**Software Requirements Specification**

**for**

**StockWise**

**Version 1.0 approved**

**Prepared by:**

22K-4477 Syed Muqeet-Ur Rehman

22K-4487 Muhammad Usman Sohail

22K-4658 Syed Hussamuddin

**FAST NUCES Main Campus, Karachi**

**Date created:** 09-05-2025

**Table Of Contents**

[**1.**](#_heading=h.gjdgxs) **Introduction 3**

[1.1. Purpose 4](#_heading=h.30j0zll)

[1.2. Intended Audience and Reading Suggestions 4](#_heading=h.1fob9te)

[1.3. Product Scope 4](#_heading=h.3znysh7)

[1.4. References 4](#_heading=h.2et92p0)

[**2.**](#_heading=h.tyjcwt) **Overall Description 5**

[2.1. Product Perspective 5](#_heading=h.3dy6vkm)

[2.2. Product Functions 5](#_heading=h.1t3h5sf)

[2.3. User Classes and Characteristics 5](#_heading=h.4d34og8)

[2.4. Operating Environment 6](#_heading=h.2s8eyo1)

[2.5. Design and Implementation Constraints 6](#_heading=h.17dp8vu)

[2.6. User Documentation 6](#_heading=h.3rdcrjn)

[2.7. Assumptions and Dependencies 6](#_heading=h.26in1rg)

[2.8. Process model 7](#_heading=h.lnxbz9)

[2.9. Project Plan 7](#_heading=h.35nkun2)

[2.10. Feasibility Report 8](#_heading=h.1ksv4uv)

[2.11. Homogenization Plan 8](#_heading=h.44sinio)

[2.12. Use Case descriptions 8](#_heading=h.2jxsxqh)

[2.13. Video of Working Software 9](#_heading=h.z337ya)

[**3.**](#_heading=h.3j2qqm3) **External Interface Requirements 9**

[3.1. User Interfaces 9](#_heading=h.1y810tw)

[3.2. Hardware Interfaces 9](#_heading=h.4i7ojhp)

[3.3. Software Interfaces 9](#_heading=h.2xcytpi)

[3.4. Communications Interfaces 9](#_heading=h.1ci93xb)

[**4.**](#_heading=h.3whwml4) **System Features 9**

[4.1. Inventory Monitoring 9](#_heading=h.2bn6wsx)

[4.2. Low Stock Alerts 10](#_heading=h.qsh70q)

[4.3. Supplier Collaboration 10](#_heading=h.3as4poj)

[4.4. Order Management 11](#_heading=h.1pxezwc)

[4.5. Sales Reporting 11](#_heading=h.49x2ik5)

[4.6. Role-based Access Control 12](#_heading=h.2p2csry)

[4.7. Integration with External Systems 13](#_heading=h.147n2zr)

[**5.**](#_heading=h.3o7alnk) **Other Nonfunctional Requirements 13**

[5.1. Performance Requirements 13](#_heading=h.23ckvvd)

[5.2. Safety Requirements 14](#_heading=h.ihv636)

[5.3. Security Requirements 14](#_heading=h.32hioqz)

[5.4. Software Quality Attributes 15](#_heading=h.1hmsyys)

[5.5. Business Rules 15](#_heading=h.41mghml)

[**6.**](#_heading=h.2grqrue) **Other Requirements 15**

[**Appendix: Analysis Models 16**](#_heading=h.vx1227)

# Introduction

## Purpose

This document specifies the software requirements for the "StockWise" (SW) version 1.0. This system is designed to streamline inventory tracking and management processes for medium to large-scale organizations. The scope of this document includes all the key functionalities and features of the SW, such as inventory monitoring, low-stock alert generation, supplier collaboration, order management, and sales reporting.

This SRS covers the complete system, including its subsystems like user management, inventory management, and alert generation, ensuring a cohesive and detailed understanding of the requirements for development, testing, and deployment. It serves as a foundation for communication between stakeholders, developers, and end users.

## Intended Audience and Reading Suggestions

This SRS is intended for developers, project managers, testers, end users, marketing staff, and documentation writers. Developers will use it to guide implementation, while testers will create test cases based on the system features and requirements. Project managers will use the document to align development with project goals, and marketing staff can ensure the product meets market needs. End users, such as admins and stock managers, will find relevant functionality outlined in the product functions and user interfaces sections. Documentation writers can reference it to create user manuals and help guides. Readers are encouraged to begin with the introduction and overall description, then focus on the sections most relevant to their role, such as system features, external interfaces, or nonfunctional requirements.

## Product Scope

The StockWise (SW) is designed to streamline inventory tracking, order management, and stock monitoring for retail businesses. Its primary goal is to enhance operational efficiency by automating inventory processes, reducing stock discrepancies, and ensuring timely alerts for low-stock items. The system benefits businesses by minimizing inventory wastage, optimizing restocking decisions, and improving overall supply chain management. SW supports key business strategies such as cost reduction, operational transparency, and improved decision-making through data analytics. By integrating features like real-time stock updates, sales reporting, and user-specific functionalities for admins, stock managers, and suppliers, it aligns with the corporate goal of leveraging technology to drive business success and customer satisfaction.

## References

* European Commission. (2020). **General Data Protection Regulation (GDPR).** <https://gdpr-info.eu/>
* California Legislative Information. (2020). **California Consumer Privacy Act (CCPA).** <https://oag.ca.gov/privacy/ccpa>
* Oracle. (2020). **PostgreSQL Documentation.** <https://www.postgresql.org/docs/>
* Open Web Application Security Project (OWASP). (2021). **OWASP Top Ten.** <https://owasp.org/www-project-top-ten/>
* Jovanovic, M., & Milinkovic, D. (2018). **Software Security: Threats and Vulnerabilities.** Springer.
* International Organization for Standardization (ISO). (2018). **ISO/IEC 27001:2013 – Information Security Management Systems.** <https://www.iso.org/isoiec-27001-information-security.html.>

# Overall Description

## Product Perspective

The StockWise (SW) is a new, self-contained software solution designed to replace traditional manual inventory processes and fragmented tools. It serves as a comprehensive platform for managing inventory, integrating seamlessly with external systems such as point-of-sale (POS) systems and supplier databases. By providing a centralized interface for stock tracking, order management, and alert generation, SW ensures smooth interconnectivity between users, devices, and the overall business workflow.

## Product Functions

The StockWise (SW) enables the following major functions:

* **Inventory Tracking:** Monitor stock levels in real-time and track product movement across storage locations
* **Order Management:** Manage customer orders, generate purchase orders, and process supplier transactions
* **User Roles:** Support multiple user roles, such as Admin, Stock Manager, and Customer, each with specific access privileges
* **Alert System:** Generate automated notifications for low stock levels and inventory discrepancies.
* **Report Generation:** Provide detailed sales and inventory reports for analysis and decision-making.
* **Secure Access:** Ensure secure access to the system with user authentication and role-based permissions.
* **Integration:** Seamlessly integrate with external systems, such as POS terminals and supplier platforms.

A use-case diagram is included to show user interactions with these functionalities. **(Appendix, Diagram 1)**

## User Classes and Characteristics

The StockWise (SW) is designed for several distinct user classes, each with different access levels, roles, and interactions with the system:

* **Admin:** Manages system settings, user roles, and generates reports. Has full access to all functionalities and settings.
* **Stock Manager:** Handles inventory tracking, low-stock alerts, and product updates. Limited to inventory-related tasks and order management.
* **Supplier:** Fulfills purchase orders and updates stock availability. Has access to order management and product details only.
* **Customer:** Places and tracks orders. Has access to product browsing and order management features only.

