CSE 428

Data Engineering

Software Requirements Specification

Inventory Management System

Submitted by:

Yassin Mamdouh – 120210002

Mark Eskander – 120210018

Abdelrahman Mohamed Salah – 120210020

Razan Mohamed Kenawy – 120210059

Nour Ahmed Wasim – 120210060

Submitted to:

Dr. Ahmed Antar

Table of Content

List of Figures

1 Introduction

The Inventory Management System (IMS) plays a crucial role in modern businesses by automating the process of tracking, managing, and maintaining inventory records. This document serves as the Software Requirements Specification (SRS) for the development of the IMS, outlining the functional and nonfunctional requirements essential for its successful implementation. The system aims to streamline stock monitoring, supplier management, and reporting tasks to enhance operational efficiency and accuracy.

1.1 Purpose

The purpose of this project is to develop an inventory management system (IMS) aimed at efficiently tracking and managing inventory levels, product details, and supplier information. The system will automate the process of monitoring stock levels, ensuring that businesses can effectively manage their products and avoid overstocking or understocking. The IMS will allow administrators and inventory managers to easily update product details, monitor stock quantities, and generate reports for decision-making. The system will also include features for supplier management, allowing businesses to track their suppliers, order history, and pricing information. By integrating these functionalities, the IMS will provide real-time insights into inventory status, enabling businesses to make data-driven decisions. Furthermore, the system will also support reporting and data analysis, allowing managers to view trends in product sales, inventory usage, and supplier performance. Ultimately, the project aims to provide a comprehensive solution for inventory management, enhancing operational efficiency, improving decision-making, and optimizing stock control for businesses.

1.2 Scope

The scope of the Inventory Management System (IMS) encompasses a comprehensive software solution designed to revolutionize inventory tracking and management for businesses across various industries. Beyond its core purpose of monitoring inventory levels, managing product information, and supporting reporting, the system aims to address the complex challenges faced by inventory managers and business administrators. By utilizing advanced technologies and providing customizable features, the IMS streamlines inventory management processes, reducing manual intervention and enhancing operational efficiency. Moreover, the system improves the accuracy and reliability of inventory records, ensuring businesses can maintain optimal stock levels, reduce waste, and meet customer demand. Through proactive features such as automated stock alerts and supplier management, the IMS helps businesses avoid overstocking, understocking, and delayed order fulfillment. Additionally, the system's flexibility allows businesses to tailor inventory policies and reporting formats to meet their specific operational requirements, empowering them to make data-driven decisions and optimize their supply chain. Ultimately, the Inventory Management System serves as a transformative tool for optimizing inventory management, improving operational effectiveness, and enhancing business outcomes by providing real-time insights, reducing costs, and improving customer satisfaction.

1.3 Definitions, Acronyms, and Abbreviations

- **IMS**: Inventory Management System. A software solution designed to track and manage inventory, including products, stock levels, and suppliers.
- **CRUD**: Create, Read, Update, Delete. The basic operations that can be performed on database records.
- **SKU**: Stock Keeping Unit. A unique identifier for each product or item in inventory.
- **DBMS**: Database Management System. A software system used to manage and organize data in the IMS, such as MySQL, PostgreSQL, or MongoDB.
- **API**: Application Programming Interface. A set of protocols and tools used for integrating and communicating between different software systems, including the IMS and external systems.
- **Supplier Management**: The process of managing supplier information, including contact details, pricing, and order history.
- **Inventory Report**: A document generated by the IMS that provides a detailed view of inventory levels, trends, and product performance.
- **User Roles**: Different access levels within the IMS, including Admin, Inventory Manager, and Viewer, each with varying permissions.

1.4 Overview

This document provides a detailed Software Requirements Specification (SRS) for the development of the Inventory Management System (IMS). The IMS is designed to streamline inventory tracking and management processes within businesses, improving efficiency, accuracy, and decision-making. This document is organized into the following sections to ensure a comprehensive understanding of the system's design, functionality, and requirements.

- **Section 2: Overall Description** - This section provides a high-level overview of the IMS, describing the system's purpose, key features, user roles, and operational environment. It outlines how the system will be integrated into existing business workflows and the value it will provide to inventory management processes.
- **Section 3: Specific Requirements** - This section outlines the functional and non-functional requirements that the IMS must meet. It details the specific features the system must support, such as inventory tracking, supplier management, and reporting, along with performance and security requirements.
- **Section 4: External Interface Requirements** - This section specifies how the IMS will interact with external systems, including database management systems, third-party APIs, and hardware components such as barcode scanners or printers.

- **Section 5: Other Requirements** - This section addresses additional requirements related to system security, backup procedures, scalability, and any constraints or assumptions that may affect the system's design and implementation.
- **Section 6: Appendices** - This section includes supplementary materials such as diagrams, flowcharts, and tables that provide additional context or clarify complex requirements.

2 Overall Description

2.1 Product Perspective

The Inventory Management System (IMS) is designed as a web-based application aimed at improving inventory tracking, stock management, and reporting for businesses of all sizes. It will be accessible through modern web browsers and, if needed, be optimized for mobile devices to ensure ease of use on various platforms. The IMS will operate within a client-server architecture, where the client interacts with the system through a user-friendly web interface, and the server manages the back-end processes, including data storage, business logic, and integration with external systems.

- **Web-Based Application**: The IMS will be accessible via standard web browsers (e.g., Google Chrome, Mozilla Firefox, Microsoft Edge), ensuring cross-platform compatibility. The web-based nature of the application allows users to access the system from any location with an internet connection, facilitating remote inventory management and real-time updates.
- **Mobile Responsiveness**: To accommodate users on various devices, the IMS will feature a mobile-responsive design, ensuring that the user interface is intuitive and functional across smartphones and tablets. This will enable business owners and inventory managers to monitor and manage inventory on the go, increasing flexibility and accessibility.
- **Client-Server Architecture**: The IMS will be built on a client-server architecture, where the client-side (web browser or mobile app) interacts with the server-side to process requests, retrieve data, and update inventory records. The server will host the database, manage the application logic, and ensure secure communication between the client and the system's data.

2.2 Product Functions

The Inventory Management System (IMS) is designed to support key functionalities that streamline and enhance inventory management processes. These functions are critical to ensuring the efficient operation of businesses, helping managers track inventory levels, update stock, manage suppliers, and generate reports. The following are the primary functions of the system:

- **Inventory Tracking**: The IMS will enable businesses to track inventory items in real time. Each product will be identified by a unique SKU, and the system will record stock levels, product details, and movement history, allowing users to monitor current stock quantities and trends over time.
- **Stock Updates**: Users will be able to update stock levels easily by adding new products, adjusting quantities, and recording sales or purchases. The system will automatically adjust stock levels based on these updates, ensuring that the inventory records are always accurate and up-to-date.
- **Supplier Management**: The IMS will allow businesses to manage their supplier relationships. Users can store supplier contact details, track purchase orders, view delivery histories, and monitor supplier performance. This functionality ensures seamless integration with the procurement process.
- **Reports Generation**: The IMS will support the generation of detailed reports, including inventory summaries, sales performance, stock turnover, and supplier performance. These reports will be available in various formats (e.g., PDF, Excel) and will provide businesses with insights to make data-driven decisions.

