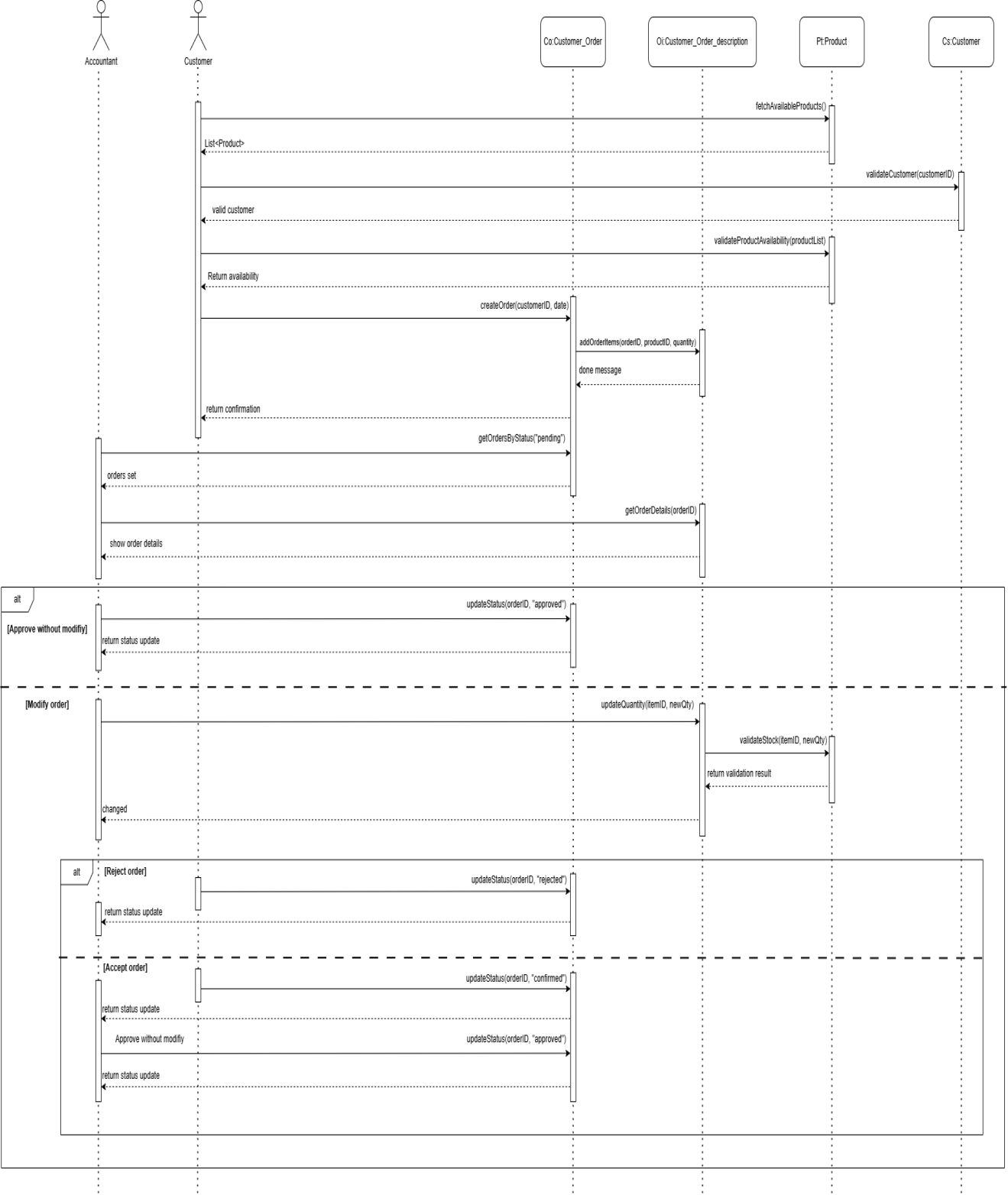


Figure 16 :Create and Send Order (Customer) Sequence Diagram

#### Create and Send Order (Sales Representative) Sequence

**Figure 18:** is a sequence diagram for “Create and Send Order (Sales Representative)” that shows the steps between the Sales Rep, the order-handling modules, and the accountant to create an order and then approve it, modify it, or reject it.



Figure 17 : Create and Send Order (Sales Representative) Sequence Diagram

#### Supplier Receives and Responds Stock-In Order Sequence

**Figure 19:** is a sequence diagram for “Supplier Receives and Responds Stock-In Order”, that shows the stock-in order components, and the accountant interact to fetch pending orders, review and adjust item quantities, and update the order status (approved or rejected).

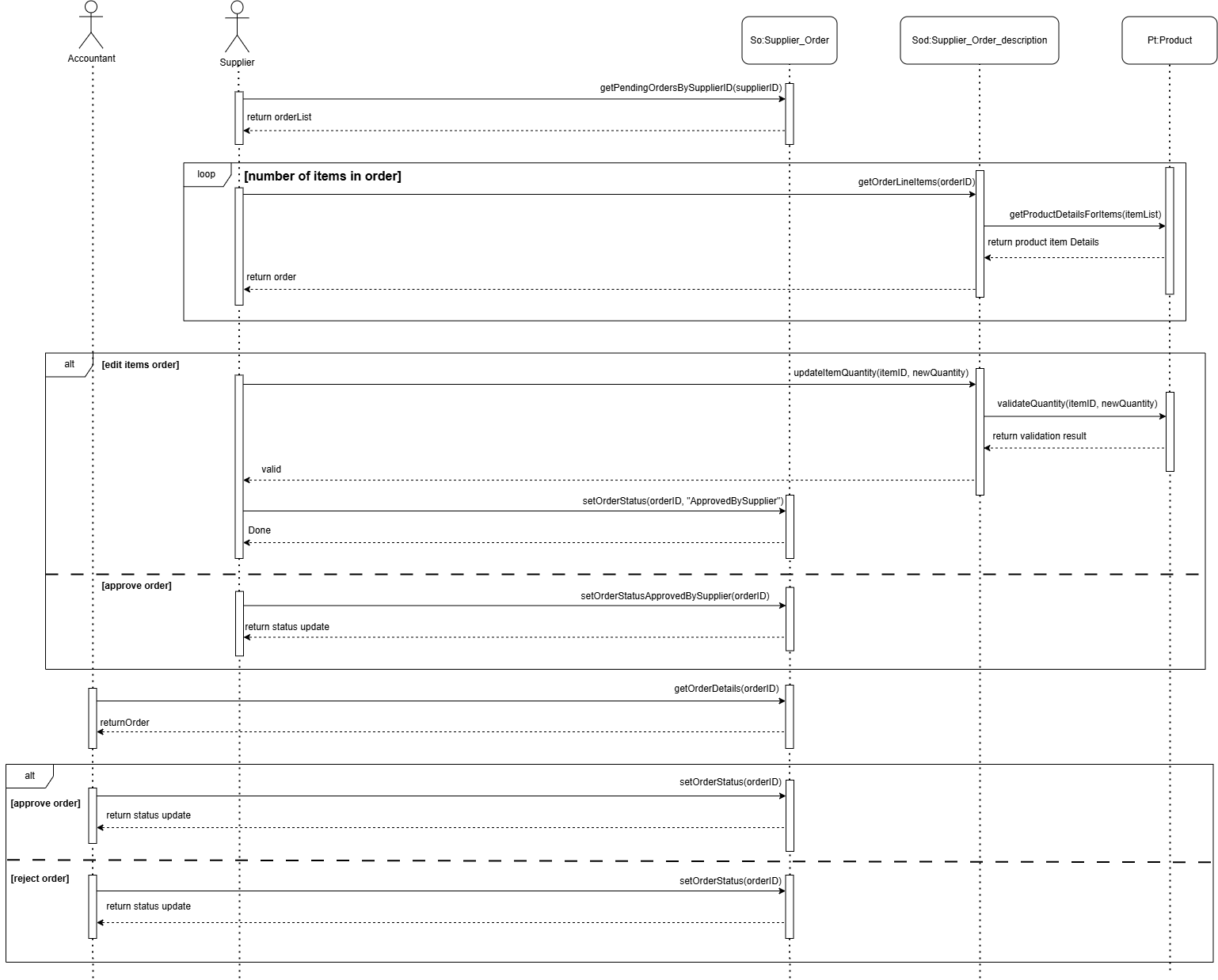


Figure 18 : Supplier Receives and Responds Stock-In Order Sequence Diagram

#### Storage Staff Handles Stock-Out Order Sequence

**Figure 20:** is a sequence diagram for “Storage Staff Handles Stock-Out Order” that shows the order-handling modules, and the accountant work together to view assigned orders, accept or reject them, adjust quantities during preparation, and notify when the order is ready for delivery.

