Software Requirements and Design Document

for

HissabTrack

Prepared by

Hamza Ahmad 22i-0753

Hussain Ali Zaidi 22i-0902

M. Daniyal Aziz 22i-0753

GoatTech

November 13, 2024

Table of Contents

Table of Contents ii

1. Introduction 1

1.1 Purpose 1

1.2 Product Scope 1

1.3 Title 1

1.4 Objectives 1

1.5 Problem Statement 1

2. Overall Description 1

2.1 Product Perspective 1

2.2 Product Functions 2

2.3 List of Use Cases 2

2.4 Extended Use Cases 2

2.5 Use Case Diagram 2

3. Other Nonfunctional Requirements 2

3.1 Performance Requirements 2

3.2 Safety Requirements 2

3.3 Security Requirements 2

3.4 Software Quality Attributes 2

3.5 Business Rules 3

3.6 Operating Environment 3

3.7 User Interfaces 3

4. Domain Model 3

5. System Sequence Diagram 3

6. Sequence Diagram 3

7. Class Diagram 4

8. Package Diagram 4

9. Deployment Diagram 4

# Introduction

## Purpose

## This document specifies the software requirements for HissabTrack, an inventory management system designed for emerging corporations with increasing inventory management needs. The system aims to automate and streamline inventory tracking, sales monitoring, and product management. It provides real-time updates on stock levels, generates sales reports, and manages inventory across multiple locations. This document outlines the core functionalities and requirements for the initial version of the system, developed using Java, JavaFX, and SQL for data storage.

## Product Scope

**HissabTrack** is a comprehensive inventory management solution intended to address the growing needs of corporations as they scale. The system will support efficient tracking of inventory, product categorization, sales monitoring, and report generation. The software will allow users to manage inventory in a streamlined manner, ensuring accurate stock levels, better decision-making, and optimized inventory control. It is designed to be scalable and flexible, capable of handling an increasing volume of inventory and transactions as business operations expand.

The product will integrate an intuitive user interface with real-time database management to simplify inventory management tasks. **HissabTrack** is aligned with the goal of automating key business functions to improve operational efficiency and reduce manual errors.

## Title

**HissabTrack: Scalable Inventory Management System for Growing Corporations**

**Project Aim**: **HissabTrack** provides an inventory management solution that scales with a corporation’s growth. The system will track inventory, manage products, monitor sales, and generate reports, all aimed at improving inventory control and operational efficiency.

## Objectives

The primary objectives of the **HissabTrack** project are:

* **Efficient Inventory Management**: Track and manage inventory in real-time, ensuring accurate stock levels and seamless operations.
* **Sales and Reporting**: Monitor sales data and generate reports to provide actionable insights into inventory performance.
* **Scalability**: Design the system to easily scale as the business grows, capable of handling an increasing number of products and locations.
* **User-Friendly Interface**: Create an intuitive interface that enhances user experience and simplifies inventory management tasks.
* **Data Integrity**: Implement a reliable database system for storing and retrieving inventory and sales data with high accuracy and consistency.

## Problem Statement

Corporations with expanding operations face challenges in managing inventory effectively, leading to issues like stockouts, overstocking, and inefficiencies in sales tracking. Manual systems and spreadsheets are often inadequate for handling large volumes of inventory and transactions, resulting in operational delays and inaccurate stock records.

**HissabTrack** addresses these problems by offering an automated, scalable inventory management system that ensures real-time tracking, accurate reporting, and streamlined product management. The system will provide businesses with the tools to optimize inventory control, enhance decision-making, and scale operations efficiently. The implementation of widely accepted technologies like **Java**, **JavaFX**, and **SQL** ensures the solution is robust, scalable, and cost-effective for growing corporations.

# Overall Description

## Product Perspective

**HissabTrack** is a self-contained inventory management system designed specifically for growing corporations that need a scalable and efficient solution for inventory tracking and product management. Unlike traditional manual methods or outdated inventory software, **HissabTrack** offers a modern, automated solution built using **Java**, **JavaFX**, and **SQL**, which ensures real-time updates, scalability, and user-friendly interface for better inventory control.

The product does not follow any existing product family or replace legacy systems but provides a standalone solution to manage inventory across multiple business locations. **HissabTrack** will not be part of a larger system but will integrate with external tools as needed (such as accounting software or e-commerce platforms) through APIs or data import/export features.

In terms of system architecture, the software will consist of three major components:

1. **User Interface (UI)** – Designed in **JavaFX** to ensure a responsive and intuitive user experience.
2. **Database Management** – **SQL** for reliable and secure storage of inventory data, product records, sales transactions, etc.
3. **Business Logic** – Core functionality written in **Java** to perform tasks such as inventory updates, sales tracking, and report generation.

A simple diagram illustrating the relationship between these components will be provided in the design phase.

## Product Functions

The major functions of **HissabTrack** are summarized as follows:

* **Inventory Tracking**: The system tracks inventory levels for all products in real-time, including updates for stock additions, sales, and returns.
* **Product Management**: Users can add, edit, and remove products from the inventory, assigning relevant details such as price, category, and stock level.
* **Sales Monitoring**: The system records and tracks sales transactions, providing reports on sales trends and product performance.
* **Report Generation**: The system generates various reports, including inventory status reports, sales reports, and low-stock alerts.
* **Role-Based Access Control**: The system allows different levels of access for different user roles, ensuring that only authorized personnel can perform sensitive tasks such as adding products or modifying inventory levels.
* **Scalability**: The software is designed to handle growing inventories and an increasing number of products, supporting businesses as they expand their operations.

These core functions work together to provide businesses with a comprehensive inventory management solution that is scalable, efficient, and easy to use. Further details on each function will be elaborated in Section 3 of the document.

Additionally, a high-level data flow diagram will be used to represent how these functions interact with each other and the system's overall structure. This will provide clarity on the data flow, user interaction, and how the system processes various tasks.