2.3 User Classes and Characteristics

The Inventory Management System (IMS) is designed to cater to different types of users, each with varying levels of access and responsibilities. The system defines three primary user classes: Admin, Inventory Manager, and Viewer/User. Each class has specific permissions and functionalities, ensuring that users can perform their tasks effectively while maintaining the security and integrity of the system.

- **Admin: Full Access**
  - **Responsibilities:** The admin has the highest level of access and is responsible for managing the overall system, including user management, configuration settings, and system maintenance.
  - **Capabilities:** The admin can add, edit, and delete users; manage inventory settings; configure system preferences; generate all types of reports; and view detailed logs of system activities. They also have access to sensitive data and the ability to modify system-wide settings.
- **Inventory Manager: Manage Stocks**
  - **Responsibilities:** The inventory manager is responsible for overseeing the day-to-day inventory operations, including tracking stock levels, adding new products, and updating product quantities.
  - **Capabilities:** The Inventory Manager can view and update stock levels, add new inventory items, process stock movements (e.g., sales, purchases), and generate inventory-related reports such as stock status and order history. However, they do not have access to system configuration settings or user management.

- **Viewer/User: View Reports and Stock Levels**
  - **Responsibilities:** The viewer/user is a read-only user with limited access to the system. They can view reports and stock levels but cannot make any changes to the system or inventory.
  - **Capabilities:** The viewer/user can view predefined reports, check current stock levels, and monitor inventory performance. They are primarily intended for stakeholders or business executives who need to monitor the system's performance without interacting with the inventory directly.

2.4 Operating Environment

The Inventory Management System (IMS) will be developed using the .NET MVC framework and will operate in a Windows-based environment, optimized for deployment on a Microsoft IIS web server. The system will utilize SQL Server as the database backend to ensure robust data management, scalability, and seamless integration with the .NET ecosystem. The following are the key components of the operating environment:

- **Web Browsers:**
  - The IMS will be accessible via modern web browsers, ensuring compatibility across various devices and platforms. Supported browsers include:
    - Google Chrome
    - Mozilla Firefox
    - Microsoft Edge
    - Safari
  - The system will feature a responsive design, providing an optimized user interface for both desktop and mobile devices, adapting to various screen sizes and resolutions.
- **Server:**
  - The IMS will be hosted on a Microsoft IIS (Internet Information Services) web server, ensuring smooth integration with the .NET MVC framework.
  - The server will support the deployment of the application through IIS, with the following configurations:
    - Windows Server (2012 or later) or a Windows-based environment with IIS enabled.
    - .NET Framework 4.7 or later (for .NET MVC compatibility).
  - The system will be optimized for performance, scalability, and security in a Windows-based environment, with built-in support for .NET applications.
- **Database:**
  - The IMS will use SQL Server as the relational database management system (RDBMS). The system will be compatible with SQL Server 2012 or later versions to ensure high performance, security, and data integrity.

  - The database will be designed to store inventory information, user accounts, product details, transactions, and report data.
  - The SQL Server Management Studio (SSMS) will be used to manage the database, perform backups, and run queries.

2.5 Design and Implementation Constraints

The development of the Inventory Management System (IMS) will be subject to several design and implementation constraints to ensure high performance, security, maintainability, and scalability. These constraints are critical to the system's success and are outlined below:

- **Use of Specific Frameworks:**
  - The IMS will be developed using the .NET MVC framework, which is a widely used and proven technology for building web applications. This framework will be used to create the front-end user interface and back-end logic, ensuring seamless integration with the database and supporting best practices for web application development.
  - The system will be designed using C# for back-end logic and HTML, CSS, and JavaScript for the front-end. The use of Bootstrap will be implemented to create a responsive and visually appealing user interface.
  - SQL Server will be used as the database management system, ensuring smooth data integration with the .NET ecosystem and offering features such as strong security, scalability, and high availability.
- **Scalability and Performance:**
  - The system will be designed to handle high traffic and large volumes of data efficiently. Caching strategies, such as using Redis or Memcached, may be implemented to reduce database load and improve response times.
  - The architecture will support easy scalability, allowing the system to handle increased user load and data growth over time by optimizing database queries, implementing load balancing, and utilizing cloud hosting solutions if necessary.

2.6 Assumptions and Dependencies

The development and operation of the Inventory Management System (IMS) are based on several key assumptions and dependencies, which are critical for its successful implementation and performance:

- **Assumptions:**
  - The system assumes that users have basic computer literacy and familiarity with web-based applications.

  - The IMS assumes reliable internet access for cloud-based features, including the login process, real-time inventory updates, and report generation.
  - The system assumes that the business has an existing structure for inventory management, such as a product catalog and an inventory tracking process, that will be integrated into the system.

- **Dependencies:**
  - The IMS depends on the .NET MVC framework for building the web application and server-side logic, ensuring compatibility with Microsoft IIS web servers.
  - The system depends on SQL Server as the database backend for data storage, ensuring efficient data management, scalability, and reliability.
  - The IMS depends on third-party libraries and frameworks for certain functionalities, such as:
    * Bootstrap for responsive web design.
    * jQuery for enhanced interactivity and DOM manipulation.
    * Entity Framework for object-relational mapping (ORM) and seamless database interaction.
    * ASP.NET Identity for user authentication and authorization.
  - The system also depends on Microsoft IIS for hosting and running the web application on Windows Server.

3 System Requirement Satisfaction

3.1 Functional Requirements

The following functional requirements are crucial for the Inventory Management System (IMS) to support key operations such as inventory tracking, sales management, stock ordering between branches, and more. Each requirement is described in detail below.

- **FR1: User Authentication and Authorization**
  - The system shall allow users to log in and log out securely.
  - The system shall implement role-based access control (RBAC), with roles including Admin, Manager, and User, each with different levels of access.
- **FR2: Inventory Management**
  - The system shall allow users to add, edit, and delete inventory items.
  - The system shall track the quantity of each product at each branch location.
  - The system shall automatically update inventory levels when sales are made or stock is transferred between branches.
- **FR3: Sales Management**

  - The system shall allow employees to record sales to customers.
  - The system shall capture customer details, product sold, quantity, and pricing.
  - The system shall allow employees to input special fields for specific products (e.g., windshield code, adhesive amount) during sales transactions.

- **FR4: Stock Ordering Between Branches**
  - The system shall allow branches to request products from other branches.
  - The system shall allow users to create and manage inter-branch orders.

- **FR5: Product Catalog**
  - The system shall maintain a catalog of products, including detailed information such as product name, type, and color.
  - The system shall display the product catalog with the option to search and filter by product attributes (e.g., name, type, color).

- **FR6: Client Management**
  - The system shall store customer data for record-keeping and future reference.
  - The system shall allow employees to associate customer details with sales transactions.

- **FR7: Reporting and Analytics**
  - The system shall generate reports on inventory levels, sales transactions, and product performance.
  - The system shall support custom report generation, including daily, weekly, and monthly reports.

- **FR8: Email Notifications**
  - The system shall send low-stock email alerts to the admin when inventory levels fall below predefined thresholds.

- **FR9: User Role Management**
  - The system shall allow administrators to manage user roles (admin, manager, user) and assign appropriate access permissions.

- **FR10: Search Functionality**
  - The system shall provide search functionality to allow users to search for inventory items by name, type, or location.

3.2 Non-Functional Requirements

This section outlines the non-functional requirements essential for ensuring the quality and performance of the Inventory Management System.