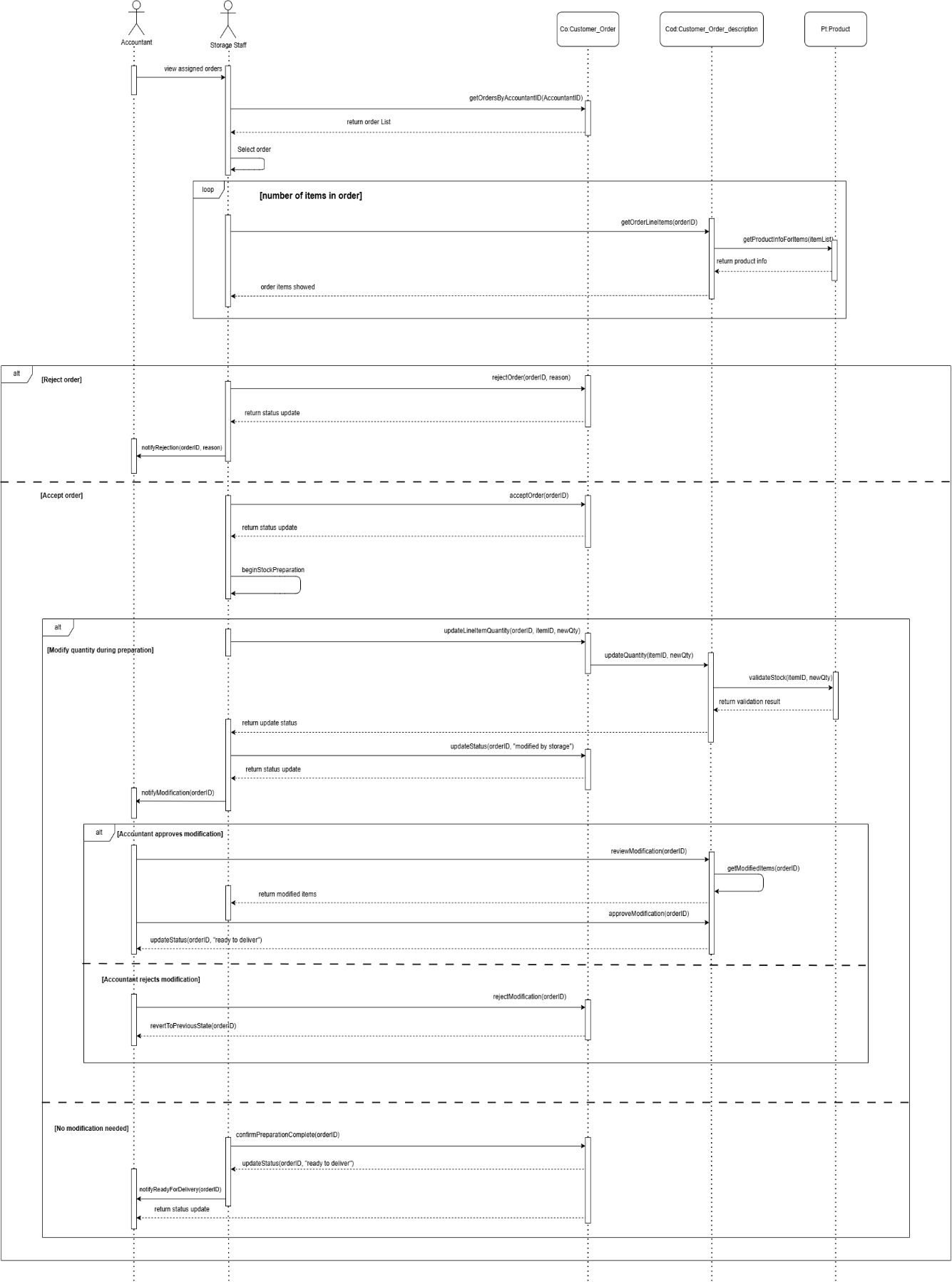


Figure 19 : Storage Staff Handles Stock-Out Order Sequence Diagram

### Activity Diagram

#### Create and Send Order (Customer) Activity

**Figure 21:** is an activity diagram for “Create and Send Order (Customer)” that show the flow of actions across the Customer and Accountant, from creating and sending the order, through optional quantity updates and resending, to the final decision point.

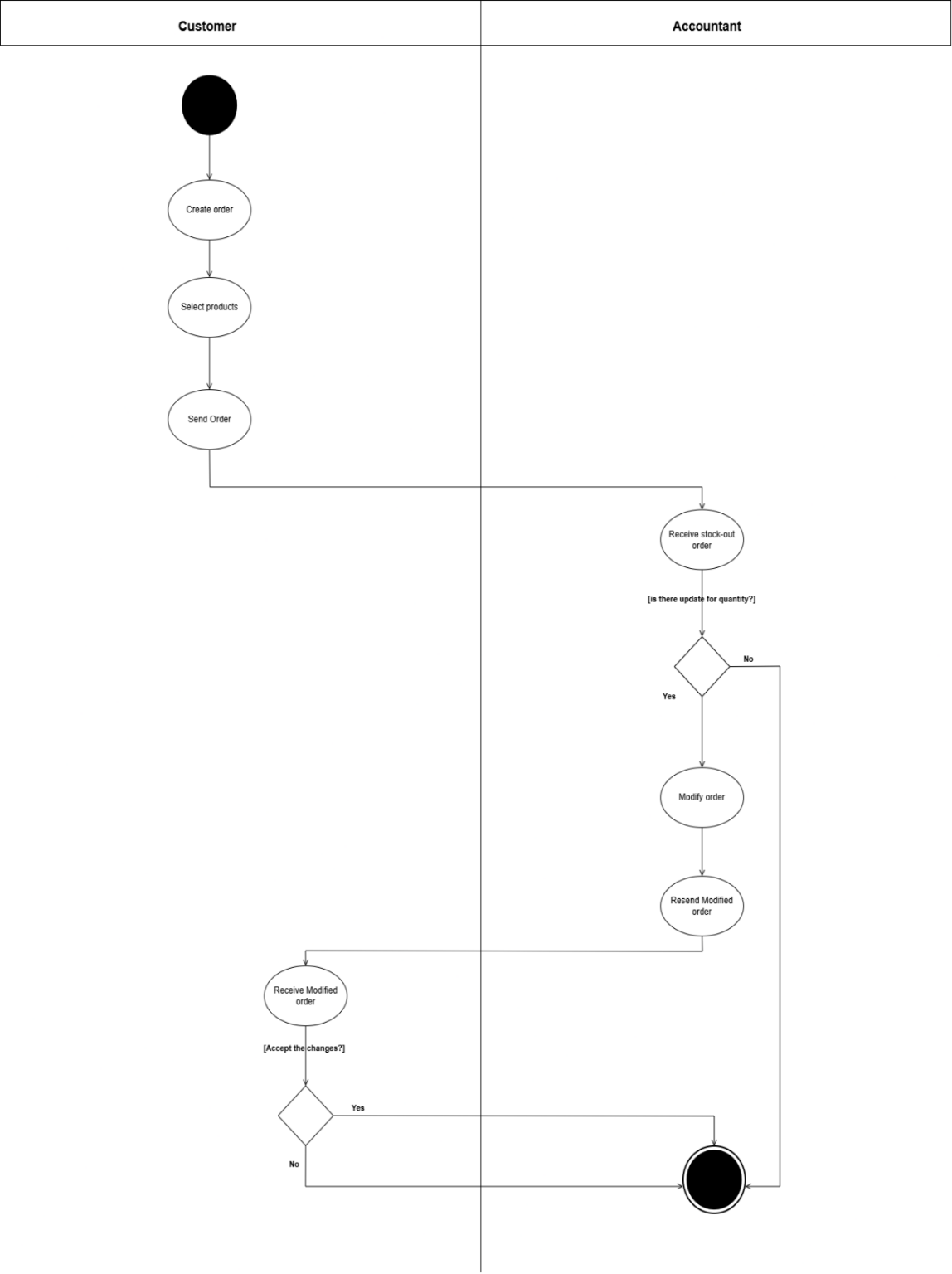


Figure 20 : Create and Send Order (Customer) Activity Diagram

#### Create and Send Order (Sales Representative) Activity

**Figure 22:** is an activity diagram for “Create and Send Order (Sales Representative)” that show the flow of actions across the Sales Representative and Accountant, from selecting the customer, creating and sending the order, through optional quantity updates and resending, to the final decision point.