## List of Use Cases

**Admin:**

1. Manage Supplier
   1. Add Supplier
   2. Remove Supplier
   3. Update Supplier
2. Manage Inventory Manager
   1. Add Inventory Manager
   2. Remove Inventory Manager
   3. Update Inventory Manager
3. Pay Invoice
4. Generate Report
5. Update Profile

**Inventory Manager:**

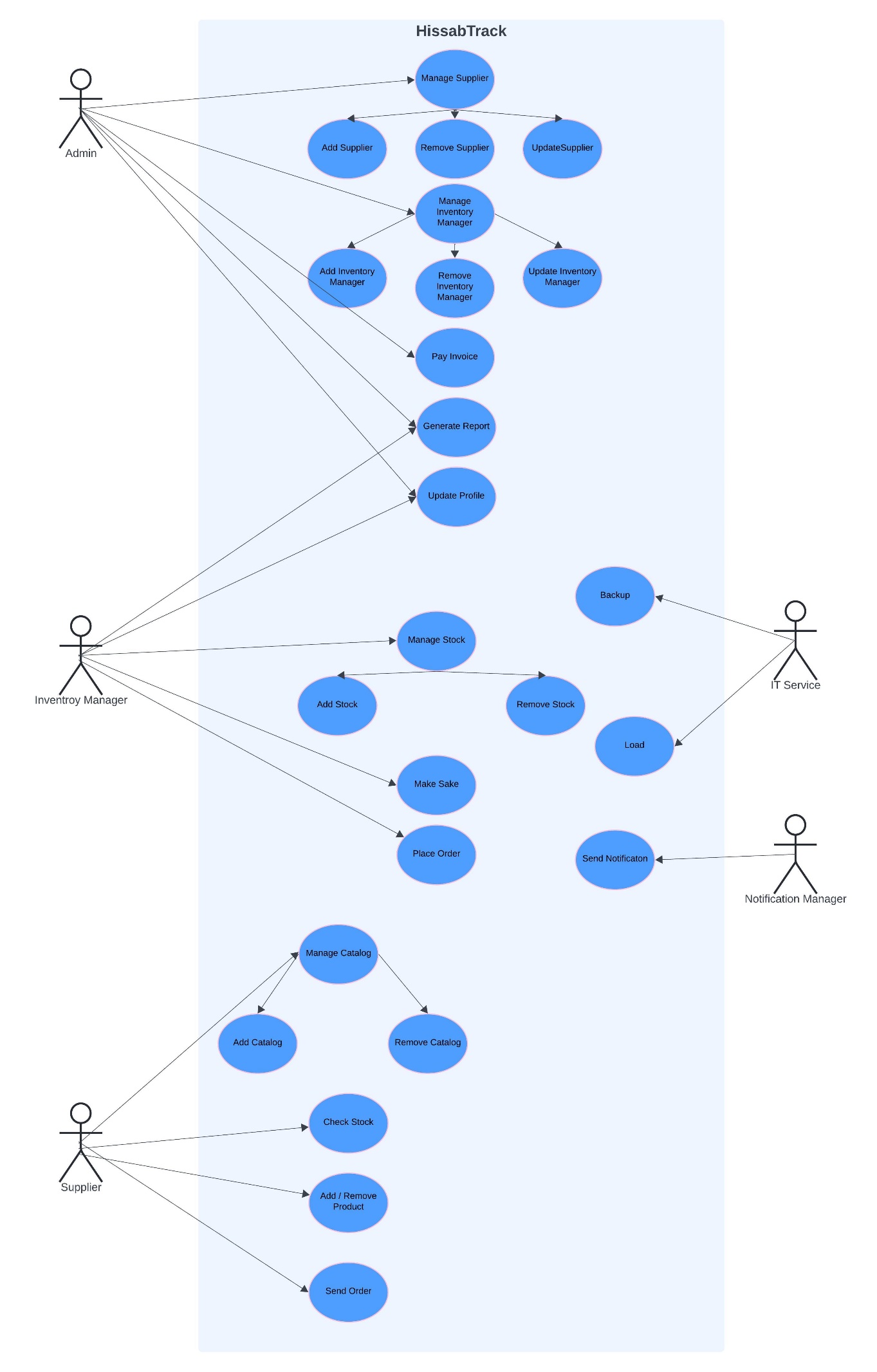
1. Manage Stock
   1. Add Stock
   2. Remove Stock
2. Generate Report
3. Update Profile
4. Make Sale
5. Place Order