Each user class has a tailored set of features and permissions based on their role, ensuring that the system meets security and functionality requirements for a variety of users. The **Admin** class is the most critical for overall system management, while the **Customer** class is less involved in system operations but essential for daily transactions.

## Operating Environment

The StockWise (SW) is designed to operate on both desktop and mobile devices, supporting standard configurations with at least an Intel i5 processor, 4GB of RAM, and 500GB of storage. The system will be hosted on Linux-based servers (Ubuntu 20.04 or later, CentOS 7 or later) and is compatible with client devices running Windows 10+, macOS, Android, and iOS. The backend of the application will utilize PostgreSQL databases, while the web server will be powered by Apache or Nginx. The frontend will use ReactJS, and the system will integrate with external tools like POS systems, ERP solutions, and payment gateways to ensure smooth data flow and operational efficiency.

## Design and Implementation Constraints

The development of the StockWise (SW) will be subject to several constraints. The system must comply with the organization’s security policies, including data encryption and user authentication standards. It must integrate seamlessly with existing POS and ERP systems, using RESTful APIs for communication. The software will be built using Node.js for the backend and ReactJS for the frontend, with PostgreSQL as the database. Performance requirements dictate that the system must handle up to 10,000 concurrent users. Additionally, the software must be compatible with cross-platform devices, including desktops and mobile devices, and adhere to the company’s design conventions and coding standards. Security features such as two-factor authentication and role-based access control are mandatory.

## User Documentation

The following user documentation components will be delivered alongside the StockWise (SW) to ensure proper system usage and facilitate training:

1. **User Manual**: A comprehensive guide detailing system features, functionalities, and step-by-step instructions for users. This manual will cover common tasks such as order placement, inventory management, and report generation. It will be available in **PDF** format for easy access and printing.
2. **Online Help**: A context-sensitive help system integrated into the application, offering quick access to documentation for users while they are navigating the system. This will provide searchable articles, FAQs, and troubleshooting tips.
3. **Video Tutorials**: Short, easy-to-follow videos demonstrating how to perform key tasks within SW, such as setting up user roles, processing orders, and generating reports. These will be available via the company's internal website or a video platform like YouTube.
4. **Administrator Guide**: A specialized document for system administrators, detailing setup, configuration, user management, and system maintenance procedures. This will also be available in PDF format.
5. **Training Materials**: Slide decks and supplementary documents designed for training new users and staff, ensuring they can quickly understand and adopt the system.

## Assumptions and Dependencies

The development of the StockWise (SW) assumes that reliable internet connectivity will be available for all users to ensure real-time inventory tracking and order management. It is also assumed that third-party tools, such as POS systems and ERP integrations, will be compatible with the RESTful API standards we intend to use. The system will depend on PostgreSQL databases, which must be configured and maintained according to the specifications provided. Additionally, we assume that the user base will have access to modern devices with up-to-date operating systems. The project is also dependent on the availability of external payment gateways and supplier platforms for order processing and stock updates. Any changes or incompatibility with these external components could affect the project timeline and functionality.

## Process model

The StockWise (SW) will be developed using an Agile development methodology**,** with iterative sprints allowing flexibility and continuous improvement throughout the project lifecycle. The project will follow a two-week sprint cycle**,** where specific features and functionalities will be planned, developed, and tested. Regular daily stand-up meetings will ensure that the team stays aligned and can quickly address any issues. At the end of each sprint, there will be a review and retrospective to assess progress and incorporate feedback, ensuring the system meets evolving user needs. Test-driven development (TDD) will be employed to ensure high-quality code, with unit tests covering critical functionality. This approach allows for frequent releases, enabling stakeholders to see incremental progress, while ensuring that changes or new features are well-tested and meet requirements before deployment.

## Project Plan

The development of the StockWise (SW) will be carried out in a structured manner, following a clear timeline with key milestones:

1. **Requirements Gathering (2 weeks)**:
   * Finalize functional and nonfunctional requirements.
   * Gather stakeholder feedback and finalize system specifications.
2. **Design and Architecture (3 weeks)**:
   * Design system architecture, including database schema and high-level component design.
   * Prepare use case diagrams, data flow diagrams, and other supporting models.
3. **Development Phase (6 weeks)**:
   * **Sprint 1**: Implement core features, including inventory tracking and user roles.
   * **Sprint 2**: Develop order management system, alerts, and supplier integrations.
   * **Sprint 3**: Integrate reporting functionality, finalize user interface, and backend API.
   * Continuous testing and bug fixing during development.
4. **Testing Phase (4 weeks)**:
   * Perform unit testing, integration testing, and system testing.
   * User acceptance testing (UAT) with stakeholders.
5. **Deployment (2 weeks)**:
   * Deploy the application to production environment.
   * Provide user training and documentation.
   * Collect feedback for future enhancements.
6. **Post-Deployment Support (Ongoing)**:
   * Monitor system performance and user feedback.
   * Address any critical issues or enhancements.

This timeline is flexible and will be adjusted as necessary based on project needs, stakeholder feedback, and unforeseen challenges.

## Feasibility Report

The StockWise (SW) uses ReactJS for the frontend, Node.js for the backend, and PostgreSQL for the database, offering scalability and cross-platform compatibility. It is expected to support up to 10,000 concurrent users and deliver a strong ROI, projected at 150% within the first year due to automation and improved efficiency. The system is user-friendly and integrates well with existing POS and ERP systems.

However, the system may face challenges during initial setup, including the integration of existing systems and user training. Scaling beyond the expected user load or large inventories may require additional infrastructure. Security concerns, particularly around sensitive data, will require ongoing attention. Despite these drawbacks, SW is technically sound, financially feasible, and operationally effective.

## Homogenization Plan

The Homogenization Plan for the StockWise (SW) focuses on ensuring consistent and accurate data across various modules and third-party integrations. This will be achieved through automated data validation and standardization processes that normalize data inputs from different sources, such as suppliers and external systems. Regular data audits will be conducted to ensure integrity and consistency across the system, addressing discrepancies such as mismatched product information, inventory counts, and order histories. Additionally, all data entered into the system will be standardized to a common format to prevent errors and ensure seamless integration. This plan also includes user training to ensure that the data entry processes are consistent and adhere to established standards, further reducing the risk of data inconsistency.

## Use Case descriptions

**1 Place Order**

Customer logs in to their account, and searches for desired products. They then browse through the available products and add them to their cart, and proceed to checkout. During checkout, the customer provides payment information, which the system validates before confirming the order. Once the order is confirmed, the system updates order status to “Processing”. This allows customer to track the order status.

**2 Update Inventory**

The stock manager logs in to their account and views any inventory alerts indicating low stock level for a product. Stock manager then generates a purchase order for the product, and the supplier responds by supplying the products and sending the invoice. Stock manager then restocks the item, the system validates this and shows updates in stock levels in real time.

**3 Generate Sales Report**

Admin logs into the system and selects the type of reports they wish to generate. After specifying the desired date range for the report, the system processes the data and generates the report. The user can then review the information and, if needed, export the report in various formats, such as PDF or CSV.

