1. Introduction

This document outlines the Low-Level Design (LLD) for an **Inventory Management System** that manages the stock of products, tracks inventory levels, handles orders and shipments, and provides reports. The system aims to ensure seamless inventory tracking, order processing, and warehouse management.

This design supports both **Java (Spring Boot)** and **.NET (ASP.NET Core)** frameworks for backend development.

2. Module Overview

2.1 Product Management Module

- Manages product details such as name, description, price, and stock levels.
- Allows admins to add, update, and delete product entries.

2.2 Order Management Module

- Handles the creation, tracking, and processing of customer orders.
- Includes functionality for managing order statuses and shipments.

2.3 Stock Management Module

- Tracks product inventory levels and handles stock updates based on orders and shipments.
- Provides alerts when inventory levels reach low thresholds.

2.4 Supplier Management Module

- Manages suppliers and their contact information.
- Tracks supplier relationships and stock procurement.

2.5 Reporting and Analytics Module

- Provides reporting capabilities to view inventory trends, order histories, and product performance.
- Generates data-driven insights for inventory optimization.

3. Architecture Overview

3.1 Architectural Style

- Frontend: Angular or React
- Backend: REST API-based architecture
- Database: Relational Database (MySQL/PostgreSQL/SQL Server)

3.2 Component Interaction

- The frontend communicates with the backend via REST APIs for product management, order processing, and inventory updates.
- The backend processes requests, interacts with the database to store and retrieve data, and provides responses to the frontend.

4. Module-Wise Design

4.1 Product Management Module

Features:

- Admins can add, update, and delete products.
- Users can view available products and their details.

Data Flow:

- Admins use the frontend to initiate product-related operations.
- The backend processes these requests and updates the product database.

Entities:

- **Product**:
  - ProductID
  - Name
  - Description
  - Price
  - StockLevel

4.2 Order Management Module

Features:

- Create and manage customer orders.
- Track order status and shipment progress.

Data Flow:

- Users place orders via the frontend.
- The backend processes the order, updates inventory levels, and handles payment and shipment tracking.

Entities:

- **Order**:
  - OrderID
  - CustomerID
  - ProductID
  - Quantity
  - OrderDate
  - Status (Pending, Shipped, Delivered)

4.3 Stock Management Module

Features:

- Tracks product stock levels and updates based on orders and deliveries.
- Notifies admins when stock levels are low.

Data Flow:

- Stock levels are updated automatically when an order is placed or a shipment is received.
- Admins can view current stock levels and inventory trends.

Entities:

- **Stock**:
  - ProductID
  - Quantity
  - ReorderLevel

4.4 Supplier Management Module

Features:

- Manage supplier profiles and contact information.
- Track orders and shipments from suppliers.

Data Flow:

- Admins enter and update supplier data via the frontend.
- Backend processes supplier interactions and updates supplier information in the database.

Entities:

- **Supplier**:
  - SupplierID
  - Name
  - ContactInfo
  - ProductsSupplied

4.5 Reporting and Analytics Module

Features:

- Generate reports on inventory levels, sales performance, and order history.
- Provide data-driven insights on stock trends, low-stock products, and supplier performance.

Data Flow:

- Admins generate reports through the frontend.
- The backend aggregates data and provides comprehensive reports on various parameters.

Entities:

- **Report**:
  - ReportID
  - ReportType (Inventory, Order, Supplier)
  - StartDate
  - EndDate
  - Data

5. Deployment Strategy

5.1 Local Deployment

- Frontend: Use ng serve for Angular or equivalent local servers for React.
- Backend: Deploy the REST API locally using Spring Boot or ASP.NET Core.

- Database: Set up a local instance of MySQL/PostgreSQL/SQL Server for development.

6. Database Design

6.1 Tables and Relationships

1. **Product**
    - Primary Key: ProductID
    - Foreign Key: None
2. **Order**
    - Primary Key: OrderID
    - Foreign Key: CustomerID, ProductID
3. **Stock**
    - Primary Key: ProductID
    - Foreign Key: ProductID
4. **Supplier**
    - Primary Key: SupplierID
    - Foreign Key: None
5. **Report**
    - Primary Key: ReportID
    - Foreign Key: None

7. User Interface Design

7.1 Wireframes:

- Product Management Dashboard
- Order Management Console
- Stock Overview Page
- Supplier Management Interface
- Reporting Dashboard

8. Non-Functional Requirements

8.1 Performance

- The system should be capable of handling up to 200 concurrent users.

8.2 Scalability

- The system must be scalable to accommodate an increase in product inventory, orders, and suppliers as the business grows.

8.3 Security

- User authentication and authorization must be implemented for different user roles (admin, manager, staff).
- All sensitive data must be encrypted both in transit and at rest.

8.4 Usability

- The user interface should be intuitive, with easy navigation and responsive design.