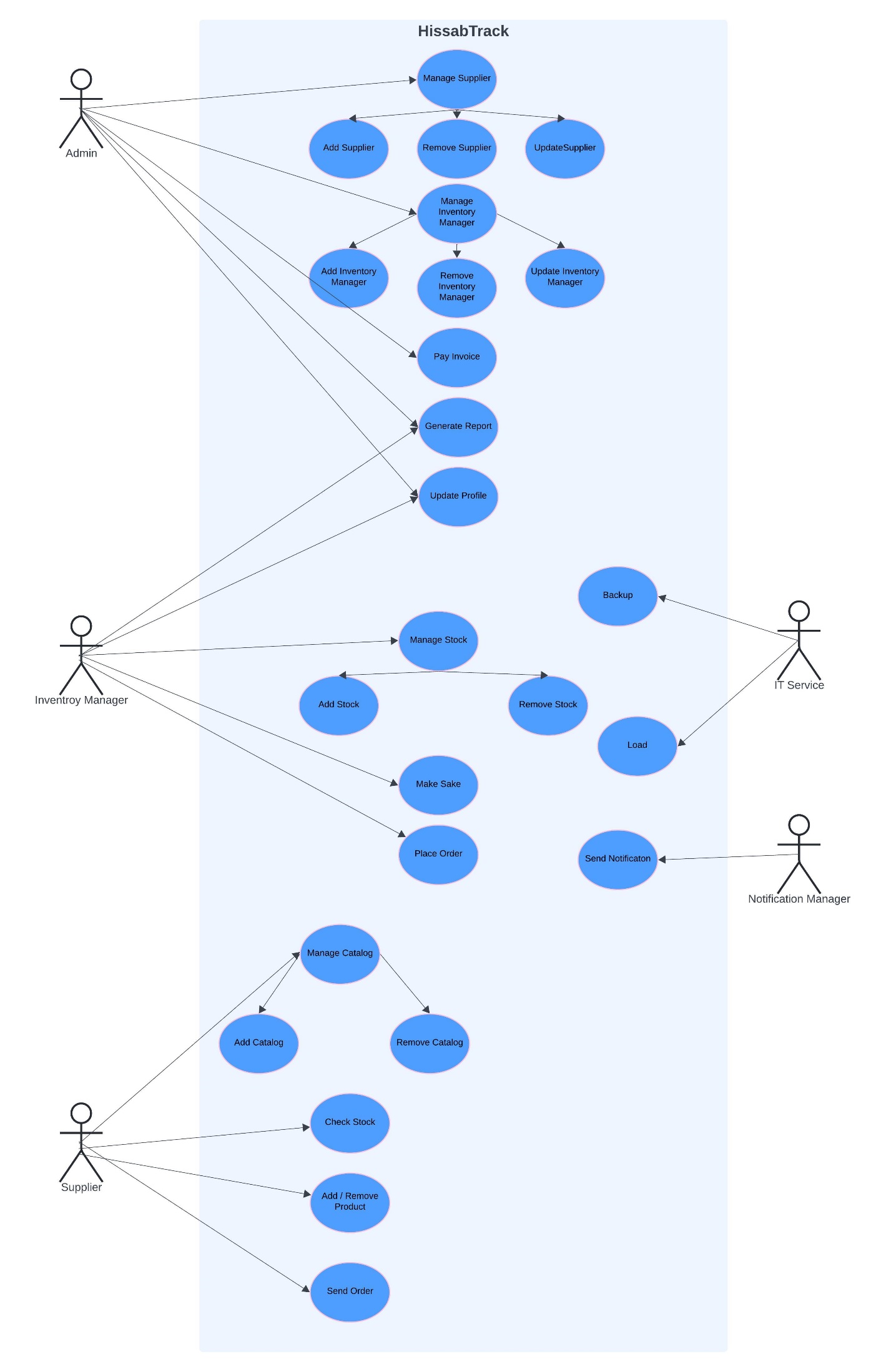
**Supplier:**

1. Manage Product
   1. Add Product
   2. Remove Product
   3. Manage Product
2. Send Order

**IT Service:**

1. BackUp





## Extended Use Cases

**Admin Use Cases:**

**Inventory Manager Use Cases:**

**Supplier Use Cases:**

**Use Case Name:** Manage Items  
**Scope of the System Under Design:** HisaabTrack  
**Level:** User-goal  
**Primary Actor:** Supplier

**Stakeholders and Interests:**

* **Supplier:** Able to manage item data related to their supply, including adding, removing, and updating items they supply.
* **Inventory Manager:** Needs accurate and timely updates to item data provided by suppliers for effective stock and order management.
* **Admin:** Oversees supplier activity to ensure accuracy and compliance with system protocols.

**Preconditions:**

* Supplier is logged in and authorized to manage their items.
* Item data is stored in the system’s database.

**Postconditions:**

* Item records managed by the supplier are updated in the system, including additions, deletions, and modifications.
* The system logs all changes made to the item records by the supplier.
* **Main Success Scenario (Add Item):**

1. Supplier selects the "Manage Item" option.
2. System displays a list of current items associated with the supplier and options to add, remove, or update items.
3. Supplier selects "Add Item."
4. System prompts the supplier to enter the item details (e.g., name, SKU, description, price, quantity).
5. Supplier enters the item information and submits the form.
6. System validates the entered details (e.g., ensuring a unique SKU and compliance with supplier's allowed parameters).
7. Supplier confirms the addition.
8. System adds the new item to the database and associates it with the supplier's account.
9. System logs the action and confirms the item’s successful addition.

**Extensions:**

* **5a. Invalid Data Entered:**
  + If the supplier enters incomplete or invalid data, the system prompts them to enter correct information before proceeding.
* **6a. Item Already Exists:**
  + If an item with the same SKU already exists in the system, the system notifies the supplier and suggests updating the existing record instead.
* **Main Success Scenario (Remove Item):**

1. **Supplier** selects the "Manage Item" option.
2. System displays a list of current items associated with the supplier.
3. Supplier selects an item from the list and chooses the "Remove Item" option.
4. Supplier confirms the removal.
5. System verifies if the item is associated with any active orders or stock.
6. System removes the item from the database after confirming it is not linked to any active orders or inventory stock.
7. System logs the action and confirms the successful removal of the item.