- **NFR-1: Technology Stack**
  - The system shall be developed using the ASP.NET MVC framework with C# as the programming language. Microsoft SQL Server will be used for database management, and Entity Framework will serve as the ORM.
- **NFR-2: Performance**
  - The system must be able to handle at least 50 concurrent users performing sales and inventory updates without performance degradation.
- **NFR-3: Responsiveness**
  - The system should respond to user actions within 2 seconds for at least 95% of operations.
- **NFR-4: Security**
  - All user credentials must be securely hashed using modern cryptographic standards (e.g., SHA-256). Role-based access control must be implemented to restrict sensitive operations to authorized users.
- **NFR-5: Backup and Recovery**
  - The system must support scheduled automatic backups of the database and provide tools for recovery in the event of data loss or corruption.
- **NFR-6: Usability**
  - The system interface must be user-friendly and accessible on commonly used modern browsers. It should support both desktop and mobile screen sizes.
- **NFR-7: Validation and Error Handling**
  - All form inputs must be validated both on the client side and the server side. Clear error messages should be shown for invalid or incomplete input.
- **NFR-8: Logging and Monitoring**
  - The system must log key user actions, such as logins, sales, inventory updates, and transfers, for auditing purposes.
- **NFR-9: Maintainability**
  - The codebase shall follow clean coding standards and modular architecture to allow for future maintenance and updates with minimal effort.
- **NFR-10: Compliance**
  - The system must comply with applicable data protection and privacy regulations, ensuring secure handling of customer and employee information.

3.3 Database Requirements

The database for the Inventory Management System is a core component, structured using Microsoft SQL Server and accessed via Entity Framework. It is designed to ensure efficient storage, retrieval, and management of all inventory-related data. Below are the primary tables and their essential fields:

- **Products Table**
  - ProductID (Primary Key)
  - Name
  - Type
  - Color
- **Inventory Table**
  - InventoryID (Primary Key)
  - ProductID (foreign key)
  - LocationID (Foreign Key)
  - Quantity
- **Locations Table**
  - LocationID (Primary Key)
  - LocationName
  - Address
- **Orders Table**
  - OrderID (Primary Key)
  - SourceLocationID (Foreign Key)
  - DestinationLocationID (Foreign Key)
  - OrderDate
  - Status
- **OrderDetails Table**
  - OrderDetailID (Primary Key)
  - OrderID (Foreign Key)
  - ProductID (foreign key)
  - Quantity
- **Sales Table**
  - SaleID (Primary Key)
  - EmployeeID (Foreign Key)

  - ClientID (Foreign Key)
  - SaleDate
  - WindshieldCode (nullable)
  - AdhesiveAmount (nullable)

- **SaleDetails Table**
  - SaleDetailID (Primary Key)
  - SaleID (Foreign Key)
  - ProductID (foreign key)
  - Quantity
  - Price

- **Clients Table**
  - ClientID (Primary Key)
  - FullName
  - ContactNumber
  - Email
  - Address

- **Employees Table**
  - EmployeeID (Primary Key)
  - FullName
  - Username
  - PasswordHash
  - Role
  - AssignedLocationID (Foreign Key)

4 External Interface Requirements

4.1 User Interfaces

The system provides a clean and intuitive web-based user interface designed using Razor Views and standard HTML/CSS. Below are the key interfaces:

- **Login Page:** Provides secure access to the system using a username and password. Role-based redirection ensures employees only see relevant features for their role (e.g., admin, sales clerk, manager).
- **Dashboard:** Displays high-level overviews such as stock summaries per branch, recent sales, pending inter-branch orders, and alerts (e.g., low stock warnings).

- **Inventory Management Screen:** Allows users to view, filter, and manage product quantities by location. Users can update inventory manually or through sales/transfer transactions. Includes product details and current stock levels.
- **Sales Entry Interface:** Enables employees to record customer sales. Includes dynamic dropdowns for product selection, quantity, pricing, and special fields for car-related data such as windshield codes and adhesive amounts.
- **Inter-Branch Orders Page:** Allows users to initiate, approve, and track stock transfer requests between branches. Displays order history and current status.
- **Reports Page:** Generates printable reports for stock, sales, inter-branch transfers, and customer activity. Filters allow users to select date ranges, branches, and product categories.
- **Notifications Panel:** Displays system alerts such as low inventory, pending orders, and login/logout audit records.
- **Employee Management Interface:** For administrators to add, edit, and assign employees to locations. Also used to manage roles and access rights.
- **Client Management Screen:** Stores and displays customer information. Useful for looking up past transactions or managing frequent clients.

4.2 Hardware Interfaces

The Inventory Management System is a web-based application designed to run on standard computing hardware. The following hardware interfaces are required for proper deployment and operation:

- **Web Server:** The application will be hosted on a Windows-based web server that supports IIS (Internet Information Services). The server must meet the following minimum specifications:
  - Processor: Quad-core 2.0 GHz or higher
  - RAM: 8 GB minimum
  - Storage: 100 GB free disk space
  - Operating System: Windows Server 2016 or later
  - Software: .NET Framework 4.7 or higher, IIS enabled
- **Database Server:** The system requires a backend database server running Microsoft SQL Server. The database server may be hosted on the same machine as the web server or separately in larger deployments.
  - RAM: 8 GB minimum
  - Storage: SSD preferred, with redundancy and backup capability
  - Software: Microsoft SQL Server 2017 or later

- **Client Devices:** End-users will access the system through modern web browsers on standard client devices:
  - Desktop or Laptop (Windows/macOS/Linux)
  - Minimum screen resolution: 1366×768
  - Modern web browsers: Google Chrome, Microsoft Edge, Mozilla Firefox
  - Internet connection required for real-time operations
- **Networking:** A stable local area network (LAN) or wide area network (WAN) is required to ensure communication between branches and the central server. VPN connectivity is recommended for secure inter-branch communication.

4.3 Software Interfaces

The Inventory Management System interacts with several software components and technologies to provide full functionality. The key software interfaces include

- **Operating System:** The application is built to run on Windows-based environments. Windows Server 2016 or later is required for hosting, and client-side access is compatible with Windows 10/11 and modern operating systems (macOS, Linux) via web browsers.
- **.NET Framework and ASP.NET MVC:** The system is developed using C# on the ASP.NET MVC framework. The web application requires .NET Framework 4.7 or later. Razor Views are used for rendering the user interface dynamically.
- **Entity Framework:** Entity Framework is used as the Object-Relational Mapper (ORM) to interact with the SQL Server database. It provides abstraction over SQL queries and handles CRUD operations.
- **Microsoft SQL Server:** The backend relational database runs on Microsoft SQL Server (2017 or later). All core data, such as inventory records, sales transactions, users, and product information, are stored and retrieved using SQL queries or LINQ.
- **Web Browser Interface:** Users interact with the system through standard web browsers (Google Chrome, Microsoft Edge, Mozilla Firefox). No additional plugins or extensions are required. JavaScript is enabled for dynamic UI behavior.
- **RESTful APIs (optional):** The system can be extended via RESTful APIs to allow integration with external systems, such as external reporting tools, mobile apps, or vendor platforms. APIs may be used for:
  - Fetching inventory levels from mobile clients
  - Submitting inter-branch orders externally
  - Syncing customer data with CRM platforms
- **Third-party Email Service:** For alerting purposes, the system may integrate with a third-party email service such as SendGrid, SMTP-based services, or Outlook 365 to send:

- Low-stock notifications
- Order confirmation emails
- Sales summaries to admins

4.4 Communications Interfaces

The system supports secure and reliable communication between components and users. The following interfaces are utilized:

- **HTTPS Protocol:** All communication between the client (web browser) and the web server is secured using HTTPS. This ensures confidentiality and integrity of data transmitted over the network, such as login credentials, sales data, and customer information.
- **Internal Network Communication:** Within a corporate intranet, branches accessing the application hosted on a central server use secured TCP/IP connections. Network firewalls and access control lists (ACLs) are used to restrict access to only authorized IP ranges.
- **Database Communication:** The application communicates with the Microsoft SQL Server using ADO.NET over a trusted internal network. Connection strings are secured through configuration files, and encrypted authentication is used where supported.
- **Optional REST APIs:** If RESTful APIs are exposed, they are accessible over HTTPS only. Authentication tokens (e.g., JWT or API keys) are required to access endpoints. These APIs support secure integration with mobile apps or third-party systems.
- **Email Notifications:** Communication with email services (e.g., SMTP or SendGrid) is conducted over secure SMTP channels (TLS/SSL). This is used to send automated alerts and transactional emails, such as low-stock notifications and sales summaries.
- **Session Management:** Secure cookies and session tokens are used to manage authenticated sessions. Anti-CSRF tokens and session timeout policies are enforced to enhance security during communication.

5 Other Non-functional Requirements

In addition to the core functional requirements, the following non-functional requirements are critical to the success and sustainability of the Inventory Management System:

5.1 Backup and Recovery Plan

A comprehensive backup and recovery plan is implemented to ensure the availability and integrity of data in case of system failure or data loss. This includes:

- **Regular Backups:** Daily, weekly, and monthly backups of the database and application data are scheduled. These backups are stored securely, with redundancy across multiple locations (on-premises and cloud-based).
- **Automated Backup Scripts:** Automated backup scripts are set up to ensure that backups occur without manual intervention. Backup logs are generated and monitored to detect any issues.
- **Recovery Procedures:** Detailed recovery procedures are established to restore the system in case of failure. This includes restoring both the application and the database from backup files with minimal downtime.
- **Backup Testing:** Periodic testing of the backup and recovery process ensures the ability to restore data in a timely manner.

5.2 Audit Logs for Item Changes

To ensure traceability and accountability, the system will maintain audit logs for all changes to inventory items, sales, orders, and other critical data. This includes:

- **Change Logging:** All updates, deletions, and additions to the database (e.g., inventory quantity changes, sales transactions, and order updates) are logged, including the user who made the change, timestamp, and nature of the change.
- **Log Storage:** Logs are stored in a secure, centralized location and protected from unauthorized access. Older logs are archived periodically for long-term storage.
- **Audit Access:** Only authorized personnel can access audit logs, and the logs are reviewed regularly to identify any potential issues or suspicious activity.

5.3 Future Scope: Barcode Scanning Integration

The system is designed with future scalability in mind, with plans to integrate barcode scanning capabilities. This will streamline inventory management and sales processes by enabling:

- **Barcode Scanning for Sales:** Cashiers and employees will be able to scan product barcodes to quickly add items to sales transactions, reducing human errors and improving transaction speed.

- **Barcode-Based Inventory Management:** Inventory updates, stock movement, and product ordering can be enhanced with barcode scanning, enabling faster and more accurate tracking of inventory across branches.
- **Integration with Barcode Scanning Devices:** The system will support various barcode scanners, and mobile apps will allow users to scan products using smartphone cameras for better accessibility.

6 Appendices

This section includes supplementary materials that provide additional context and clarity to the main body of the document. The materials are essential for understanding and implementing the system and include various diagrams, flowcharts, and tables that illustrate the architecture, workflows, and relationships between different entities in the system. These appendices aim to make the requirements more comprehensible and facilitate smoother development and implementation of the Inventory Management System.

6.1 Tables and Data Models

This section includes data models and tables that define the structure of key entities within the system. These tables describe the data fields, their relationships, and constraints that will be implemented in the database. The following tables are provided:

- **Product Table:** Defines the fields for storing product information, including product name, type, color, and other attributes.
- **Inventory Table:** Describes how products are linked to specific locations with corresponding quantities.
- **Sales Table:** Provides a structure for recording each sale transaction, including customer details, products sold, quantities, and pricing.
- **Order Table:** Contains information on inter-branch stock orders, detailing which products are requested, from which location, and in what quantity.

6.2 Additional Supporting Materials

Additional materials may be included to help clarify specific parts of the requirements or to support development decisions. These can include:

- **Sample Data:** Examples of input data for testing and validation of the system.
- **API Specifications:** If applicable, details of any APIs that will be used to interface with other systems or external tools.
- **User Stories:** Descriptions of system interactions from the perspective of different types of users, illustrating common use cases and workflows.

6.3 Diagrams and Flowcharts

The following diagrams and flowcharts visually represent the system architecture, business processes, and data flows:

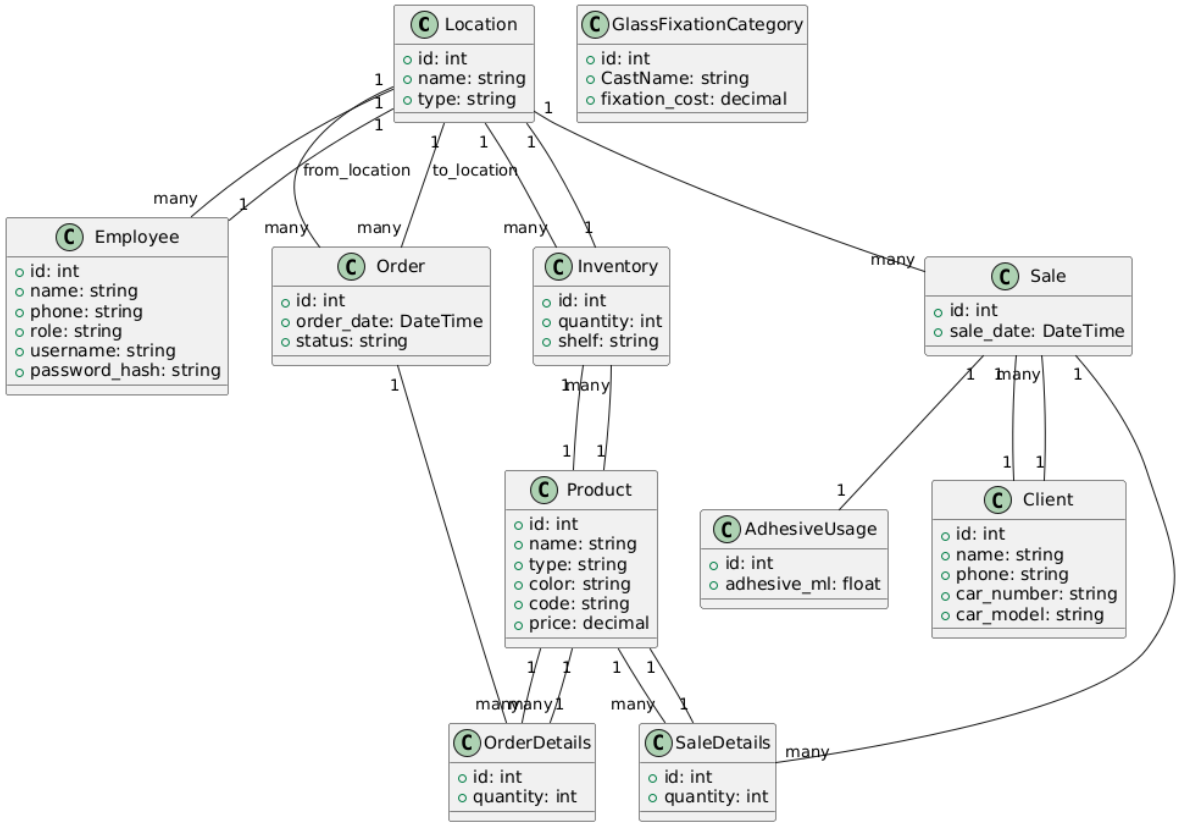

Figure 1: Class Diagram

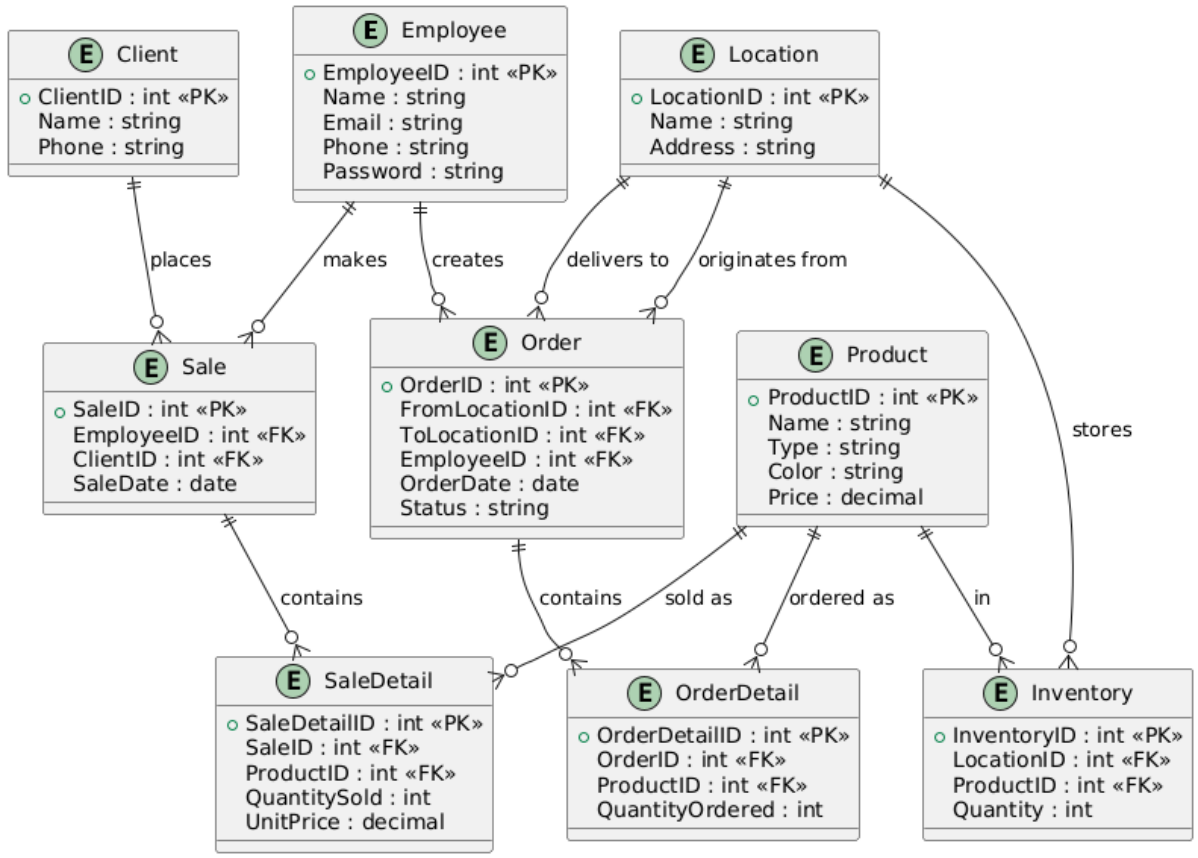

Figure 2: Entity Relationship Diagram (ERD)

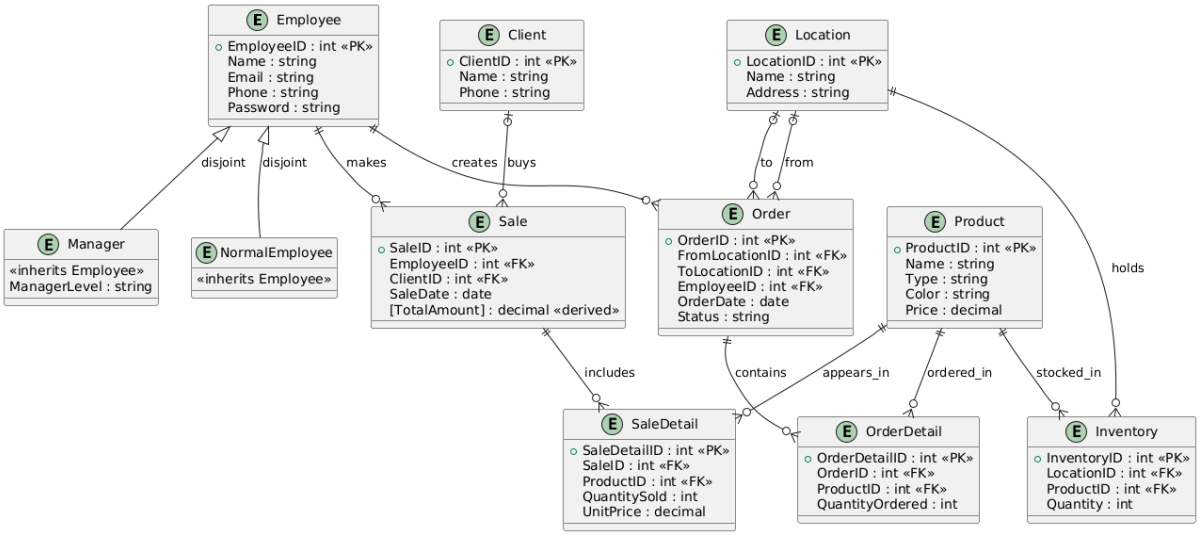

Figure 3: Enhanced Entity Relationship Diagram (EERD)