## Video of Working Software

# External Interface Requirements

## User Interfaces

The StockWise (SW) will feature a clean, intuitive user interface designed to accommodate various user roles, including Admin, Stock Manager, Supplier, and Customer. The interface will follow modern GUI standards, ensuring a seamless experience across desktop, tablet, and mobile devices. A fixed navigation bar will provide easy access to key sections like inventory management, orders, reports, and settings. Standard buttons such as "Save," "Cancel," and "Submit" will appear consistently across screens, and every page will include a help icon for quick access to documentation and FAQs. Clear and concise error messages will be displayed for invalid actions, with suggestions for resolution. Keyboard shortcuts for common tasks (e.g., "Ctrl + S" for saving) will enhance efficiency. The design will prioritize accessibility, ensuring the system is usable for all users. Detailed specifications of the user interface design will be provided in a separate document.

## Hardware Interfaces

The StockWise (SW) will interact with various hardware components to support its operations. The system will be compatible with standard devices, including desktop computers, laptops, tablets, and mobile phones. The software will communicate with hardware through standard input/output devices like keyboards, mice, and touchscreens for user interaction. It will also interface with barcode scanners and receipt printers in the case of inventory updates and order processing. Data communication between the software and these hardware devices will use common protocols such as USB for peripherals like barcode scanners and Wi-Fi or Ethernet for networked devices. The system will ensure real-time updates and reliable data transfer between the software and hardware components, supporting seamless integration across all devices.

## Software Interfaces

The StockWise (SW) will interface with several key software components to ensure seamless functionality. It will use PostgreSQL (version 13 or higher) as its database for storing product, order, and inventory data. The system will operate on Linux-based operating systems(Ubuntu 20.04 or later) and utilize RESTful APIs to integrate with external tools, such as ERP systems, POS platforms, and payment gateways. Data exchanged between SW and these systems will include inventory updates, order statuses, and payment confirmations, typically in JSON format.

To support secure financial transactions, SW will integrate with third-party libraries, such as OAuth 2.0 for user authentication and Stripe or PayPal for payment processing. Data sharing between components will be handled through API calls or webhooks, ensuring that information is consistently synchronized across systems. All communications will be encrypted using TLS/SSL protocols to protect sensitive data. Additionally, the backend will manage global data handling to ensure consistency and reliability in multitasking environments. These interfaces ensure smooth operation and integration with external systems while maintaining data integrity and security.

## Communications Interfaces

The StockWise (SW) will utilize HTTP/HTTPS for secure communication between the client and server. RESTfulAPIs will be employed for communication with external systems such as ERP, POS platforms, and payment gateways, exchanging data in JSON format. For notifications, emailalerts will be sent for events like low stock or order updates, formatted in HTML for clarity. Communication will be encrypted using TLS**/**SSL to ensure data security. Real-time updates, such as inventory changes, will use WebSockets or HTTPlong**-**polling to keep users informed without refreshing the page. The system is designed to ensure fast data transfer, processing requests within **2** seconds for 90% of transactions, with reliable synchronization to maintain data consistency across the system.

# System Features

## Inventory Monitoring

This feature allows the system to monitor and display real-time stock levels for each product in the inventory. This feature ensures that stock levels are accurately updated and reflects changes immediately after any transactions, such as sales or restocks. The feature also includes functionality to search for products by various attributes (e.g., product name, category, or SKU). This feature has high priority, as it is critical for business operations and could lead to stockout and lost sales if inaccurate. The functional requirements for this feature are as follows:

**REQ-1**: The system must allow theStock Manager to view the current stock levels of all products in real-time.  
**REQ-2**: The system must automatically update stock levels whenever a transaction occurs (e.g., order placement, restocking, returns).  
**REQ-3**: The system must allow the Stock Manager to manually update the stock level for any product.  
**REQ-4**: The system must support searching for products by various criteria, including name, SKU, or category.  
**REQ-5**: If an invalid stock level update is entered (e.g., negative quantity), the system must prompt an error message indicating that the stock level cannot be negative.  
**REQ-6**: The system must provide an option to filter and sort products based on their stock levels, such as showing low-stock items first.

## Low Stock Alerts

The Low Stock Alert feature automatically notifies the Stock Manager when the inventory level of a product falls below a predefined threshold, ensuring timely restocking actions to prevent stockouts and maintain optimal inventory levels. Alerts are sent via system notifications or email to enable a quick response. The feature is of high priority, as it is critical to prevent stockouts and disruptions, which can result in missed sales and customer dissatisfaction. The functional requirements are as follows:

**REQ-1**: The system must allow the Stock Manager to configure low-stock thresholds for each product.  
**REQ-2**: The system must monitor inventory levels in real-time and compare them to the predefined thresholds.  
**REQ-3**: When a product's stock falls below the threshold, the system must automatically trigger an alert.  
**REQ-4**: The system must send the low-stock alert to the Stock Manager via email or system notification.  
**REQ-5**: The system must allow the Stock Manager to adjust alert thresholds at any time.  
**REQ-6**: If a product has no threshold set, the system must not trigger any alerts for that product.  
**REQ-7**: The system must log all triggered low-stock alerts for reporting and audit purposes.

## Supplier Collaboration

The Supplier Collaboration feature enables seamless interaction between the Stock Manager and suppliers, allowing the system to generate and send purchase orders directly to suppliers when stock levels fall below a certain threshold. It streamlines the process of replenishing inventory and ensures that stock is consistently available. This feature is of medium priority, as it is essential for maintaining supply chain efficiency but does not impact the core functionality of inventory tracking. Functional requirements for this feature are:

**REQ-1**: The system must allow the Stock Manager to create and send purchase orders to suppliers directly from the platform.  
**REQ-2**: The system must automatically trigger purchase orders when stock levels fall below the configured threshold.  
**REQ-3**: The system must provide a way for suppliers to update their delivery status and inventory upon fulfillment of an order.  
**REQ-4**: The system must notify the Stock Manager when a purchase order is successfully sent to a supplier.  
**REQ-5**: The system must allow the Stock Manager to view past purchase orders, including details such as order date, product quantities, and supplier information.  
**REQ-6**: The system must log all supplier interactions and updates for reporting and audit purposes.  
**REQ-7**: If a purchase order cannot be sent (due to system errors or connectivity issues), the system must notify the Stock Manager with an error message.

## Order Management

The Order Management feature allows the system to manage the entire lifecycle of customer orders, from creation to tracking and fulfillment. It enables customers to place orders, and the system to process, update, and track the status of each order, ensuring smooth and efficient operations. This feature is of high priority as it directly impacts customer satisfaction and business revenue. The functional requirements are as follows:

**REQ-1**: The system must allow Customers to browse products, add them to their cart, and place an order.  
**REQ-2**: The system must validate the customer’s payment information before processing the order.  
**REQ-3**: The system must generate a unique order ID for each placed order and store all relevant details (product information, quantities, payment details).  
**REQ-4**: The system must allow the Stock Manager or Adminto update the order status (e.g., “Processing,” “Shipped,” “Delivered,” “Canceled”).  
**REQ-5**: The system must provide customers with real-time updates on the status of their order, including tracking information once it is shipped.  
**REQ-6**: The system must allow customers to cancel orders within a predefined time frame after placement.  
**REQ-7**: The system must allow the Stock Manager to view and manage all orders in the order management dashboard.  
**REQ-8**: The system must notify the Stock Manager when an order is successfully placed, shipped, or canceled.  
**REQ-9**: The system must handle errors gracefully, providing clear error messages in case of payment failure or other issues.

## Sales Reporting

The Sales Reporting feature enables the generation of detailed reports on sales transactions, providing valuable insights into sales trends, inventory performance, and customer purchasing behavior. It allows the Admin to generate reports based on customizable timeframes and product categories, helping inform business decisions. This feature is of high priority, as it directly supports strategic planning and performance analysis.