**Extensions:**

* **6a. Item Linked to Active Order:**
  + If the item is associated with active orders or is still in stock, the system informs the supplier and blocks removal.
* **Main Success Scenario (Update Item):**

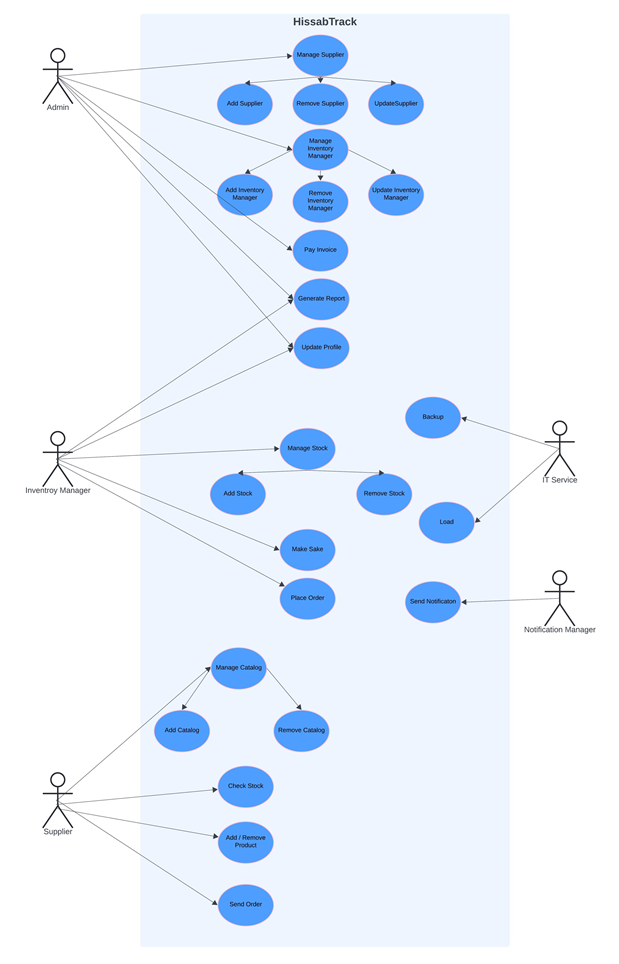
1. **Supplier** selects the "Manage Item" option.
2. System displays a list of current items associated with the supplier.
3. Supplier selects an item from the list and chooses the "Update Item" option.
4. System displays the current details of the selected item.
5. Supplier edits the required information (e.g., updating the price, description, or stock level).
6. System validates the changes to ensure compliance with supplier parameters.
7. Supplier confirms the update.
8. System updates the item’s record in the database and associates the update with the supplier’s account.
9. System logs the action and confirms the successful update of the item’s information.

**Extensions:**

* **5a. Invalid Data Entered:**
  + If the supplier enters incomplete or invalid data, the system prompts them to enter correct information before proceeding.

**IT Service Use Cases:**

## Use Case Diagram



**3. Other Nonfunctional Requirements**

**3.1 Performance Requirements**

* The system should respond to user actions (e.g., adding a product, updating inventory, or generating a report) within **2 seconds** under normal operating conditions.
* The database should support up to **10,000 inventory items** and process up to **100 concurrent users** without significant performance degradation.
* For batch processes, such as generating large reports, the system must complete the task within **5 seconds** for datasets of up to **50,000 records**.
* Real-time updates (e.g., stock level adjustments) should reflect changes across the system within **1 second** after the action is performed.
* The system must maintain a consistent uptime of **99.9%**, ensuring availability even under peak loads.

**3.2 Safety Requirements**

* The software must provide safeguards to prevent unauthorized deletion or modification of critical inventory data.
* In case of database corruption or failure, the system must maintain a **backup and recovery mechanism** to restore data up to the last recorded transaction.
* Safety mechanisms should be in place to prevent actions that could cause irreversible loss of data, such as confirmation prompts before deletions.
* The product must comply with applicable data safety regulations (e.g., **ISO/IEC 27001**) to mitigate risks associated with data breaches or accidental loss.

**3.3 Security Requirements**

* **User Authentication**: All users must log in with unique credentials. Passwords should be encrypted using industry-standard hashing algorithms (e.g., **bcrypt**).
* **Role-Based Access Control (RBAC)**: Specific functionalities should be restricted based on user roles, ensuring users access only the features necessary for their responsibilities.
* **Data Encryption**: Sensitive data such as user credentials and inventory details should be encrypted in transit and at rest.
* **Activity Logging**: The system must maintain logs of user activities, such as changes to inventory or generated reports, for auditing purposes.
* **Compliance**: The system must adhere to regulations such as **GDPR** or any regional equivalent for data privacy and security.

**3.4 Software Quality Attributes**

* **Adaptability**: The software must be scalable to accommodate additional inventory items, new users, or features as the business grows.
* **Availability**: The system should provide uninterrupted service with minimal downtime (target uptime: **99.9%**).
* **Correctness**: All operations, including stock updates and report generation, must execute with absolute accuracy.
* **Flexibility**: The software should support customization options, such as configurable reports and product categories.
* **Maintainability**: The system's architecture must be modular to simplify debugging, updates, and enhancements.
* **Portability**: The application should run seamlessly on all major operating systems (e.g., Windows, macOS, Linux).
* **Reliability**: The software should handle exceptions gracefully, ensuring stable performance even during unexpected conditions.
* **Usability**: The user interface must be intuitive, ensuring minimal training is required for new users.