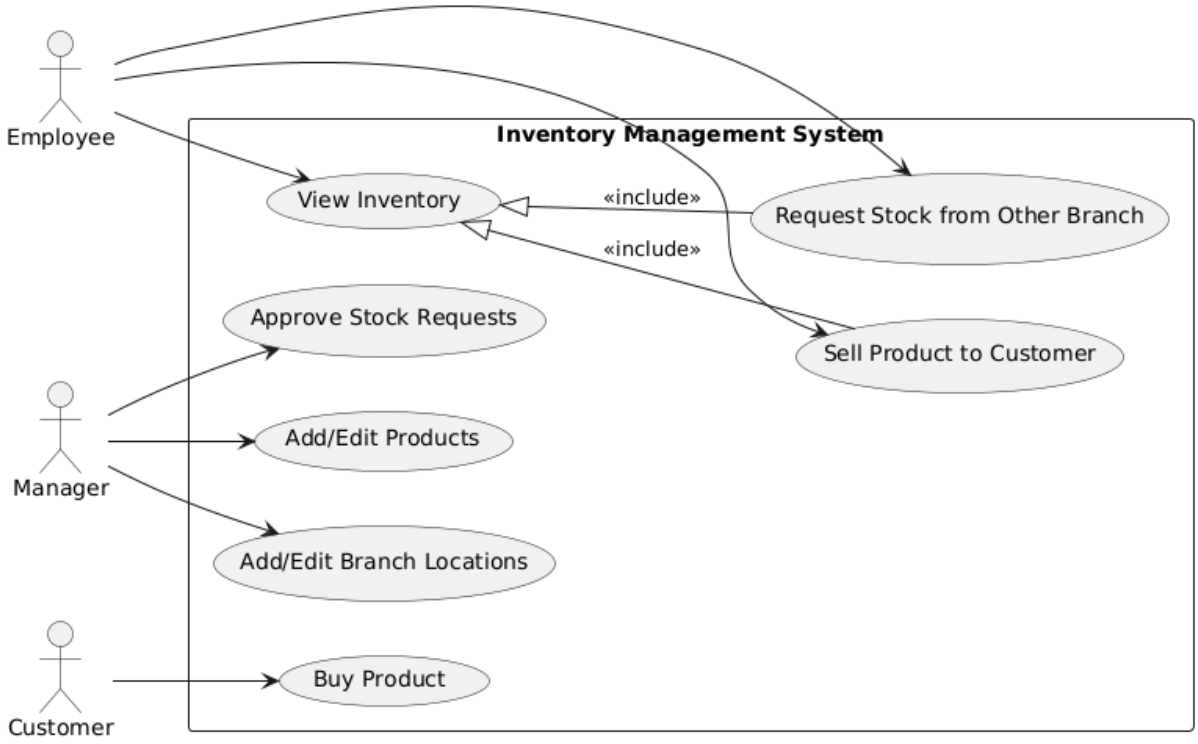

Figure 4: Use Case Diagram

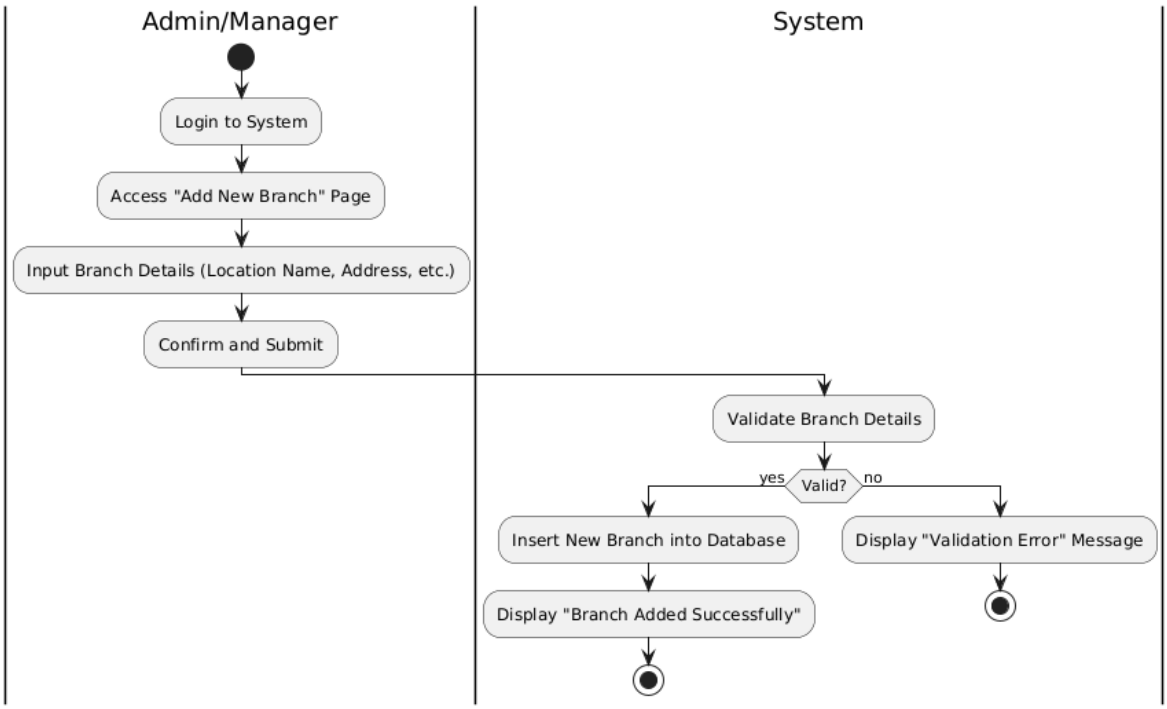

Figure 5: Activity Diagram: Add New Branch by Admin/Manager

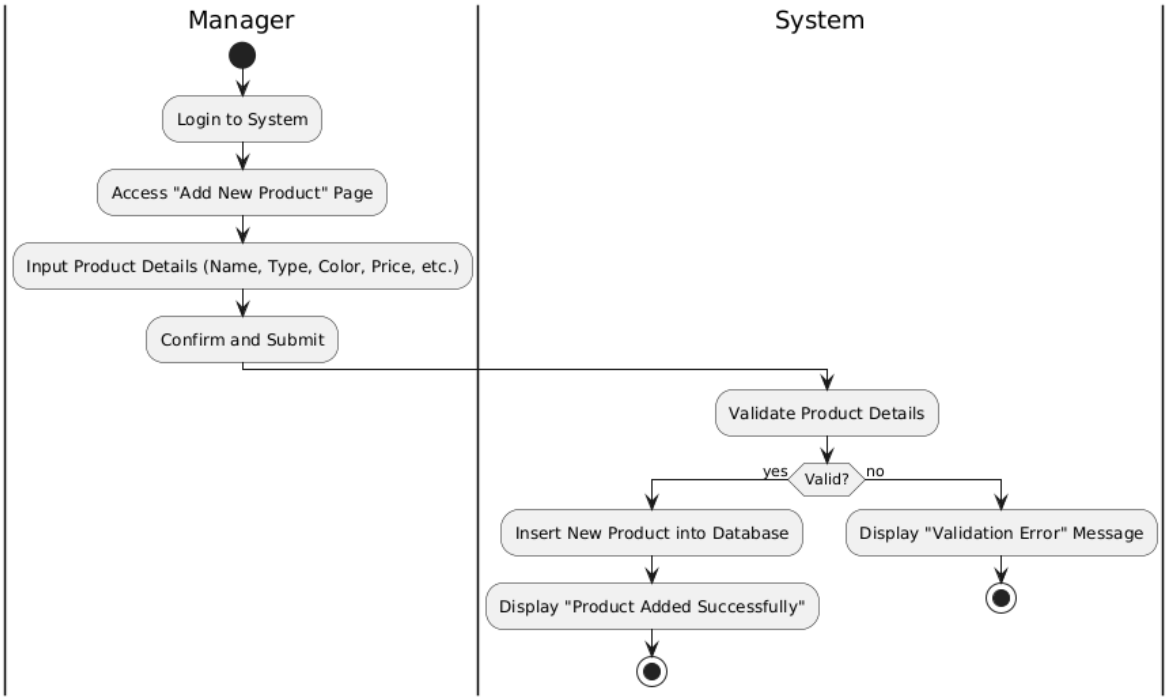

Figure 6: Activity Diagram: Add New Product by Manager