**REQ-1**: The system must allow the Adminto generate sales reports for specific time periods (e.g., daily, weekly, monthly).  
**REQ-2**: The system must allow filtering of sales reports by product, category, or customer.  
**REQ-3**: The system must provide visual representations of sales data (e.g., graphs, charts) for easy analysis.  
**REQ-4**: The system must allow reports to be exported in multiple formats (e.g., PDF, CSV, Excel).  
**REQ-5**: The system must calculate and display key metrics in sales reports, such as total revenue, average order value, and sales by product/category.  
**REQ-6**: The system must update the report data in real time based on the most recent transactions.  
**REQ-7**: The system must ensure the accuracy of the generated reports, with data validated against transaction records.  
**REQ-8**: The system must allow scheduled reports to be automatically generated and sent to designated recipients.  
**REQ-9**: The system must provide an option for the Admin to view past reports for historical analysis.

## Role-based Access Control

This feature ensures that users are granted access to the system based on their defined roles, limiting their ability to interact with sensitive data and system functionalities. It allows the Admin to define roles such as Admin**,** Stock Manager**,** Supplier**,** and Customer**,** each with specific permissions and access levels. This feature is of high priority as it ensures system security and compliance by controlling who can access and modify data.

**REQ-1**: The system must allow the Admin to create, modify, and delete user roles.  
**REQ-2**: The system must allow the Admin to assign specific permissions to each role (e.g., view, edit, delete).  
**REQ-3**: The system must restrict access to sensitive data and actions based on the user's assigned role.  
**REQ-4**: The system must prevent unauthorized users from accessing restricted features (e.g., inventory management, user management).  
**REQ-5**: The system must support different access levels, such as Admin**,** Stock Manager**,** Supplier**,** and Customer**,** each with predefined permissions.  
**REQ-6**: The system must log all access and actions performed by users, with the ability to audit these logs.  
**REQ-7**: The system must allow the Admin to modify the access level of any user at any time.  
**REQ-8**: The system must provide a secure authentication process (e.g., username and password, two-factor authentication) for all users.  
**REQ-9**: The system must allow role-based access to reports and system settings, ensuring only authorized users can view or modify them.

## Integration with External Systems

The Integration with External Systems feature enables the StockWise (SW) to communicate and exchange data with third-party systems such as Point of Sale (POS) systems, Enterprise Resource Planning (ERP) systems, and payment gateways. This functionality ensures that SW can seamlessly interact with other business tools to streamline operations, improve data accuracy, and enhance overall efficiency. This feature is of high priority, as it directly impacts the system’s ability to work within the larger business ecosystem and provide real-time data synchronization.

**REQ-1**: The system must be able to integrate with external POS systems for real-time sales data synchronization.  
**REQ-2**: The system must support integration with ERP systems to sync inventory, order, and supplier data.  
**REQ-3**: The system must allow secure communication with external systems through APIs, ensuring data privacy and integrity.  
**REQ-4**: The system must support integration with payment gateways to process customer payments.  
**REQ-5**: The system must allow the Adminto configure and manage integration settings for external systems.  
**REQ-6**: The system must log all data exchanges with external systems for auditing and troubleshooting purposes.  
**REQ-7**: The system must validate and handle any errors or discrepancies in data received from external systems, notifying the relevant users.  
**REQ-8**: The system must ensure that data transferred between SW and external systems is accurate and consistent, with regular synchronization intervals.  
**REQ-9**: The system must provide an option to disable or modify external system integrations if necessary.

# Other Nonfunctional Requirements

## Performance Requirements

The StockWise (SW) must meet several performance requirements to ensure optimal operation under varying conditions. The system should respond to user actions, such as searching for products or placing orders, within 2 seconds for 90% of requests, ensuring a seamless experience. SW must support up to 10,000 concurrent users without significant performance degradation, with scalability provisions for high traffic periods. During peak operations, the system must process at least 1,000 database transactions per second to handle inventory updates and orders efficiently. Order processing, including payment validation and inventory checks, should be completed within 5 seconds to maintain a smooth checkout process. Sales and inventory reports should be generated in under 30 seconds for data spanning up to a month, with longer reports (e.g., for a year) completed within 2 minutes. Additionally, the system must ensure 99.9% uptime, excluding planned maintenance, to guarantee reliability and continuous service. These requirements are essential to providing a responsive, reliable, and scalable system that meets the needs of users and business operations.

## Safety Requirements

The StockWise (SW) must adhere to safety standards to prevent data loss, unauthorized access, and other potential risks that could harm users or the business. The system must implement robust data backup procedures to ensure no loss of critical inventory or order information in case of system failure. Regular, automated backups should be conducted, with backup data stored securely and tested periodically for integrity. Additionally, the system must enforce strict access controls to prevent unauthorized users from gaining access to sensitive data, including customer payment information and inventory records. Actions such as unauthorized data modification, deletion of orders, or access to restricted areas must be prevented by role-based access control and two-factor authentication. The system must comply with relevant data protection and privacy regulations, including the GDPR (General Data Protection Regulation) for handling user data, and must implement encryption for sensitive data both at rest and in transit. No safety certifications are required for this system, but compliance with industry standards for software security and data privacy must be maintained to ensure the safe use of SW.

## Security Requirements

The StockWise (SW) must implement strong security measures to protect user data, prevent unauthorized access, and comply with relevant privacy regulations. All users must authenticate their identity through secure login mechanisms, including username and passwordwith a minimum of 8 characters, and two-factor authentication (2FA) for Admins and Stock Managers to access sensitive data. The system must enforce role-based access control (RBAC)**,** ensuring users only have access to functionalities and data relevant to their roles. All sensitive data, including customer payment details, product information, and inventory levels, must be encrypted both at rest and in transit using industry-standard encryption protocols (e.g., AES-256 for data storage, TLS for data transmission). The system must be compliant with GDPR (General Data Protection Regulation) and CCPA (California Consumer Privacy Act) to ensure proper handling of personal data, including provisions for data access, deletion, and export. Additionally, audit logs must be maintained to track user activity, with specific attention to actions involving sensitive data changes. The system should also adhere to the OWASP Top Ten security guidelines to minimize risks such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). Regular security testing, including vulnerability assessments and penetration testing, should be conducted to identify and mitigate potential threats.

## Software Quality Attributes

The StockWise (SW) must meet several key software quality attributes to ensure optimal performance and user satisfaction. Usability is a top priority, with a user-friendly interface designed for easy navigation and minimal training, aiming for a user satisfaction score of 85% or higher. Reliability is also critical, with the system required to maintain 99.9% uptime to ensure continuous availability for business operations. The system should be highly maintainable, with a modular architecture that supports easy updates and extensions, targeting a mean time to repair (MTTR) of no more than 4 hours for critical issues. Interoperability is essential, as the system must integrate seamlessly with external tools like POS, ERP, and payment gateways using industry-standard protocols. These attributes together ensure that SW delivers a reliable, efficient, and user-friendly solution for inventory management.