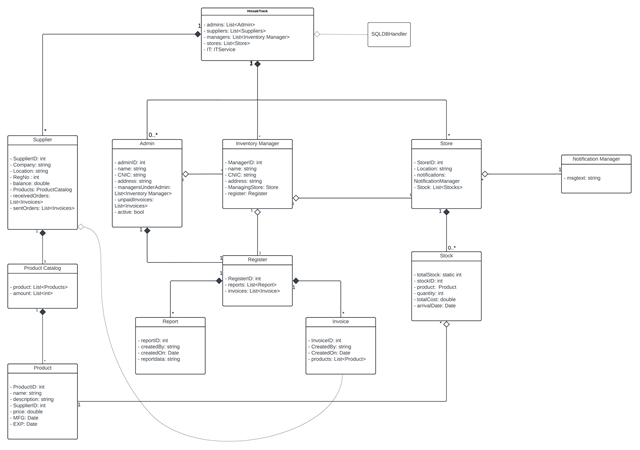
**3.5 Business Rules**

* Only authenticated users with the necessary permissions can perform tasks such as adding inventory items, updating stock levels, or generating reports.
* Critical operations (e.g., deleting inventory items) must require confirmation to prevent accidental actions.
* All data entries must pass validation checks to ensure data integrity (e.g., stock quantity cannot be negative).
* The system must enforce strict session timeout rules, logging out inactive users after a predefined period (e.g., **15 minutes**) to enhance security.
* Reports and other generated documents should carry timestamps for reference and auditing purposes.
* **3.6 Operating Environment**
* **Hardware**: The system should run on hardware with at least:
  + **Processor**: Dual-core processor or higher
  + **RAM**: 4GB minimum (8GB recommended)
  + **Storage**: 500MB for application installation, plus additional space for database storage based on inventory size.
* **Operating System**: The system should support:
  + Windows 10 or later
  + macOS 10.14 or later
  + Any modern Linux distribution with Java support.
* **Software Dependencies**:
  + **Java SE Development Kit (JDK) 11 or later**
  + **JavaFX** runtime
  + **SQL Database Management System** (e.g., MySQL, PostgreSQL).

**3.7 User Interfaces**

* **Graphical User Interface (GUI)**:
  + The GUI will adhere to modern design principles for ease of navigation and accessibility.
  + Standard elements such as a **navigation bar**, **search fields**, and **action buttons** (e.g., Add, Edit, Delete) will be present on relevant screens.
  + Error messages and confirmation dialogs will be displayed in a user-friendly and informative manner.
  + A consistent color scheme, typography, and button styles will be maintained across all screens for uniformity.
* **Standard Features**:
  + **Help Button**: Links to user manuals and FAQs.
  + **Keyboard Shortcuts**: Frequently used operations (e.g., Ctrl+S for saving changes).
  + **Responsive Design**: The interface will adapt to various screen sizes to ensure usability on both desktops and laptops.
* **Screens**:
  + **Dashboard**: Displays key performance indicators and quick actions for inventory management.
  + **Inventory Management Screen**: Provides options for searching, adding, editing, and viewing inventory details.
  + **Reports Screen**: Allows users to generate and export sales or inventory reports.

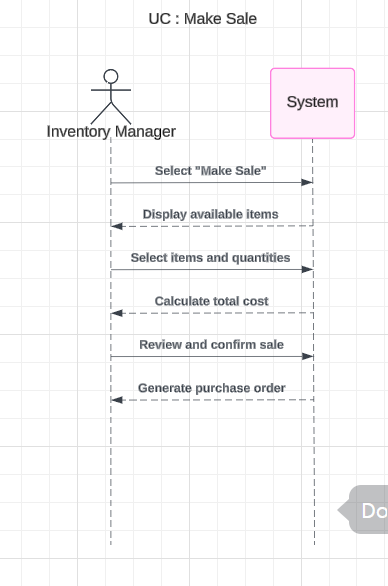
# Domain Model



# System Sequence Diagram

**Make Sale:**

**SSD:**



A diagram of a diagram

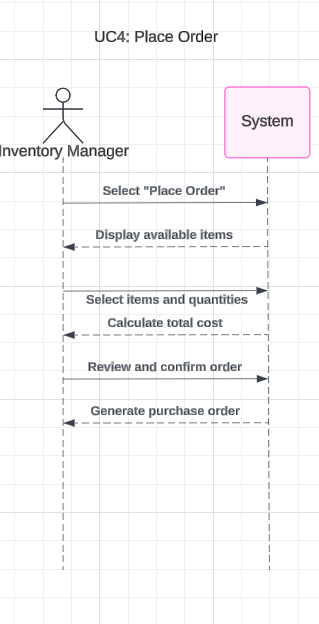
Description automatically generated with medium confidence**SD:**

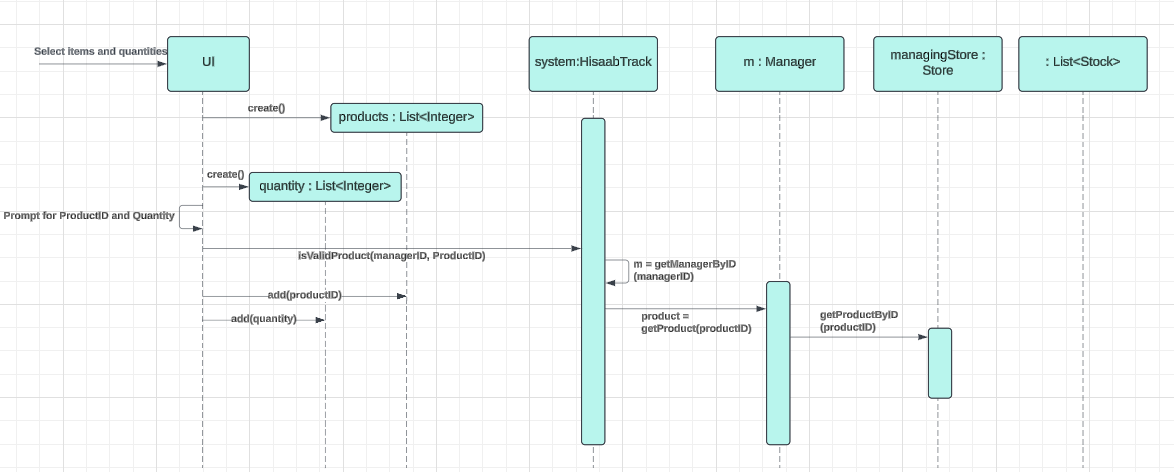
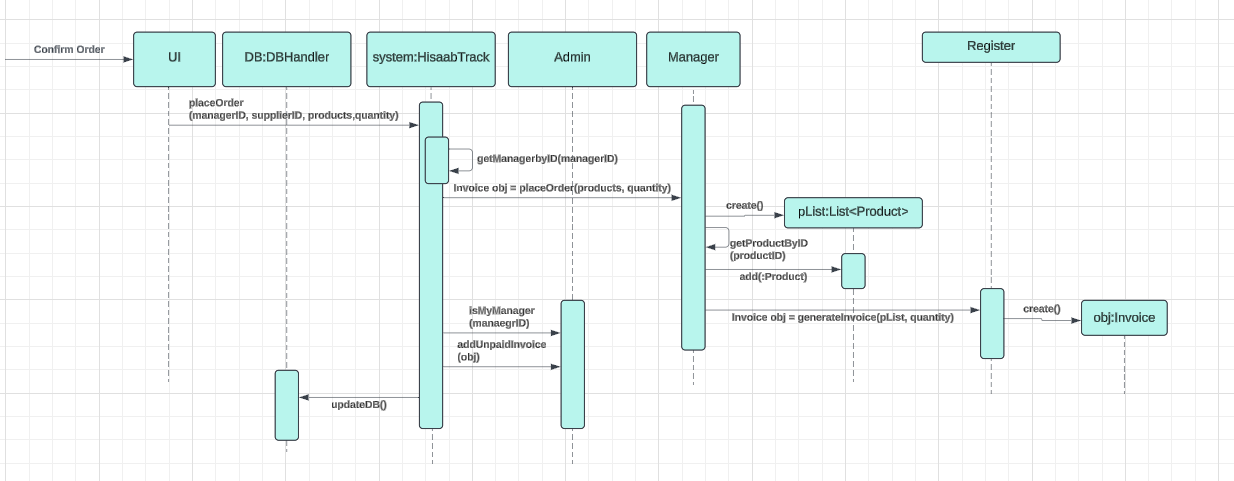
A diagram of a program