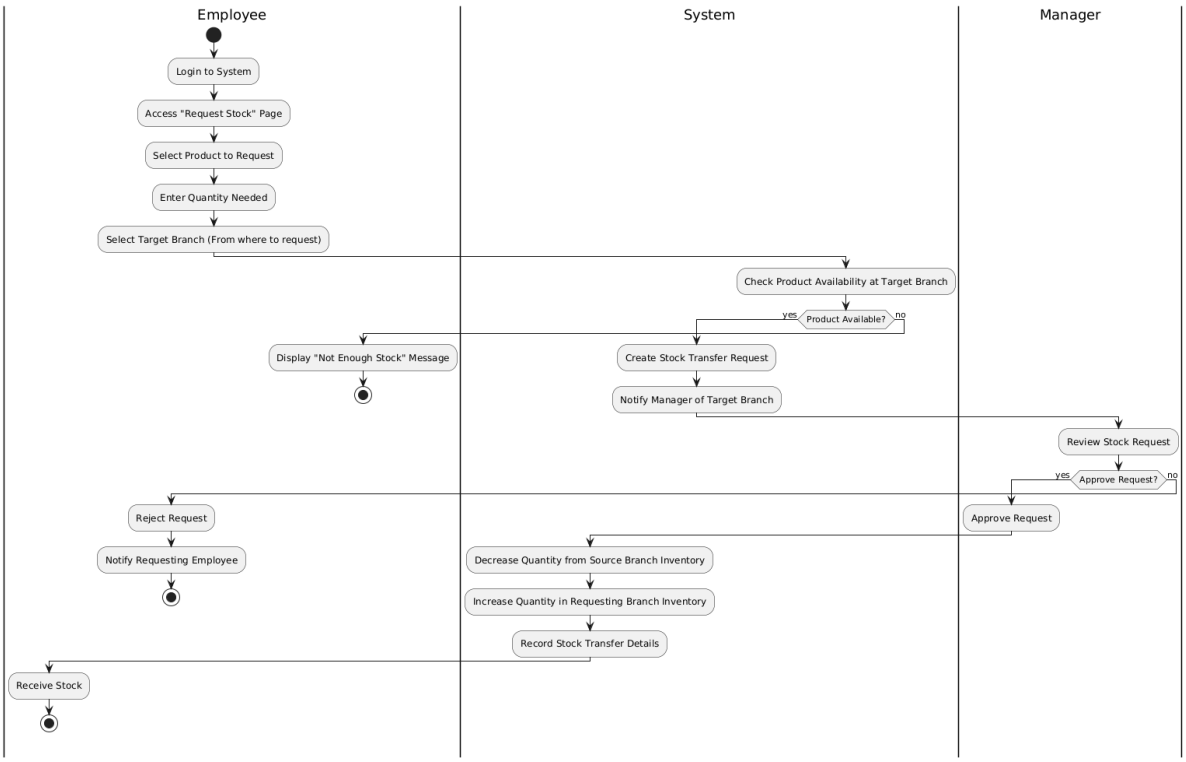

Figure 7: Activity Diagram: Request Stock from Another Branch

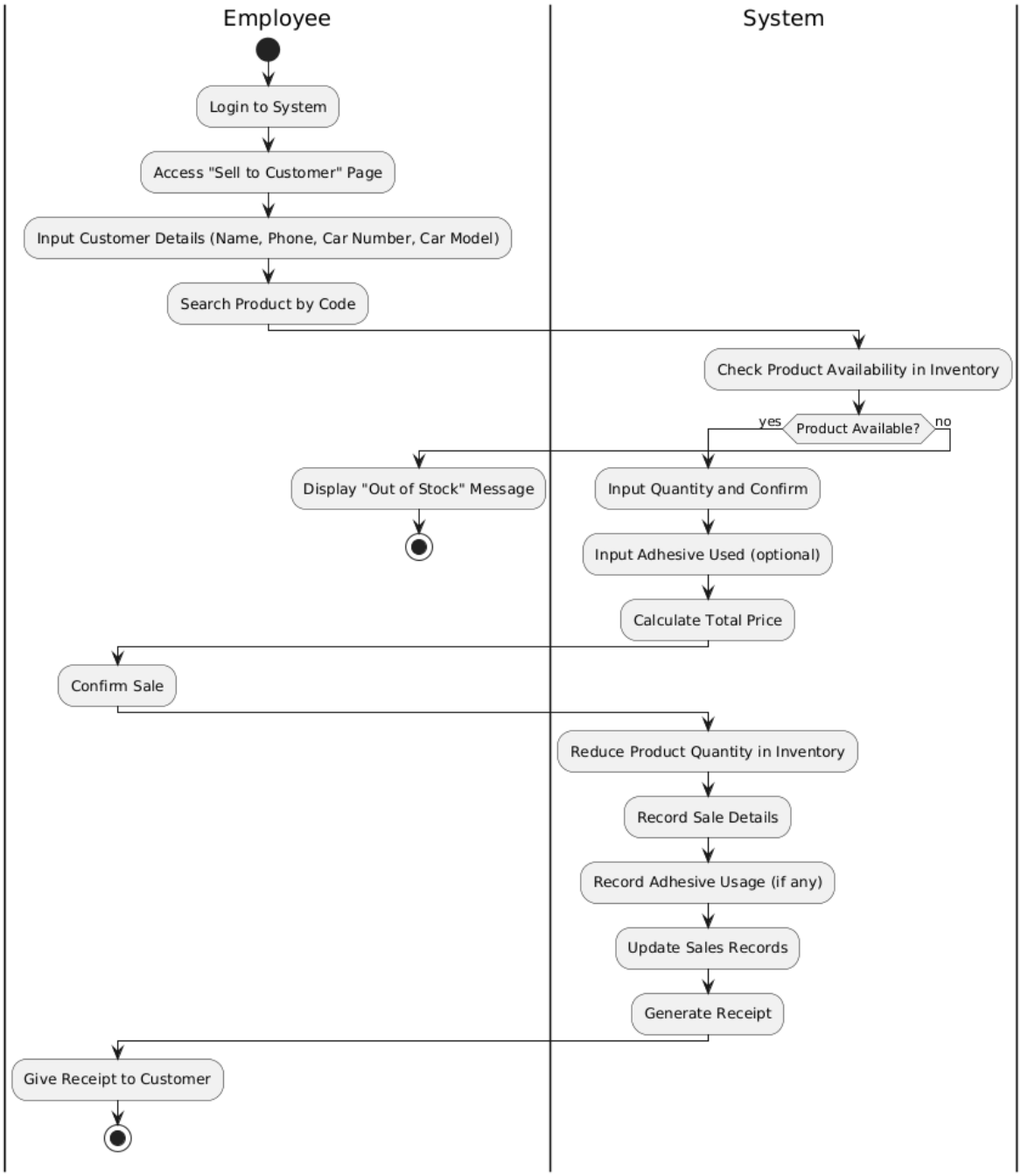

Figure 8: Activity Diagram: Sell Product to Customer

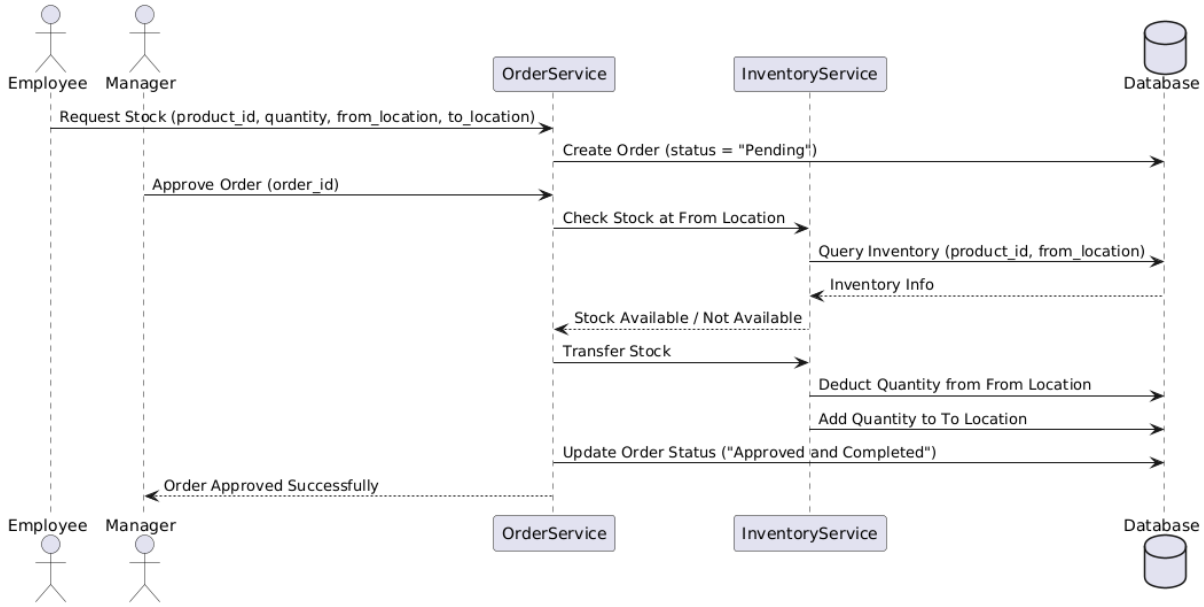