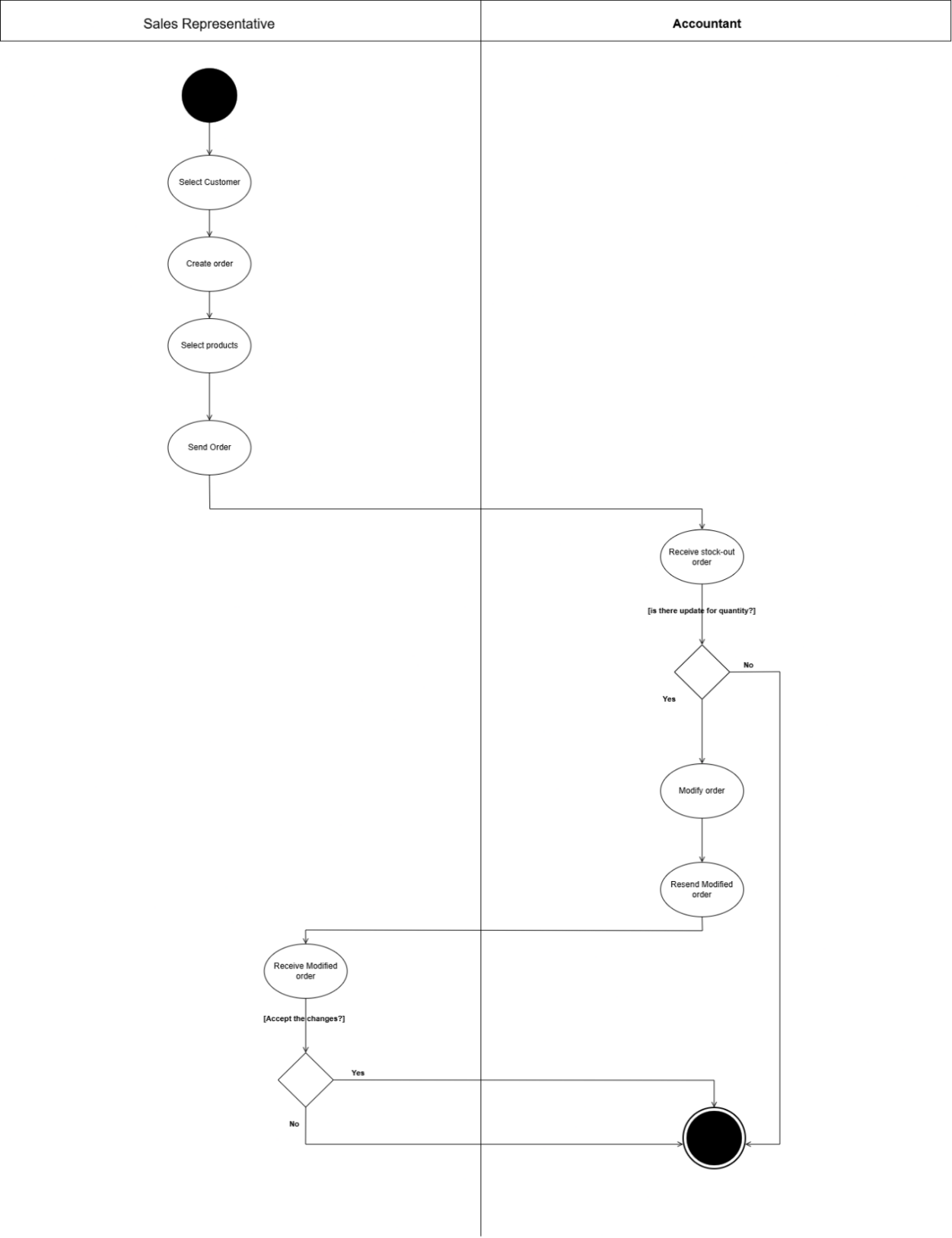


Figure 21 : Create and Send Order (Sales Representative) Activity Diagram

#### Supplier Receives and Responds Stock-In Order Activity

**Figure 23:** is an activity diagram for “Supplier Receives and Responds Stock-In Order” that show the flow of actions across the Accountant and Supplier, from creating and sending the stock-in order, through optional quantity updates and resending by the Supplier, to the final decision point.

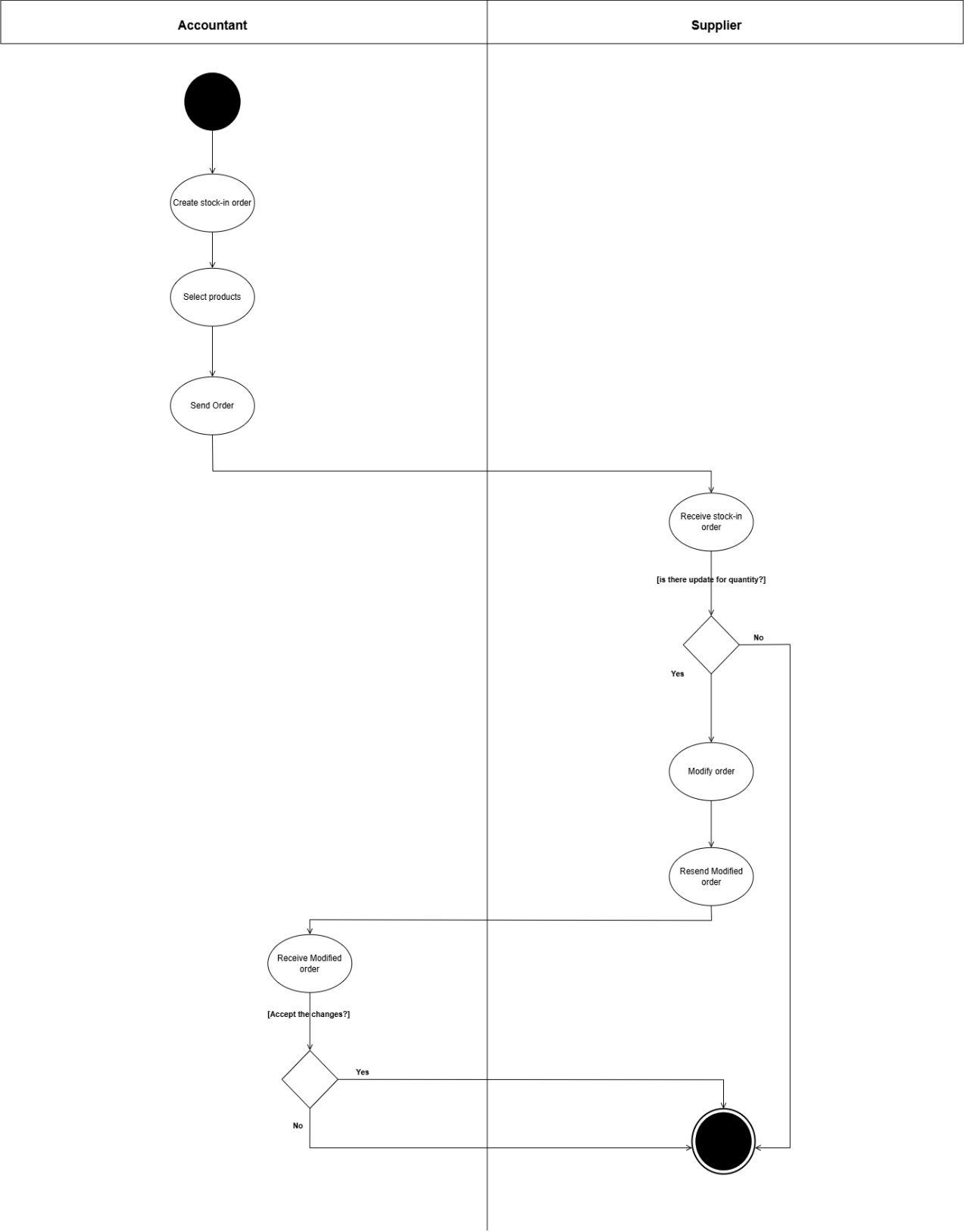


Figure 22 : Supplier Receives and Responds Stock-In Order Activity Diagram

#### Storage Staff Handles Stock-Out Order Activity

Figure 24: is an activity diagram for “Storage Staff Handles Stock-Out Order” that show the flow of actions across the Storage Staff and Accountant, from the Storage Staff receiving and reviewing a stock-out order, through optional quantity changes and resending, to the accountant’s discussion with the sender.