Description automatically generatedA diagram of a software development

Description automatically generated

**Place Order:**

****SSD:

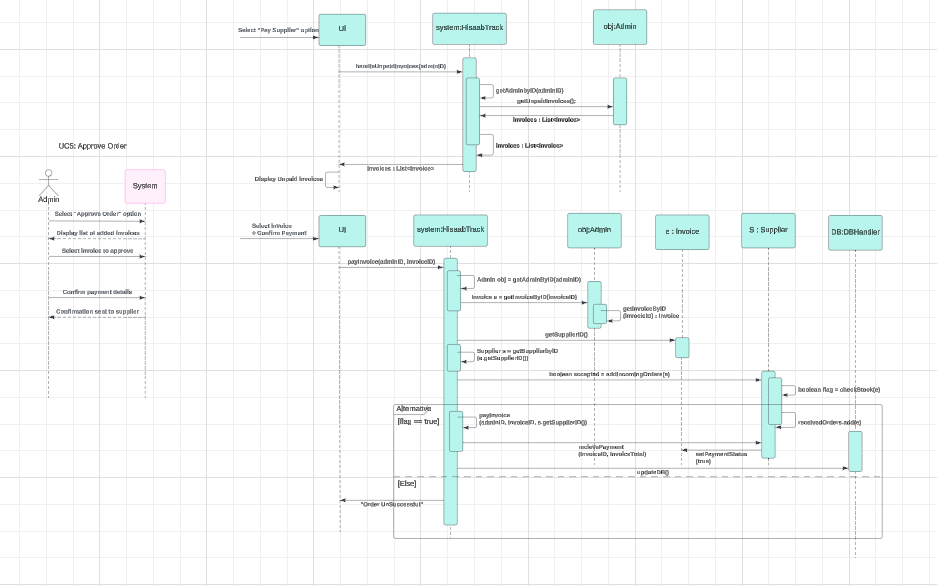
****A diagram of a system

Description automatically generated**SD:**