## Business Rules

The StockWise (SW) operates under several key business rules to ensure smooth and secure operations. Only Admins are authorized to create, modify, or delete user accounts and assign roles, while Stock Managers can update inventory levels and manage product details but cannot change user roles or access sensitive settings. Suppliers can view and fulfill purchase orders but have no access to other inventory data or customer information. Customers are allowed to cancel orders only within 24 hours of placement, provided the order hasn't been shipped. Additionally, Stock Managers must generate purchase orders when stock levels fall below the predefined threshold, but only after receiving a low-stock alert. Orders cannot proceed without successful payment verification, and low-stock alerts are triggered only for products with active thresholds set. These rules help maintain security, ensure smooth workflow, and align user actions with system responsibilities.

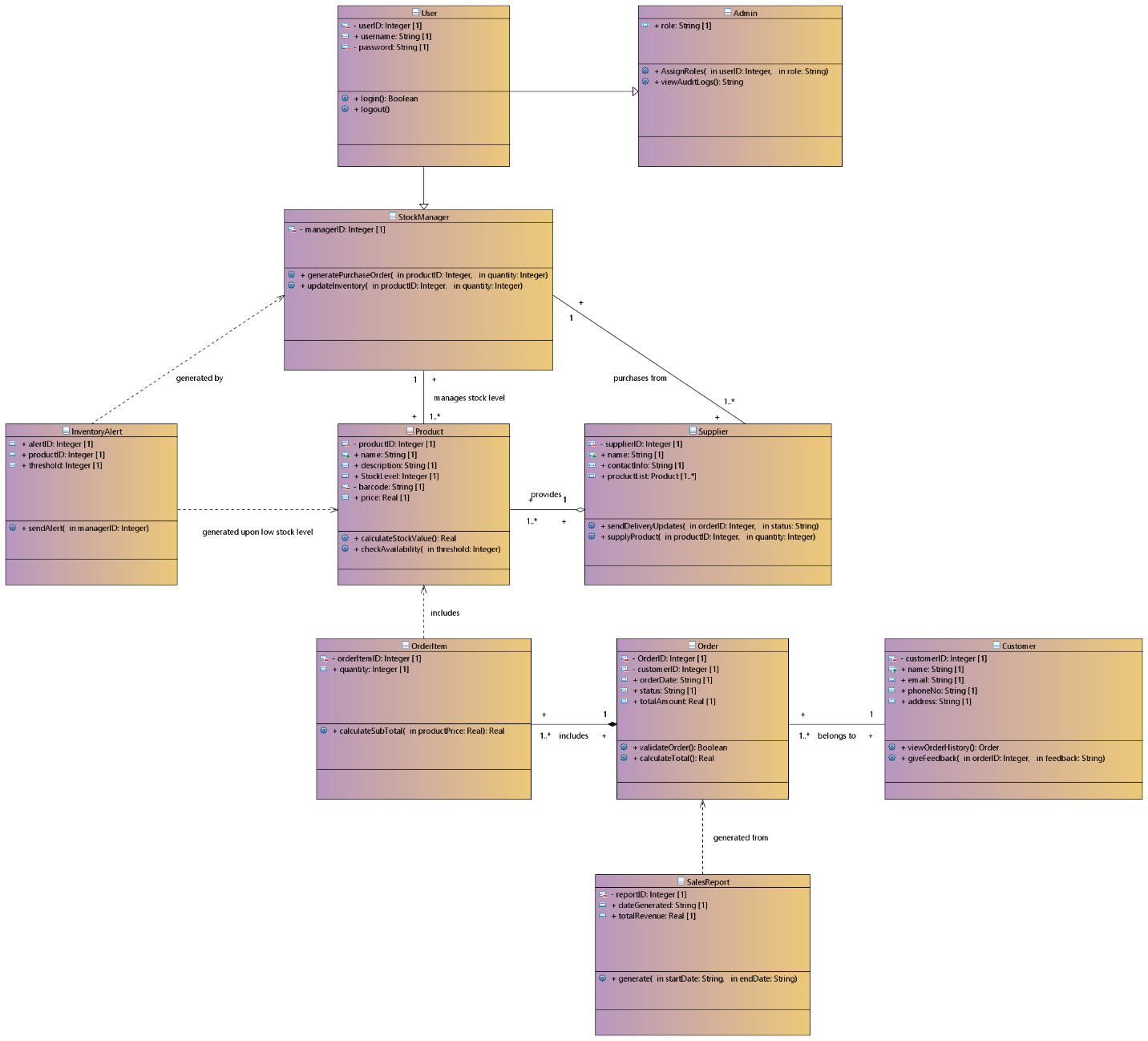
# Other Requirements

In addition to the requirements outlined in the previous sections, the StockWise (SW) must meet several other key requirements. It will use PostgreSQL as the primary database, ensuring scalability and high availability for storing product, order, and inventory data, with regular backups and a disaster recovery plan in place. The system must support internationalization, allowing users to interact with the system in different languages and manage transactions in local currencies, with adaptive date and time formats. SW must also comply with legal regulations such as GDPR and CCPA to protect customer data and ensure privacy, while supporting electronic invoicing and audit trails for financial transactions. To improve development efficiency, the system should leverage reusable software components or third-party libraries, particularly for common features like authentication and reporting. Additionally, the system must include automated backup processes and a disaster recovery plan to safeguard against data loss and ensure business continuity. These requirements ensure that SW is secure, adaptable to different regions, and compliant with necessary legal standards.