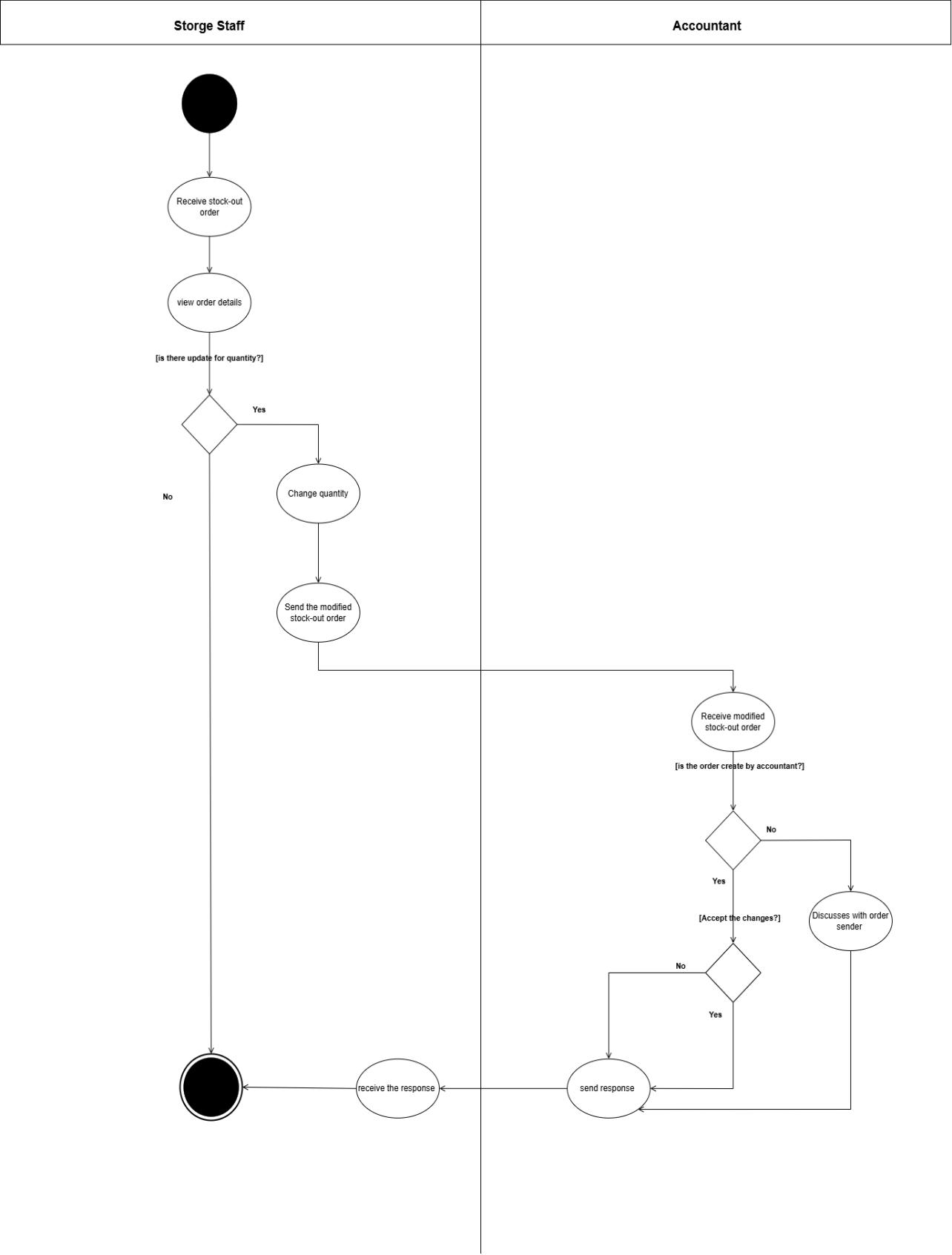


Figure 23 : Storage Staff Handles Stock-Out Order Activity Diagram

### State Chart Diagram

#### Create and Send Order (Customer) State Chart

**Figure 25:** is a state-chart diagram for “Create and Send Order (Customer)” that show the lifecycle of a customer order, from its initial creation and submission, through pending and modification loops, to its final approved, completed, or rejected states.

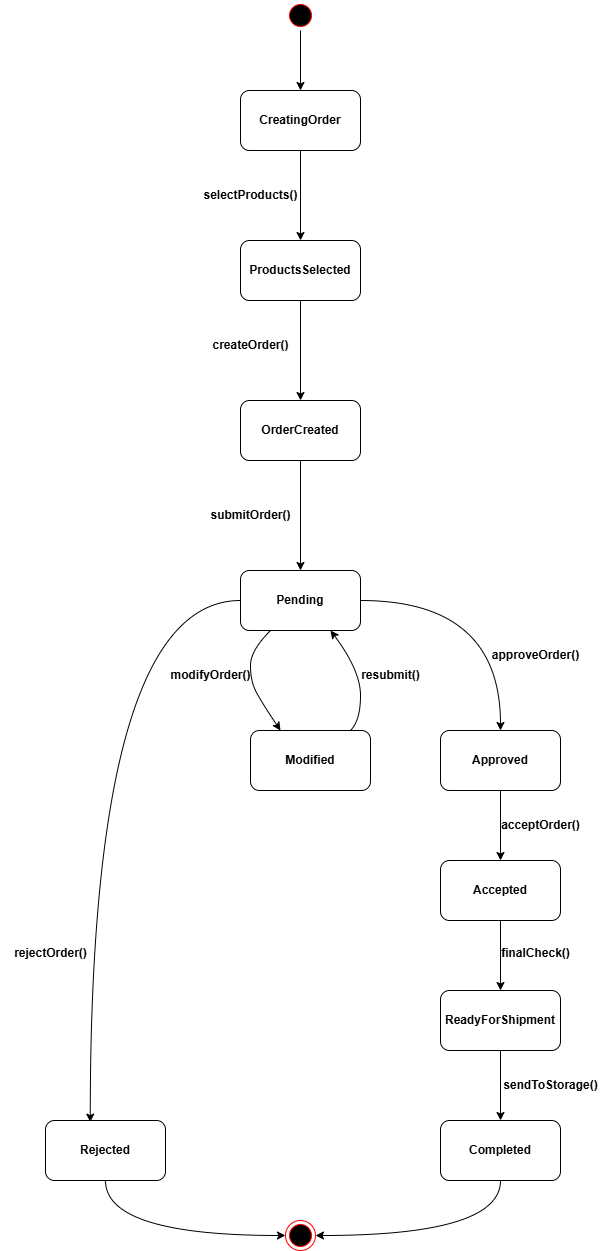


Figure 24 : Create and Send Order (Customer) State Chart Diagram

#### Create and Send Order (Sales Representative) State Chart

**Figure 26:** is a state-chart diagram for “Create and Send Order (Sales Representative)” that show the lifecycle of a sales-rep-initiated order, from its initial creation (customer and product selection), through pending review and any modification loops, to its final acceptance and readiness for storage or outright rejection.