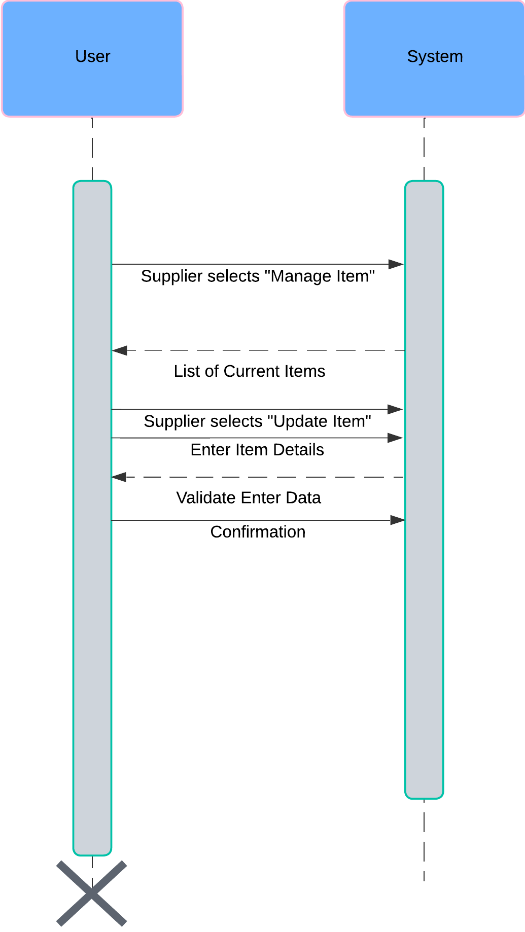
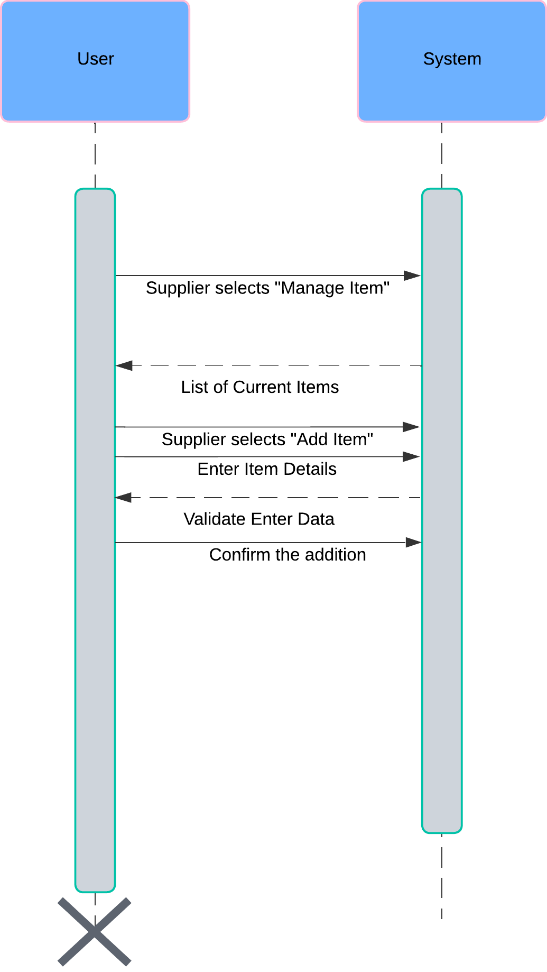
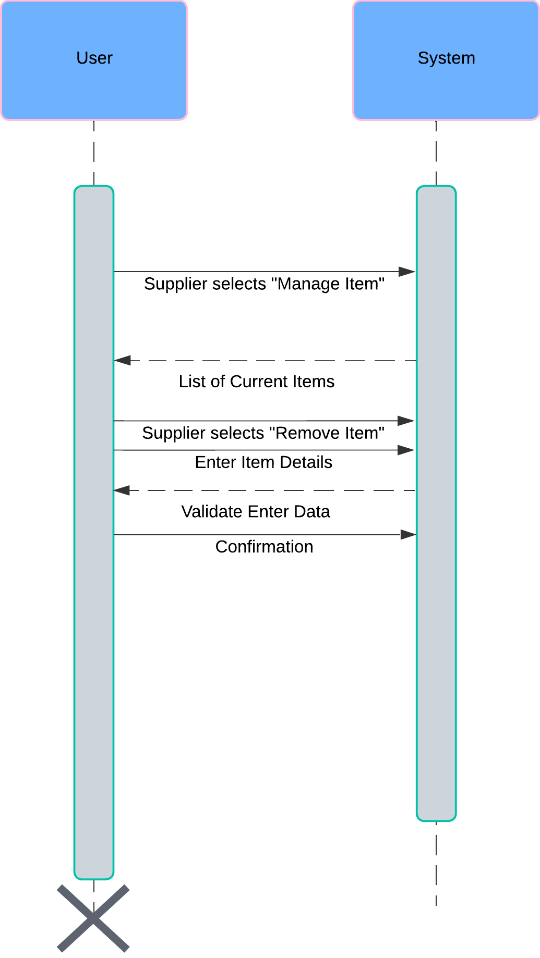
A diagram of a project

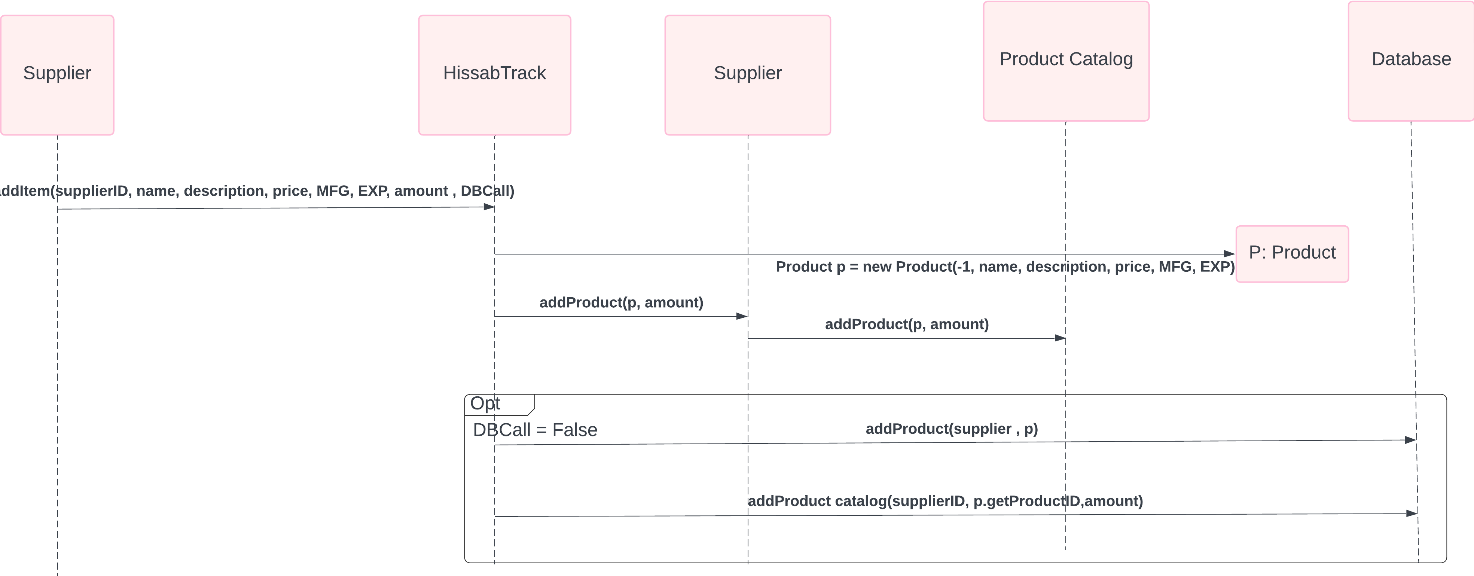
Description automatically generated with medium confidence**Send Order:**