Figure 9: Sequence Diagram: Request Stock from Another Branch

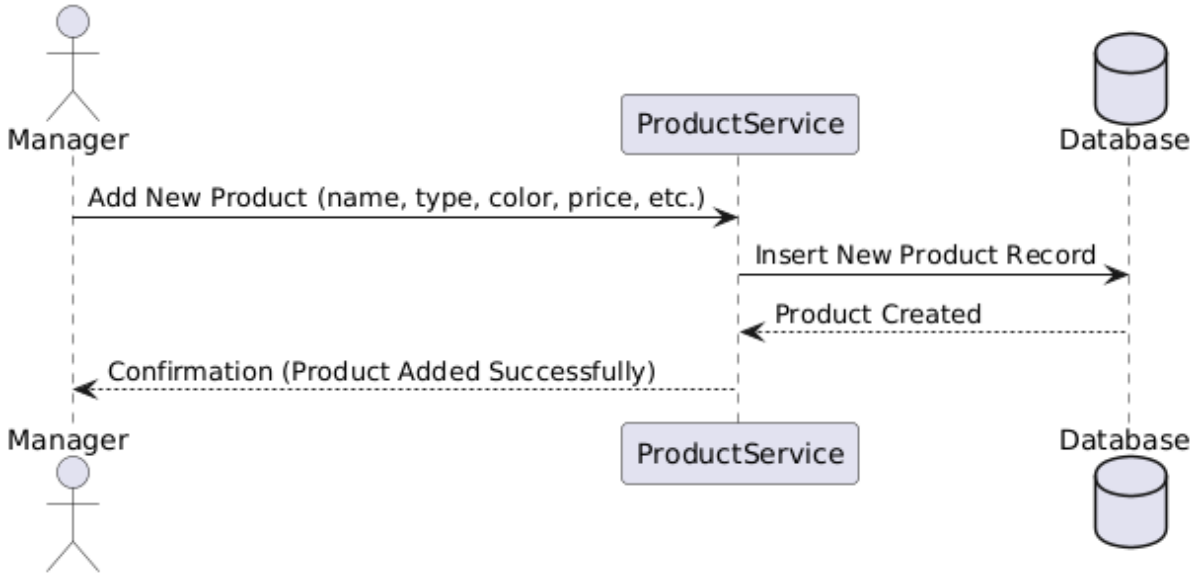

Figure 10: Sequence Diagram: Add New Product

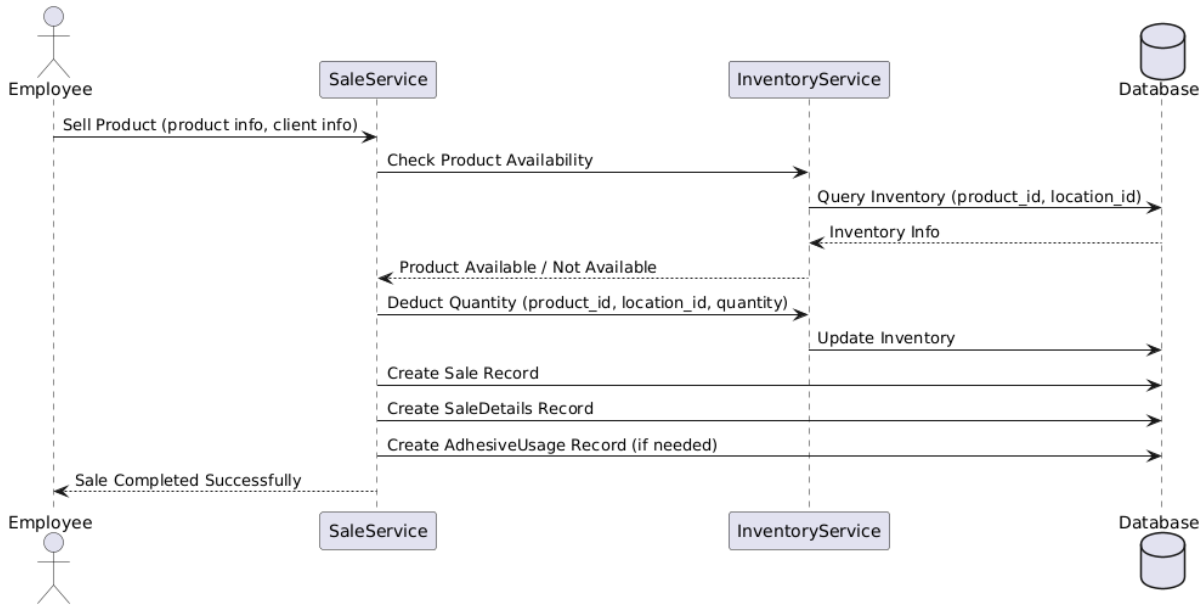

Figure 11: Sequence Diagram: Sell Product to Customer