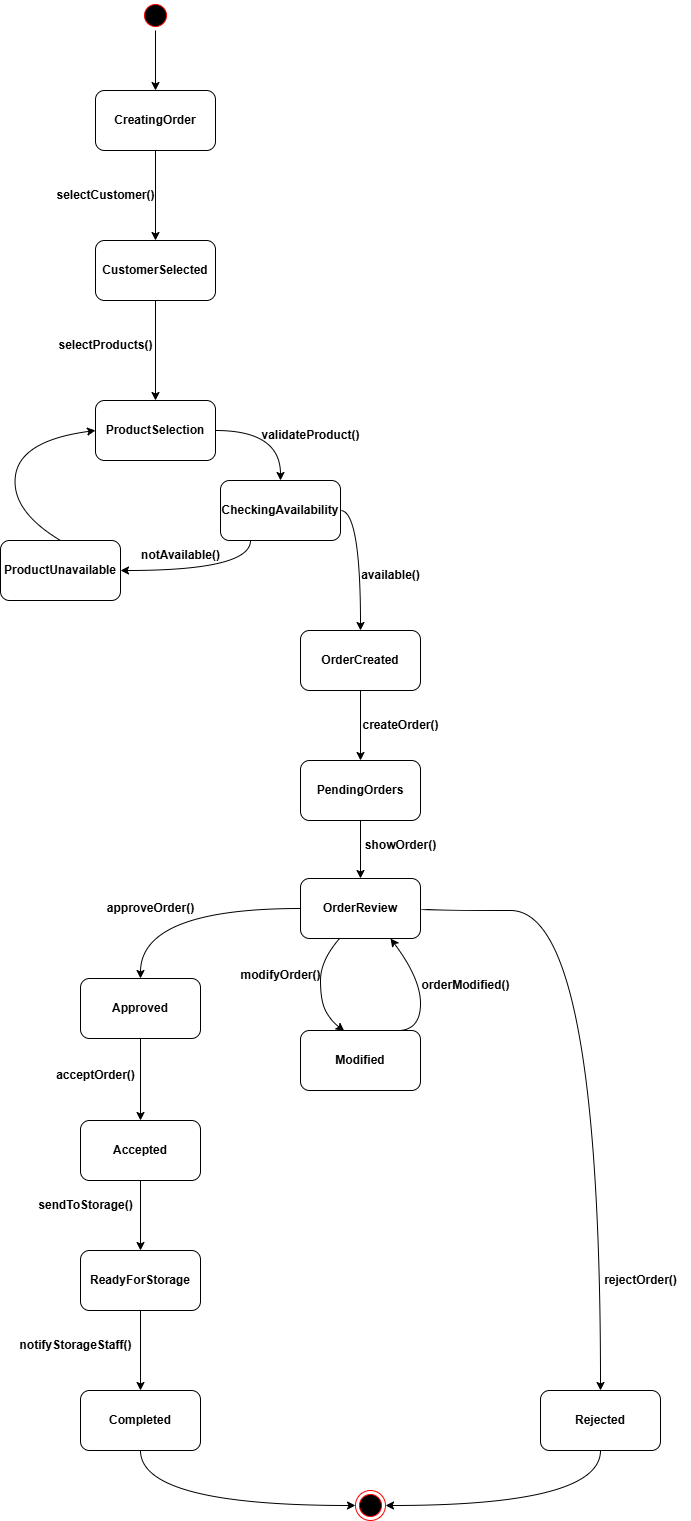


Figure 25 : Create and Send Order (Sales Representative) State Chart Diagram

#### Supplier Receives and Responds Stock-In Order State Chart

**Figure 27:** is a state-chart diagram for “Supplier Receives and Responds Stock-In Order” that show the lifecycle of a stock-in order, from its arrival in the Pending Orders state, through item editing and resubmission loops by the Supplier, to its final Approved (Completed) or Rejected state.

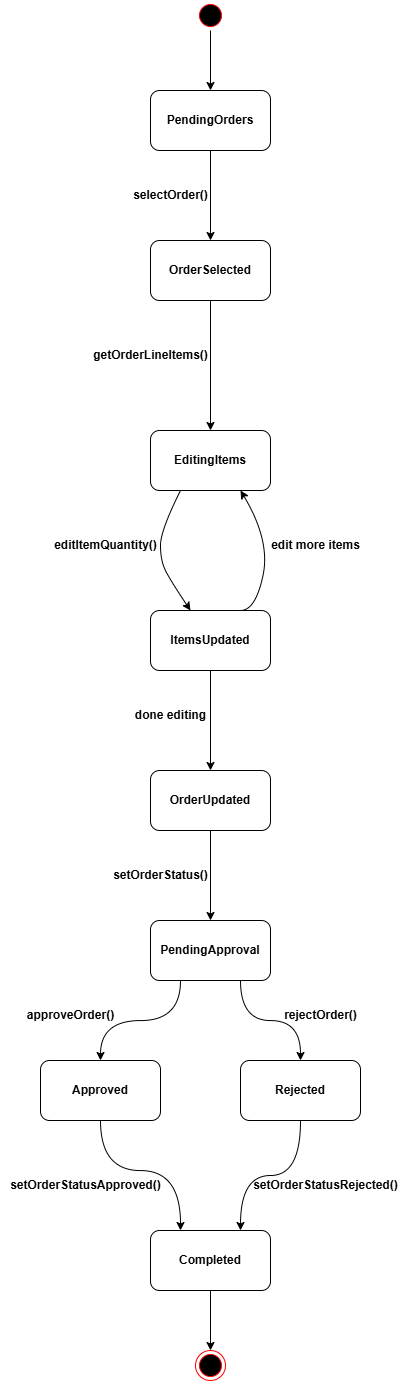


Figure 26 : Supplier Receives and Responds Stock-In Order State Chart Diagram

#### Storage Staff Handles Stock-Out Order State Chart

**Figure 28:** is a state-chart diagram for “Storage Staff Handles Stock-Out Order” that show the lifecycle of a stock-out order, from its receipt and optional quantity modifications by storage staff, through the accountant’s review and approval or rejection, to its final ready-for-delivery or completed state.

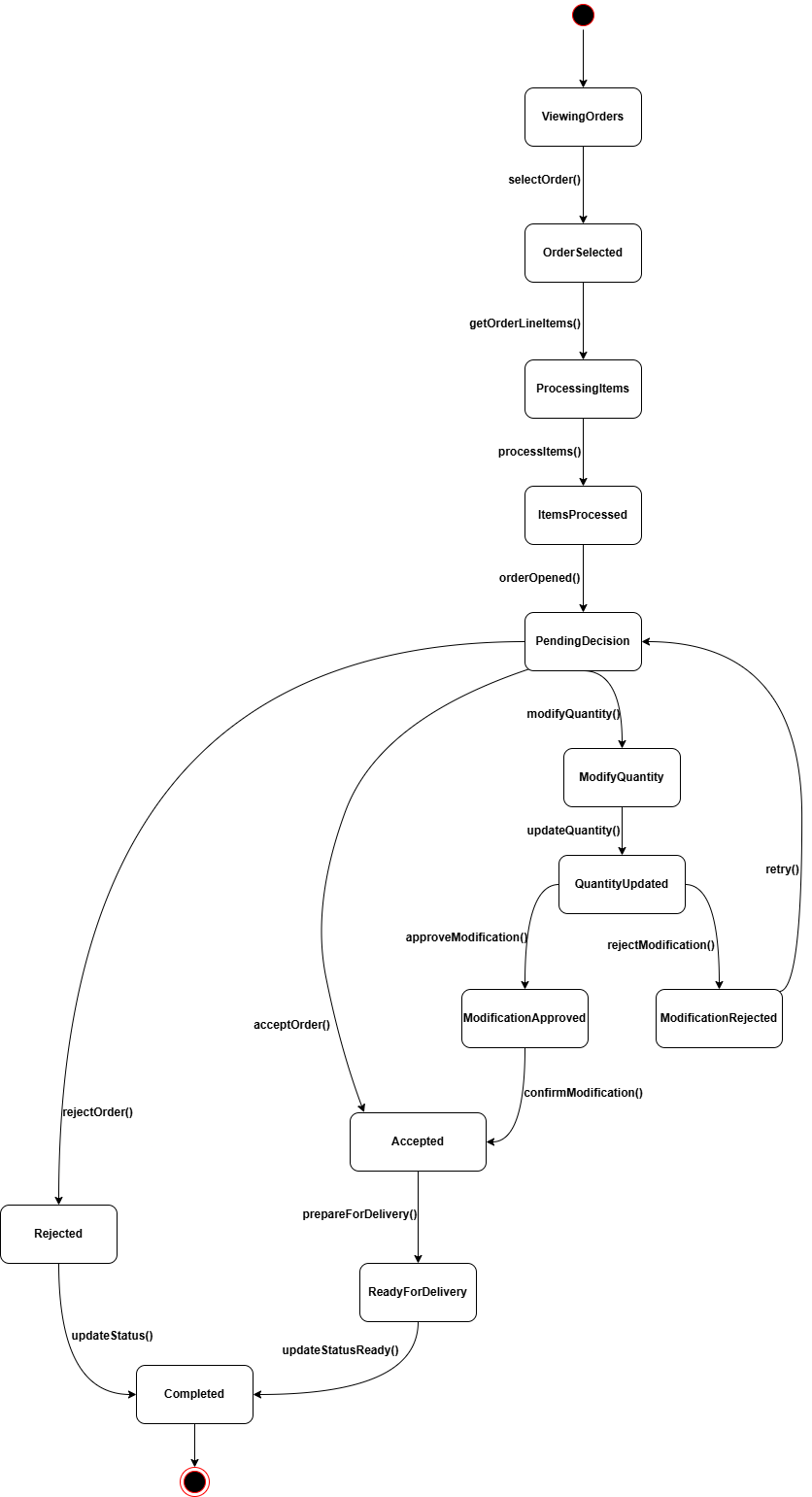


Figure 27 : Storage Staff Handles Stock-Out Order State Chart Diagram

## System Architecture

### Sub-System

1. **Stock-in Order Management System-Sub**

create a stock request, which is reviewed by the warehouse manager. The manager can approve, reject, or modify the request based on current inventory levels. Once approved, the items are added to the system and reflected in the available stock

1. **Stock-out Order Management Sub-System**

This subsystem handles customer orders leaving the inventory. The accountant initiates the request, which is sent to the warehouse for approval. The warehouse manager checks stock availability, modifies if needed, and assigns a worker to prepare the order. A notification is sent to the accountant once the order is ready

1. **Product Sub-System**

The product subsystem maintains detailed information about all items in the inventory. It includes product names, quantities, batches, and descriptions. It also ensures automatic updates to stock when items are added or removed through orders

1. **Account Management Sub-System**

This subsystem handles user accounts and roles such as accountants, warehouse managers, and workers. Each user has a specific set of permissions depending on their role. The subsystem ensures that only authorized users can access relevant parts of the system

1. **Payment Management Sub-System**

This subsystem manages the financial operations related to orders. It handles the creation of payment vouchers, receipts, and check records. Each payment is linked to a specific order

### Software Architecture

**Figure 29:** is a software architecture diagram for the Dolphin Stock System, that show its three layer design with a UI layer for user interaction, a middle layer hosting all business‐logic modules, and an infrastructure layer handling persistence and security.