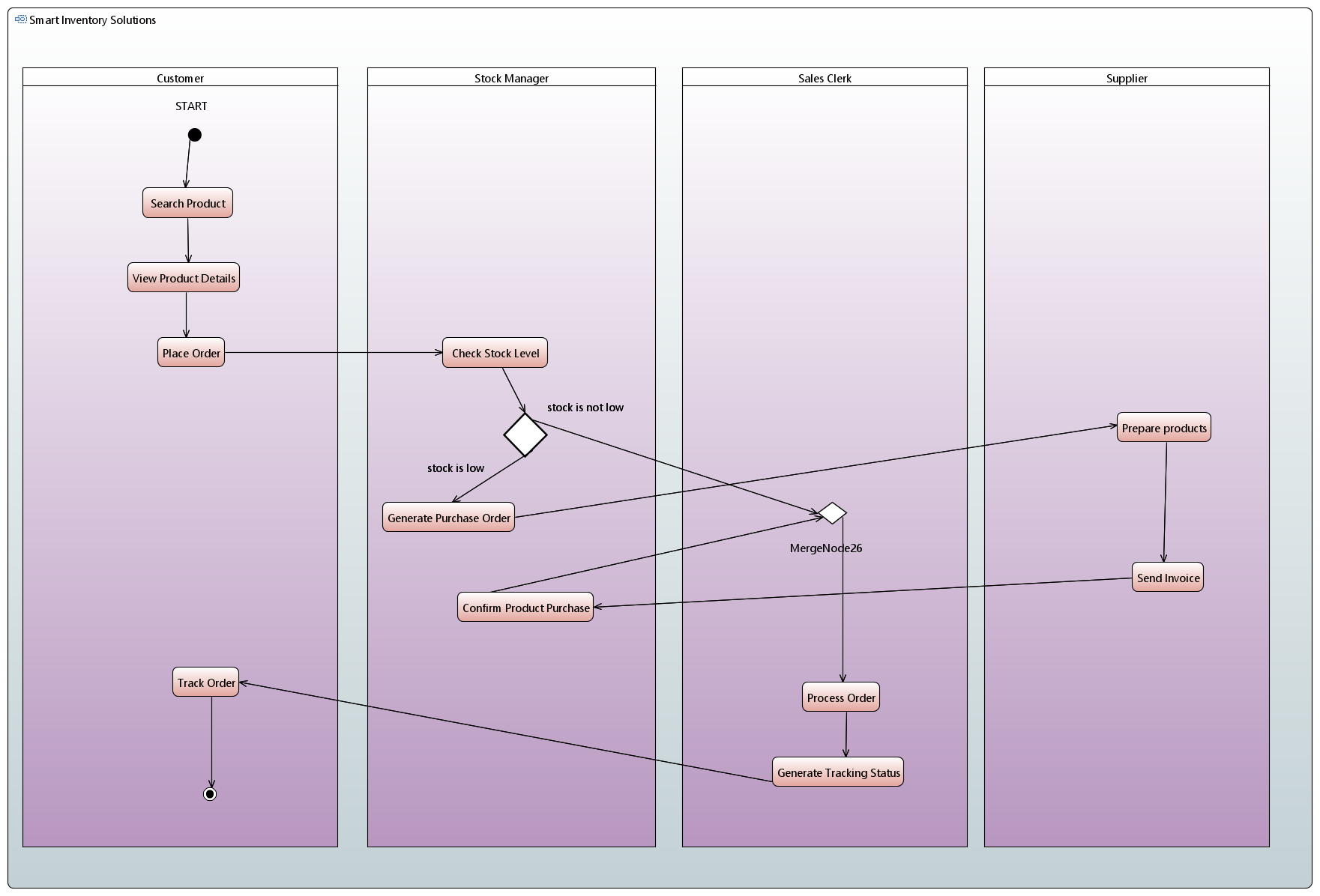
# Appendix: Analysis Models

# 

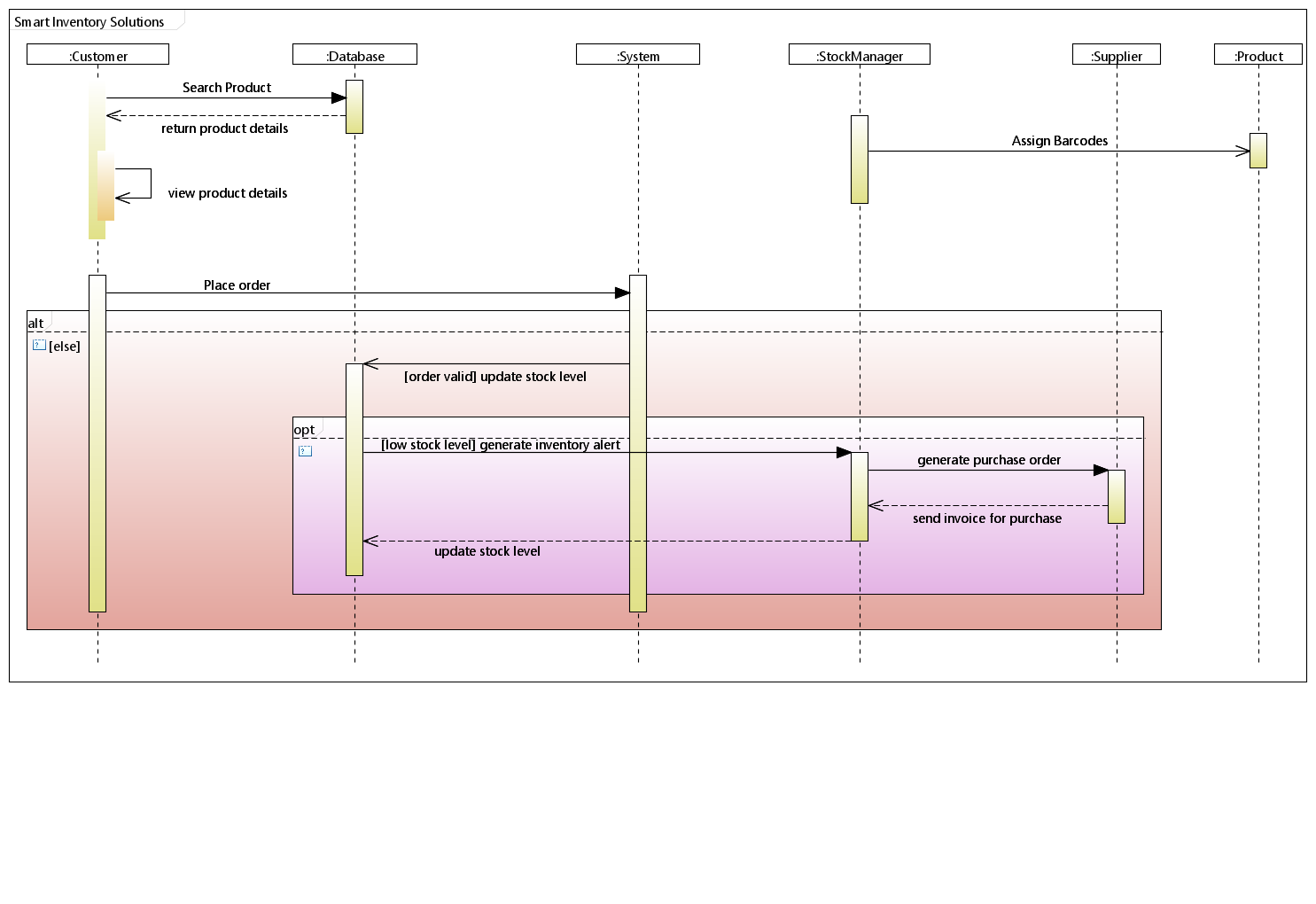
**Diagram 1: Use-Case Diagram**

****

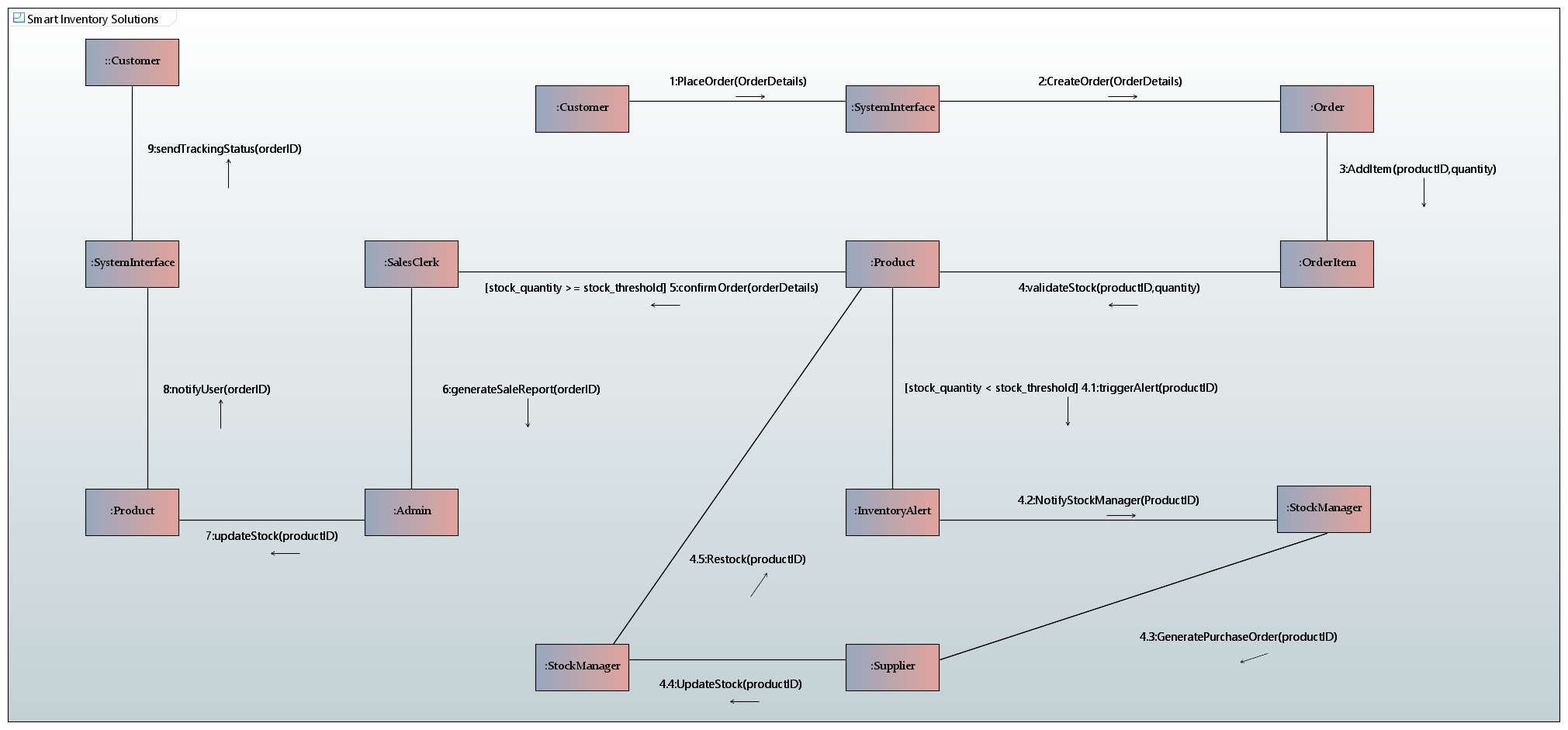