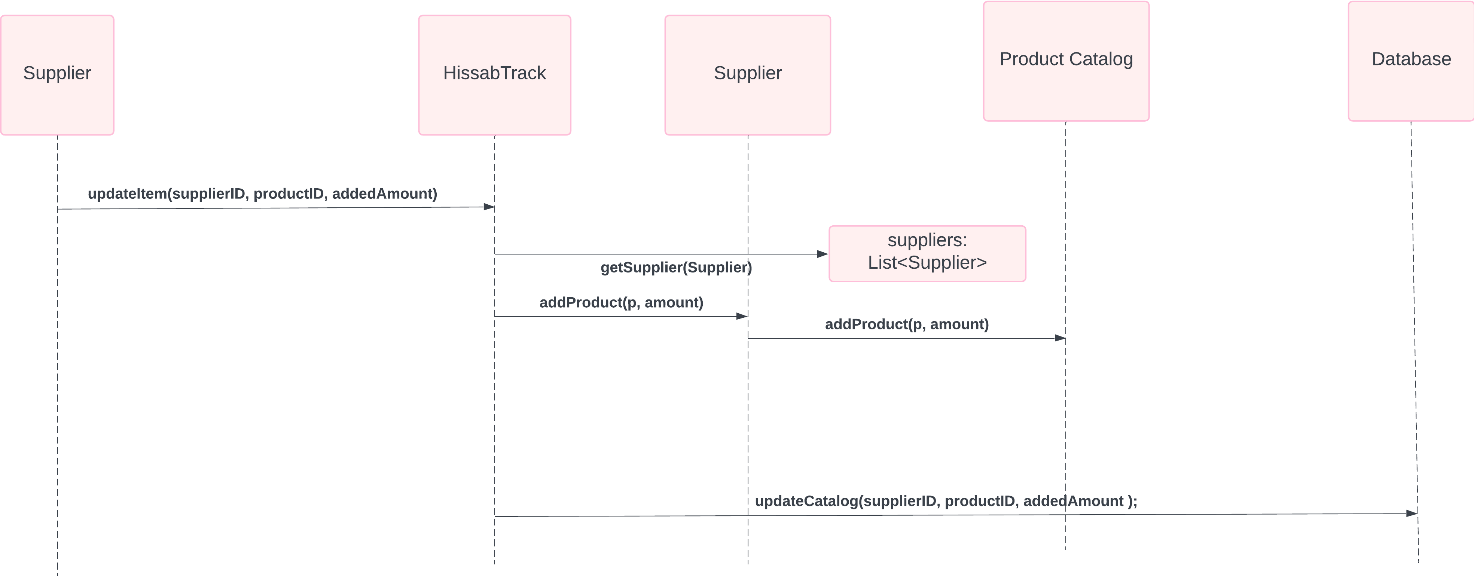
**Approve Order:**



**Manage Product:**

**SD:**

**SSD:**

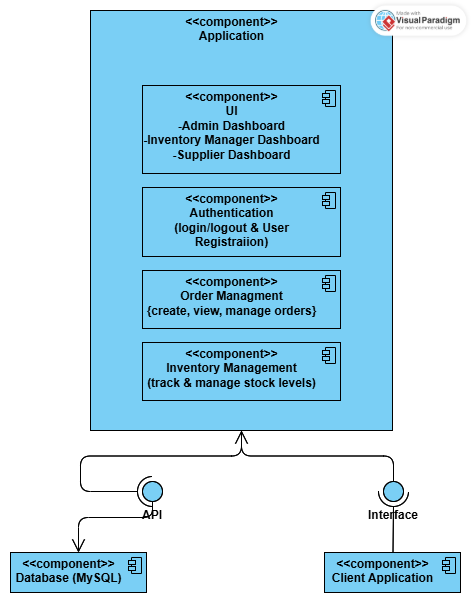
****

A screenshot of a computer

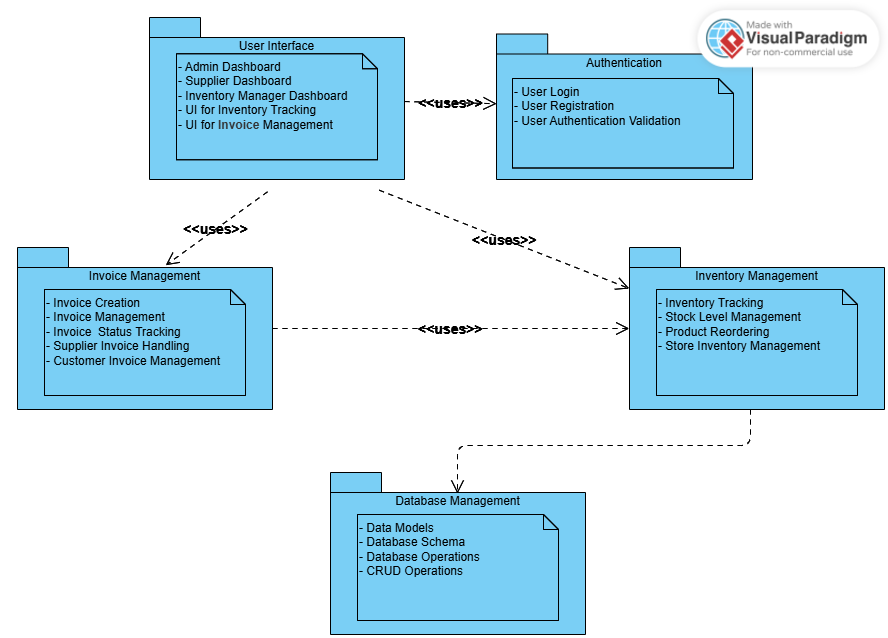
Description automatically generated

# Class Diagram

# Component Diagram



# Package Diagram



# Deployment Diagram