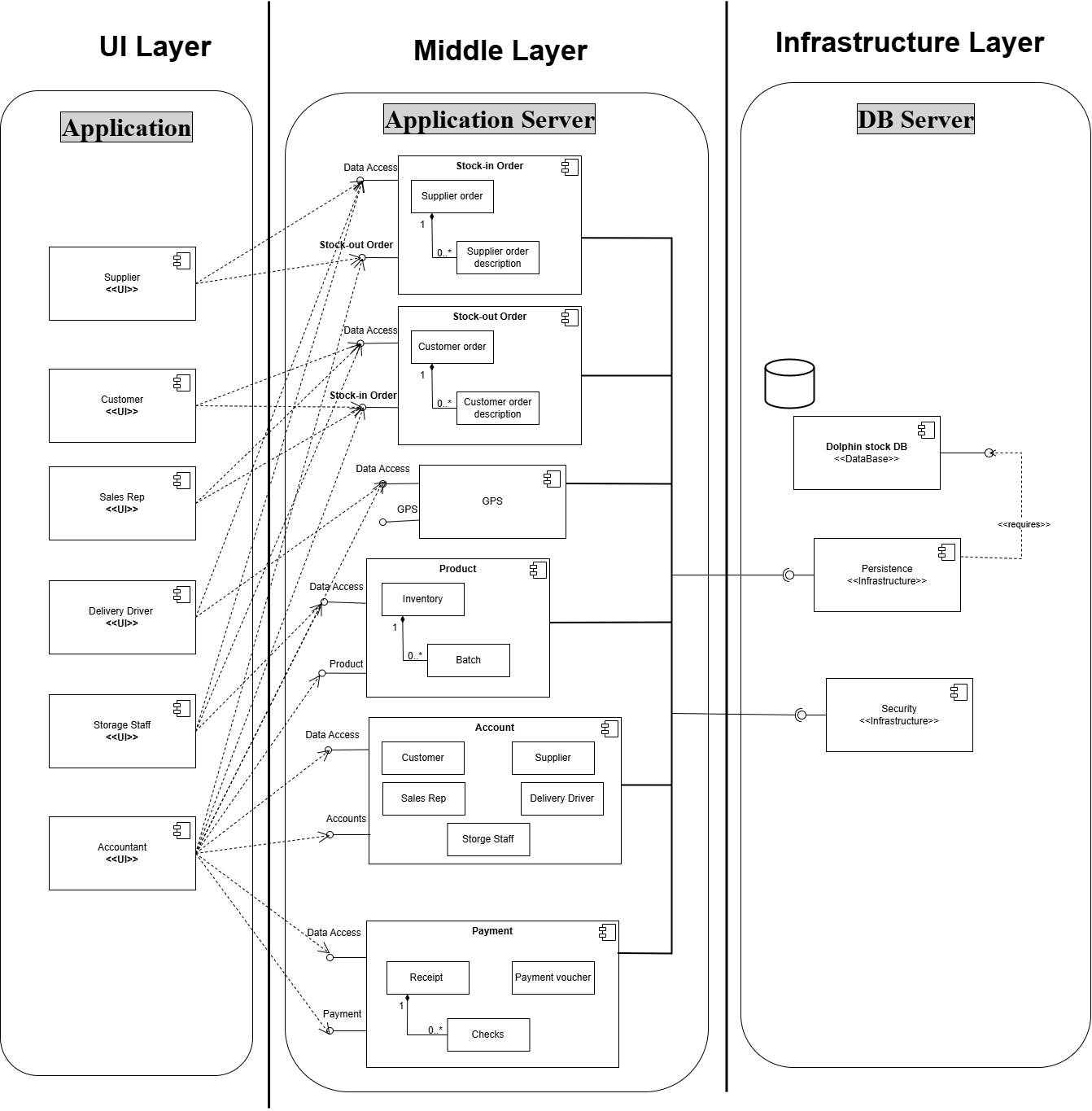


Figure 28 : Software Architecture Diagram

### Deployment Diagram

**Figure 30:** is a deployment diagram for the Dolphin Stock System that show how user‐facing UI components are deployed to the application server modules and how those modules connect to backend infrastructure nodes for the Dolphin Stock database (persistence) and the security services.

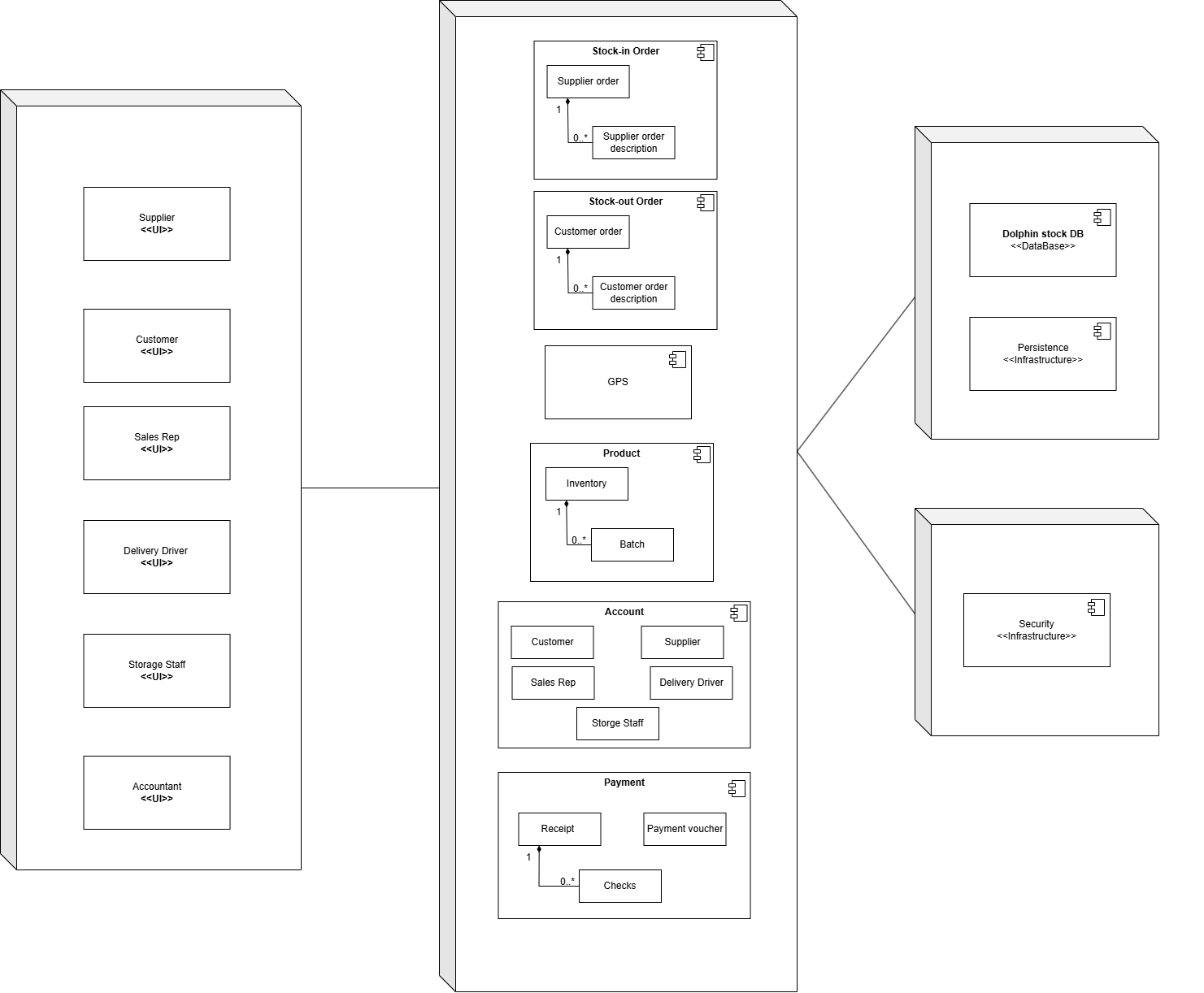


Figure 29 : Deployment Diagram

### Component Diagram

**Figure 31:** is a component diagram for the Dolphin Stock System that show how each actor’s UI component (Supplier, Customer, Sales Rep, Delivery Driver, Storage Staff, Accountant) depends on and connects to the core backend components—Stock-in Order, Stock-out Order, GPS, Product, Account, and Payment—and how those components in turn rely on the infrastructure services for persistence (Dolphin stock DB) and security.