**Diagram 2: Class Diagram**

****

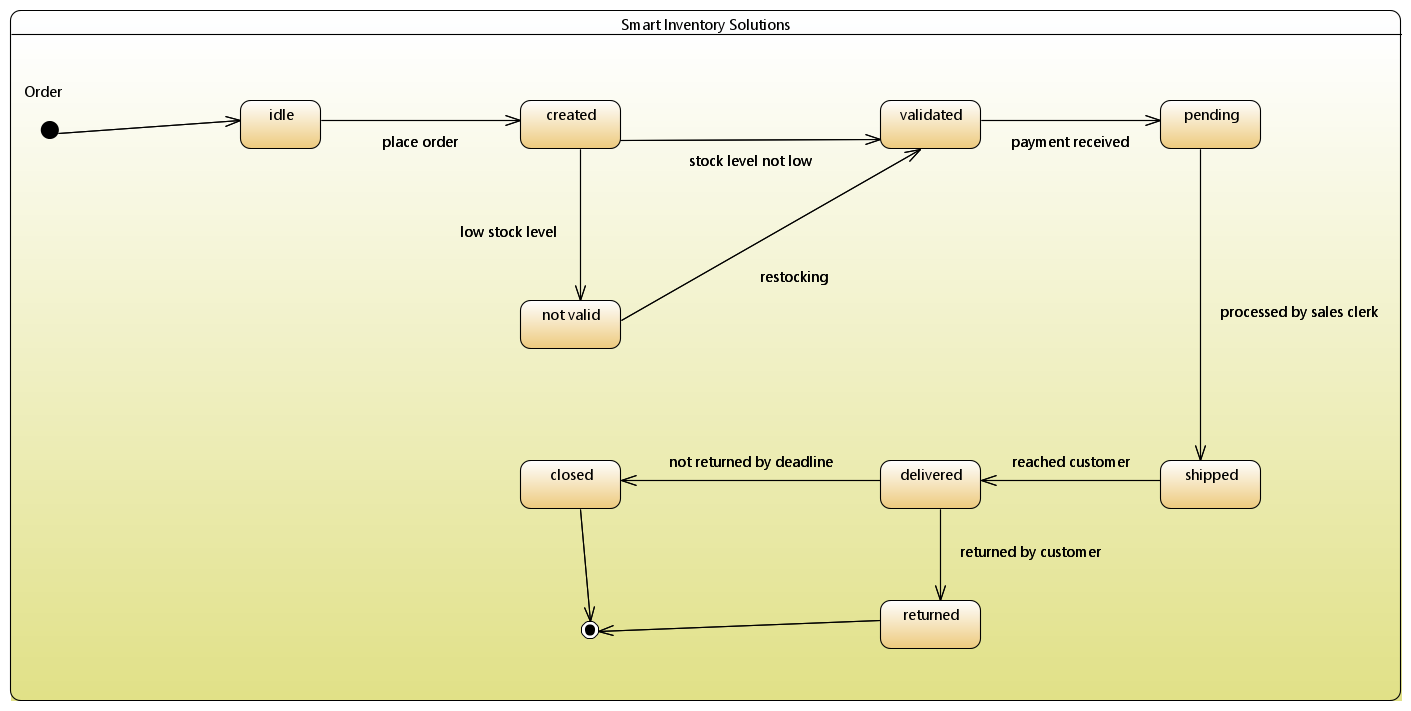
**Diagram 3: Activity Diagram**

****

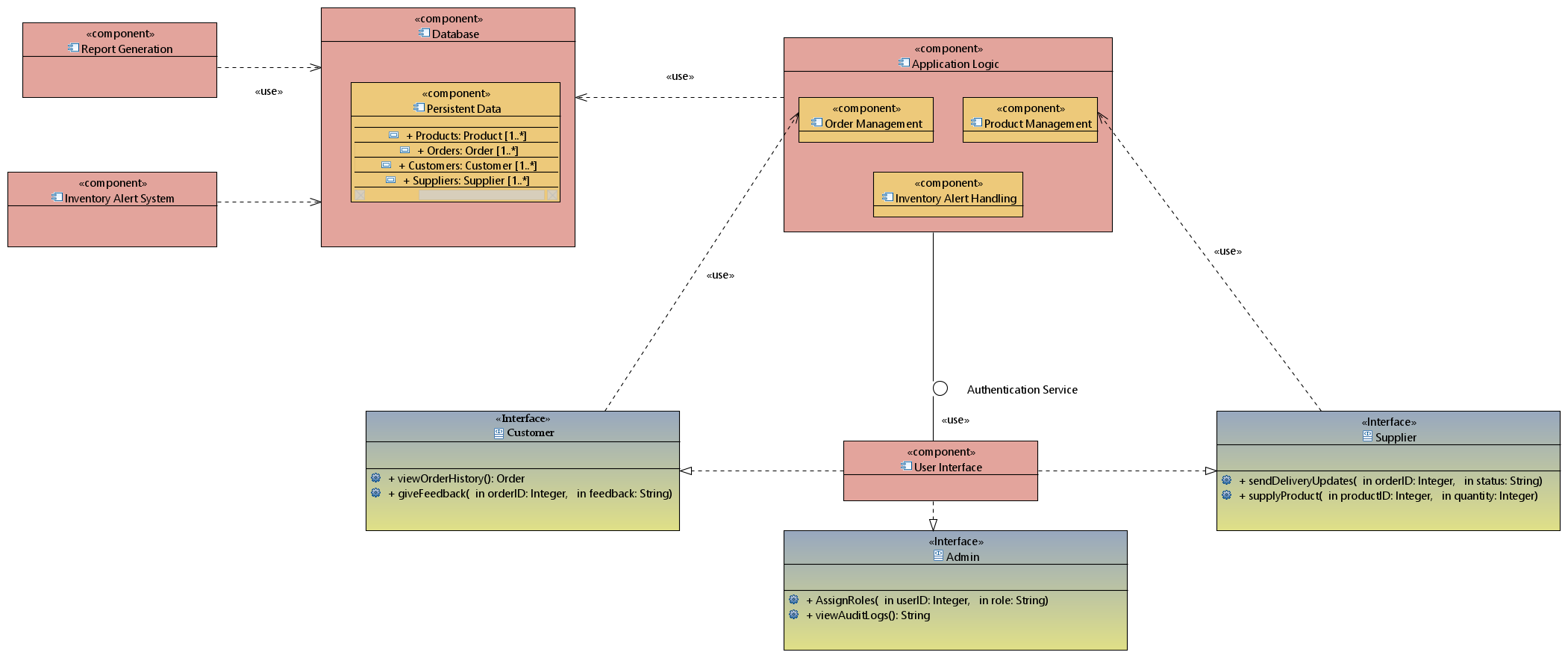
**Diagram 4: Sequence Diagram**

****

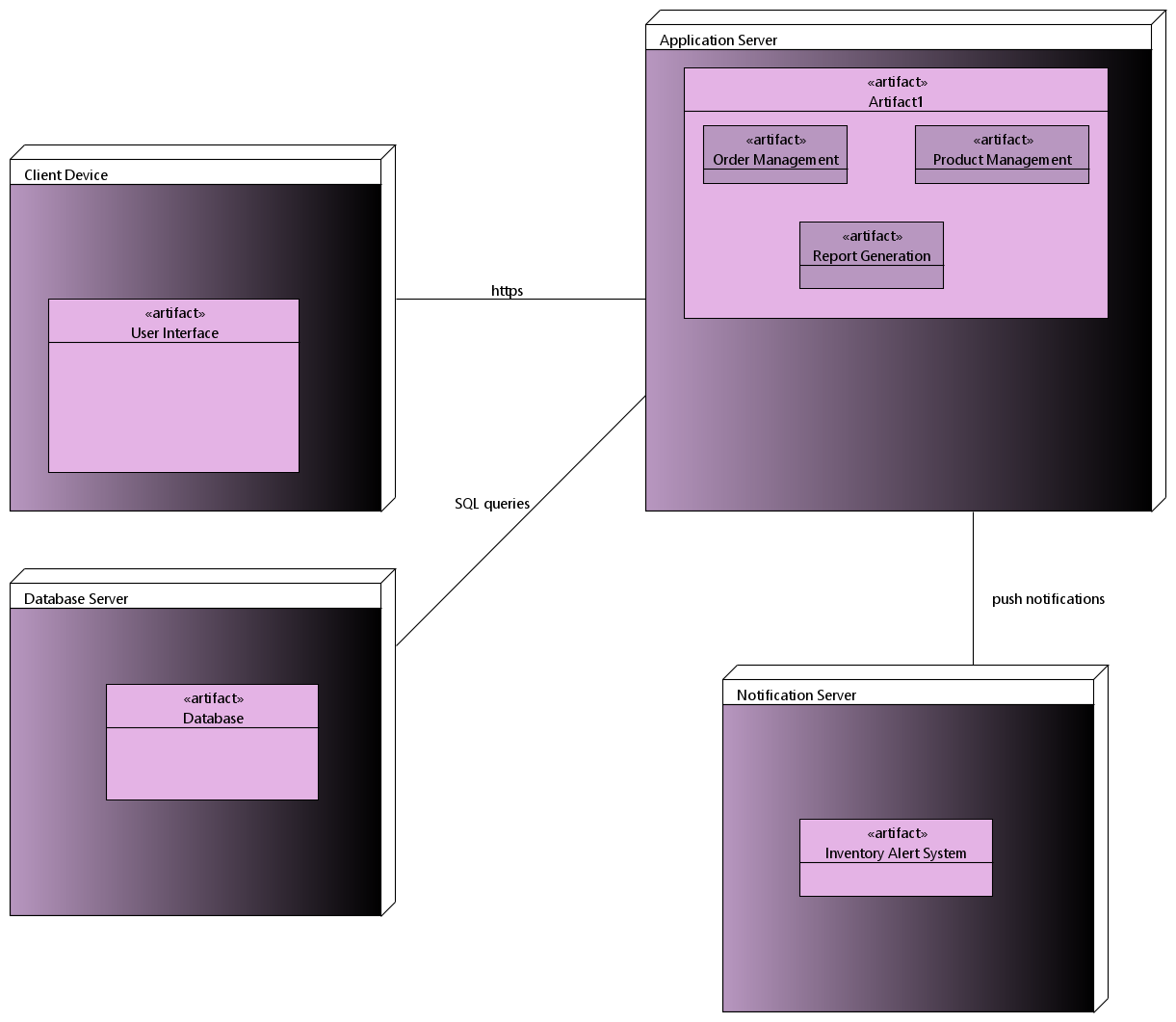