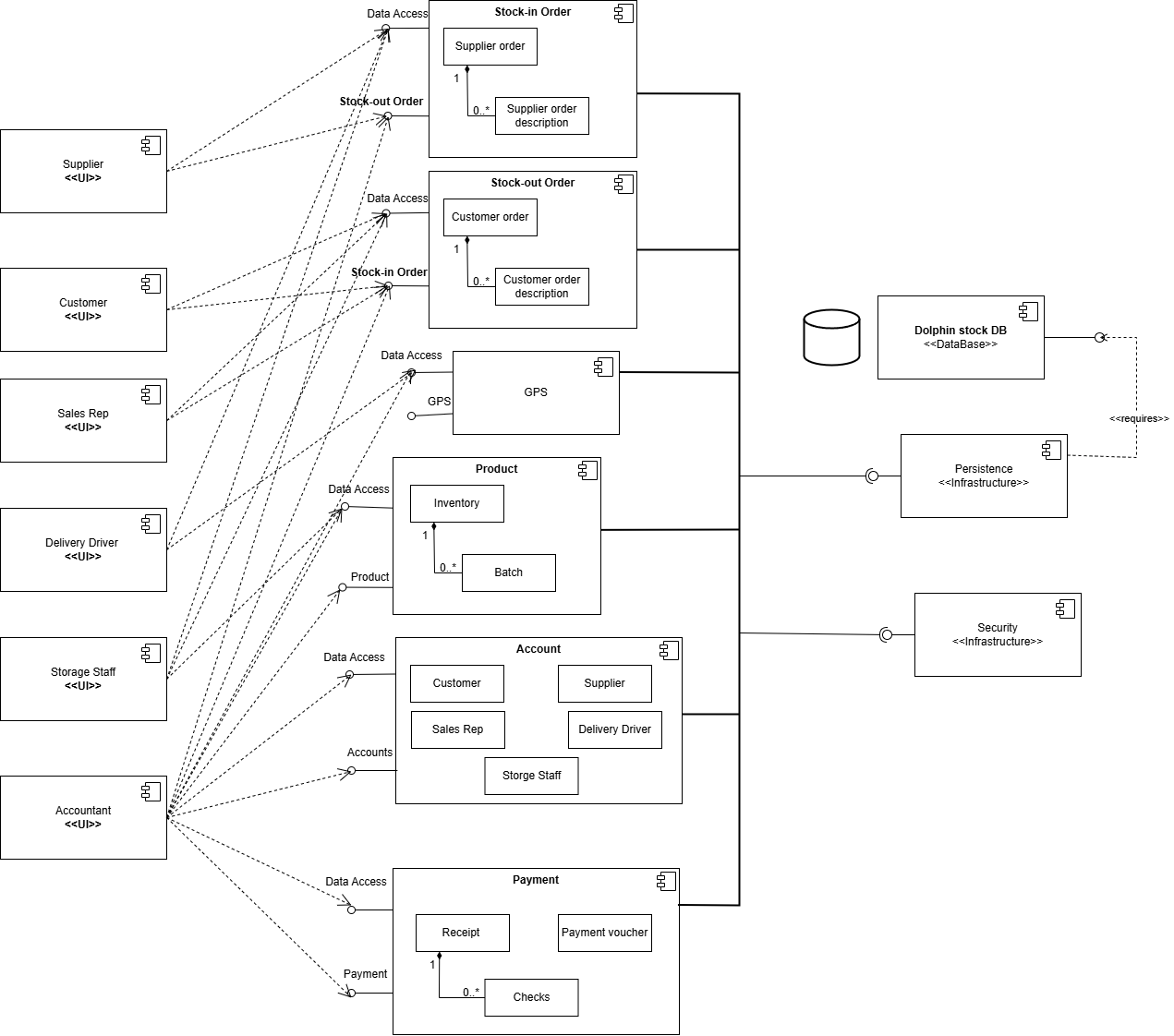


Figure 30 :Component Diagram

## Data Management and ERD

**Figure 32:** is a Data Management and ERD diagram that lays out how the system’s data is organized into tables—such as Customers, Suppliers, Orders (and their line items), Products, Inventory, Payments, and User Accounts—and shows the links (foreign key relationships) between them, ensuring the database can enforce data consistency and support all of the application’s workflows.



Figure 31 : Data Management and ERD Diagram

# Chapter 4: Implementation

# Chapter 5: Evaluation and Testing

# Chapter 6: Conclusion and Future works

## Review of the project

The **Dolphin Stock System** successfully addresses the key challenges faced by businesses that rely on manual, paper-based inventory and sales processes. Through the integration of a web interface for accountants and a mobile application for storage staff, delivery drivers, suppliers, customers, and sales representatives, the system offers a complete digital solution for managing orders, tracking orders, handling payments, and monitoring deliveries.

One of the standout features of the system is its offline functionality for storage staff, which ensures uninterrupted operations even in areas with poor internet connectivity. The use of GPS and Google Maps integration provides real-time delivery tracking, enhancing coordination and transparency. Additionally, role-based interfaces and multi-level order approval flows ensure that every actor interacts with the system in a way that reflects their responsibilities.

Overall, the Dolphin Stock System achieves its goal of improving efficiency, reducing human error, and supporting informed decision-making through real-time access to data and well-structured workflows.

## Next Semester Plan

|  |  |  |
| --- | --- | --- |
| Semester | *Works* | *Due date* |
| **First Semester** | Select the project idea | Mar-2025 |
| Describe our project, determine its objectives and methods, and read relevant literature. | Mar & Apr-2025 |
| System analysis, write requirements and draw system design | May-2025 |
| Write the final Report | Jun-2025 |
| **Second Semester** | Start learning Flutter for website and mobile | Jul-2025 |
| Building the database | Aug-2025 |
| Build the Front-end for website | Aug & Sep-2025 |
| Build the Front-end for Mobile App | Sep & Oct-2025 |
| Build the Back-end for website & mobile app | Nov & Dec-2025 |
| Write the final graduation project report | Jan-2026 |

Table 13 : Next Semester Plan

## Conclusion

The **Dolphin Stock System** successfully addresses the operational challenges faced by dolphin company that rely on manual methods to manage inventory, sales, and deliveries. By digitizing the entire workflow, the system ensures faster processing, real-time tracking, and improved coordination between departments such as accounting, Storage management and sales .

The integration of a web interface for accountants and a mobile application for Storage staff, suppliers, customers, sales representatives, and delivery drivers allows each role to work efficiently using tools specifically designed for their responsibilities. Features like offline mode for Storage staff, GPS tracking for delivery drivers, digital delivery confirmations, and automated reporting contribute to a streamlined and transparent business environment.

With a secure, role-based access model and real-time synchronization between mobile and web platforms, the **Dolphin Stock System** enhances decision-making, reduces errors, and improves customer satisfaction. Overall, the project meets its goals of creating a scalable, efficient, and user-friendly solution that aligns with the modern needs of dolphin company inventory and sales management.