**Diagram 5: Communication Diagram**

****

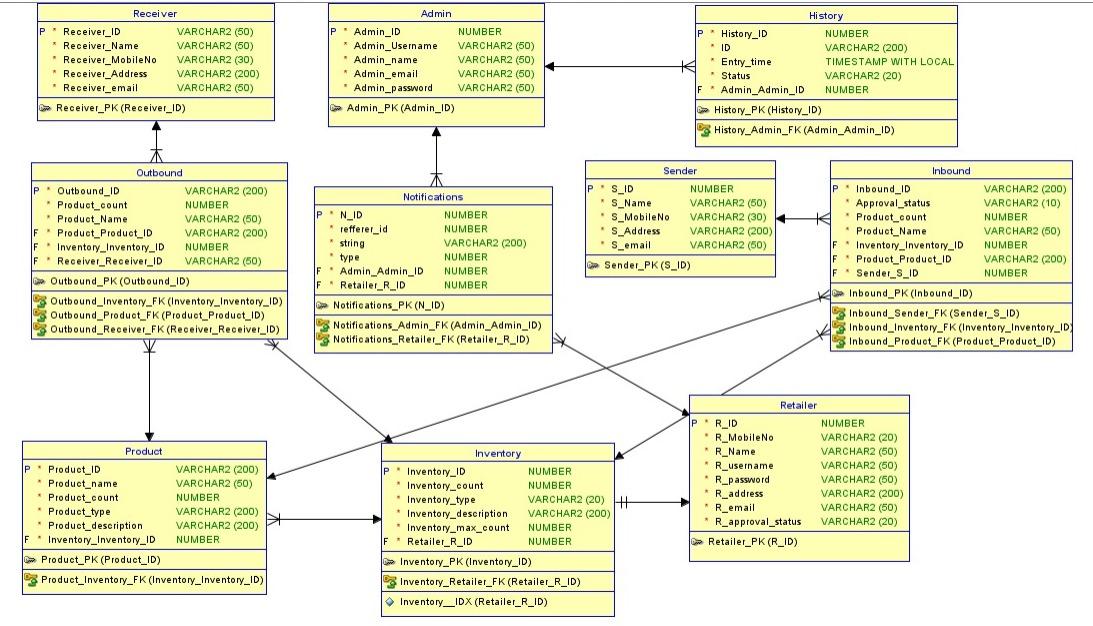
**Diagram 6: State Machine Diagram**

****

**Diagram 7: Component Diagram**

****

**Diagram 8: Deployment Diagram**

****

**Diagram 9: Entity Relationship Diagram**