# Bibliography

|  |  |
| --- | --- |
| [1] | [A Review of Existing Inventory Management Systems](https://www.researchgate.net/publication/383947700_A_Review_of_Existing_Inventory_Management_Systems) |
| [2] | [Sales and Stock Management System](https://scholarworks.lib.csusb.edu/cgi/viewcontent.cgi?article=2964&context=etd) |
| [3] | [SALES AND INVENTORY MANAGEMENT SYSTEM for RAHMAT BEE ABDUL ALEEM](https://utpedia.utp.edu.my/id/eprint/13591/1/Rahmet%20Bee_13743.pdf) |
| [4] | [Flutter\_(software)](https://en.wikipedia.org/wiki/Flutter_(software)) |
| [5] | [My SQL](https://en.wikipedia.org/wiki/MySQL) |
| [6] | [Dart (programming language)](https://en.wikipedia.org/wiki/Dart_(programming_language)) |
| [7] | [PHP](https://en.wikipedia.org/wiki/PHP) |
| [8] | [Intelisoft](https://intellisoft.io/implementing-offline-data-synchronization-in-logistics-applications/) |
| [9] | [Tech science press](https://www.techscience.com/cmc/v75n3/52630/html) |

# Appendices

## Use Case Specifications

1. *Use Case Name: View Products*

|  |  |
| --- | --- |
| Actors | Accountant |
| Description | This use case allows the accountant to view all products currently available in the inventory. |
| Preconditions | Accountant must be logged in. |
| Flow of Events | 1- Accountant opens the product management section. 2- System displays a list of products with details. 3- Accountant can scroll, filter, or search the list. |
| Alternative Flows | - If no products are available, system displays an empty list message. |
| Postconditions | Product list is displayed for review or further action. |
| Data Used | Product ID, name, category, quantity, price. |
| Trigger | Accountant wants to check product availability or information. |

Table 14 : View Products

1. *Use Case Name: View Sales Report*

|  |  |
| --- | --- |
| Actors | Accountant |
| Description | This use case allows the accountant to view summarized reports of sales activity. |
| Preconditions | Accountant must be logged in. |
| Flow of Events | 1- Accountant navigates to the reports section. 2- Accountant selects a report type (e.g., daily, weekly, monthly). 3- System generates the report and displays it. |
| Alternative Flows | - If there is no data available for the selected period, the system displays a message. |
| Postconditions | Sales report is displayed for the accountant. |
| Data Used | Sales data, report filters (date range, customer, product). |
| Trigger | Accountant needs to review sales performance. |

Table 15 : View Sales Report

1. *Use Case Name: Discuss the Changes*

|  |  |
| --- | --- |
| Actors | Accountant |
| Description | This use case allows the accountant to review and respond to change requests from warehouse staff regarding existing orders. |
| Preconditions | There must be a submitted change request from staff. |
| Flow of Events | 1- Accountant receives a notification for a change request. 2- Accountant opens the request and reviews the proposed changes. 3- Accountant accepts or rejects the changes. 4- System updates the order accordingly and notifies the staff. |
| Alternative Flows | - If the request is unclear, the accountant can send it back for clarification. |
| Postconditions | Order is updated or returned based on the accountant's decision. |
| Data Used | Order ID, change request details. |
| Trigger | Warehouse staff submits a request to change an order. |

Table 16 : Discuss the Changes

1. *Use Case Name: View Location of Delivery Driver*

|  |  |
| --- | --- |
| Actors | Accountant |
| Description | This use case allows the accountant to track the real-time location of delivery drivers using GPS integration. |
| Preconditions | Delivery drivers must be actively sharing their GPS location. |
| Flow of Events | 1- Accountant opens the delivery tracking interface. 2- System displays a map with current locations of active delivery drivers. 3- Accountant can click on each driver to view more details. |
| Alternative Flows | - If the GPS signal is lost, the system displays the last known location. |
| Postconditions | Accountant sees the live location of delivery drivers. |
| Data Used | Driver ID, GPS coordinates, delivery assignment. |
| Trigger | Accountant wants to monitor delivery progress. |

Table 17 : View Location of Delivery Driver

1. *Use Case Name: Resend Order*

|  |  |
| --- | --- |
| Actors | Accountant |
| Description | This use case allows the accountant to make corrections to an existing order and resend it for processing. |
| Preconditions | There must be an existing order that needs correction. |
| Flow of Events | 1- Accountant opens the order history or order management module. 2- Accountant selects the order to modify. 3- Accountant updates necessary fields and resends the order. 4- System replaces the old order with the updated version. |
| Alternative Flows | - If the order ID is invalid, the system displays an error message. |
| Postconditions | Order is updated and submitted for reprocessing. |
| Data Used | Order ID, product details, modification timestamp. |
| Trigger | Accountant identifies a mistake in a submitted order. |

Table 18 : Resend Order

1. *Use Case Name: Filter Products List*

|  |  |
| --- | --- |
| Actors | Accountant |
| Description | This use case allows the accountant to filter the list of products based on specific criteria for easier navigation or analysis. |
| Preconditions | Product data must be available in the system. |
| Flow of Events | 1- Accountant accesses the product listing screen. 2- Accountant selects one or more filter options (e.g., category, supplier, availability). 3- System applies the filters and displays the results. |
| Alternative Flows | - If no products match the filter, system shows an empty result message. |
| Postconditions | Filtered product list is displayed on screen. |
| Data Used | Product attributes, filter parameters. |
| Trigger | Accountant wants to narrow down product view using filters. |

Table 19 : Filter Products List

1. *Use Case Name: Select Customer*

|  |  |
| --- | --- |
| Actors | Sales Representative |
| Description | This use case allows the sales representative to select a customer from the list before creating an order. |
| Preconditions | Sales representative must be logged in. |
| Flow of Events | 1- Sales rep accesses the customer selection interface. 2- Sales rep browses or searches the list of customers. 3- Sales rep selects a customer for the upcoming order. |
| Alternative Flows | - If no customers are available, the system prompts to add a new customer. |
| Postconditions | Customer is linked to the order being created. |
| Data Used | Customer ID, name, contact info. |
| Trigger | Sales representative begins the order creation process. |

Table 20 : Select Customer

1. *Use Case Name: View Assigned Orders*

|  |  |
| --- | --- |
| Actors | Sales Representative |
| Description | This use case allows the sales representative to select a customer from the list before creating an order. |
| Preconditions | Sales representative must be logged in. |
| Flow of Events | 1- Sales rep accesses the customer selection interface. 2- Sales rep browses or searches the list of customers. 3- Sales rep selects a customer for the upcoming order. |
| Alternative Flows | - If no customers are available, the system prompts to add a new customer. |
| Postconditions | Customer is linked to the order being created. |
| Data Used | Customer ID, name, contact info. |
| Trigger | Sales representative begins the order creation process. |

Table 21 : View Assigned Orders

1. *Use Case Name: View Customer Path*

|  |  |
| --- | --- |
| Actors | Delivery Driver |
| Description | This use case allows the delivery driver to view the map route to the customer's location. |
| Preconditions | Driver must be logged in and have an assigned order with a valid delivery address. |
| Flow of Events | 1- Driver selects a delivery order. 2- System retrieves the customer’s location. 3- Map interface displays the route to the destination. |
| Alternative Flows | - If GPS is disabled, the system request’s location access. |
| Postconditions | Route to the customer is shown on the map. |
| Data Used | Order ID, customer address, GPS coordinates. |
| Trigger | Driver wants to navigate to the customer's location. |

Table 22 : View Customer Path

1. *Use Case Name: View a Receipt for the Signature*

|  |  |
| --- | --- |
| Actors | Delivery Driver |
| Description | This use case allows the delivery driver to present a digital receipt to the customer for signing upon delivery. |
| Preconditions | Driver must be logged in and the order must be marked as ready for delivery. |
| Flow of Events | 1- Driver selects the completed delivery order. 2- System generates and displays the receipt. 3- Customer signs the receipt on the mobile device of Driver. |
| Alternative Flows | - If the customer refuses to sign, driver can mark as 'Signature Refused' and provide a reason. |
| Postconditions | Customer signature is recorded or refusal is documented. |
| Data Used | Order ID, customer name, signature image or reason code. |
| Trigger | Driver completes the delivery and presents the receipt. |

Table 23 : View a Receipt for the Signature

1. *Use Case Name: Search for Product*

|  |  |
| --- | --- |
| Actors | Accountant |
| Description | This use case allows the accountant to search for a specific product using various attributes. |
| Preconditions | Product data must exist in the system. |
| Flow of Events | 1- Accountant navigates to the product search interface. 2- Accountant enters search criteria such as product name, category, or barcode. 3- System displays a list of matching products. |
| Alternative Flows | - If no products match the search criteria, the system shows a 'No Results Found' message. |
| Postconditions | Search results are displayed. |
| Data Used | Product attributes, search input. |
| Trigger | Accountant wants to find a specific product quickly. |

Table 24 : Search for Product

1. *Use Case Name: Receive Stock-In Orders*

|  |  |
| --- | --- |
| Actors | Supplier |
| Description | This use case allows the supplier to receive and acknowledge incoming stock-in orders sent by the accountant. |
| Preconditions | Supplier must be logged in and assigned pending stock-in orders. |
| Flow of Events | 1- Supplier opens the stock-in orders module. 2- System displays a list of pending orders. 3- Supplier reviews the order and confirms receipt. |
| Alternative Flows | - If there are discrepancies, supplier can send feedback or request changes. |
| Postconditions | Order status is updated to 'Received' or sent back for review. |
| Data Used | Order ID, product list, confirmation date. |
| Trigger | Supplier receives a new stock-in order. |

Table 25 : Receive Stock-In Orders

