WAREHOUSE MANAGEMENT SYSTEM

1. Introduction

1.1 Purpose

The purpose of this document is to define the requirements for a **Warehouse Management System**. This system will help warehouse managers and staff efficiently manage inventory, suppliers, stock requests, and product history to streamline warehouse operations.

1.2 Scope

The Warehouse Management System (WMS) is a Java-based software with JDBC integration designed to assist warehouse personnel in managing products, suppliers, and stock levels. The system will support functionalities like adding and viewing products, managing suppliers, tracking stock requests, and updating product histories. It will also maintain accurate records of stock updates and historical changes.

1.3 Definitions, Acronyms, and Abbreviations

- **WMS**: Warehouse Management System
- **ERD**: Entity Relationship Diagram
- **JDBC**: Java Database Connectivity
- **UI**: User Interface

1.4 References

- Database design for relational models.
- Java JDBC Documentation for MySQL integration.
- UML diagrams for visual representation of system functionality.

1.5 Overview

This document provides an overview of the functional and non-functional requirements for the Warehouse Management System and serves as a blueprint for design and development.

---

2. Overall Description

2.1 Product Perspective

The WMS is a standalone Java-based application that integrates with a MySQL database via JDBC. The application will offer a console-based interface for warehouse personnel to perform inventory management tasks.

2.2 Product Functions

- **Product Management**: Add, view, and update products.
- **Supplier Management**: Add and view suppliers.
- **Stock Requests**: Create and handle stock requests from suppliers.
- **Inventory Updates**: Track and update stock levels.
- **Product History Tracking**: Record changes in product quantities over time.

2.3 User Characteristics

- **Warehouse Manager**: Primary user responsible for inventory decisions, stock levels, and supplier management.
- **Warehouse Staff**: Assists in product management and stock updates.

2.4 Constraints

- **Database Dependency**: Requires a MySQL database with tables for products, suppliers, stock requests, and product history.
- **Data Consistency**: Database transactions must ensure data integrity for all inventory operations.
- **Platform Dependency**: Runs on systems with Java and MySQL installed.

2.5 Assumptions and Dependencies

- Users are assumed to have basic knowledge of Java-based applications.
- JDBC must be properly configured to connect to the MySQL database.

---

3. Functional Requirements

3.1 Product Management

Description: The system will allow adding new products, viewing existing products, and updating stock levels for each product.

3.1.1 Add Product

- **Input**: Product details (ID, name, type, quantity, price, storage requirements).

- **Processing**: The system saves the product details to the Products table in the database.
- **Output**: Confirmation of product addition.

3.1.2 View Products

- **Input**: None.
- **Processing**: Retrieves and displays the list of products from the Products table.
- **Output**: Displays product details (ID, name, type, quantity, price, storage requirements).

3.1.3 Update Stock

- **Input**: Product ID and new quantity.
- **Processing**: Updates the quantity of the specified product in the database.
- **Output**: Confirmation of stock update.

3.2 Supplier Management

Description: The system will allow adding new suppliers and viewing the list of suppliers.

3.2.1 Add Supplier

- **Input**: Supplier details (ID, name, contact info).
- **Processing**: The system saves the supplier details to the Suppliers table in the database.
- **Output**: Confirmation of supplier addition.

3.2.2 View Suppliers

- **Input**: None.
- **Processing**: Retrieves and displays the list of suppliers from the Suppliers table.
- **Output**: Displays supplier details (ID, name, contact info).

3.3 Stock Requests

Description: The system will allow creating stock requests from suppliers, handling them, and updating stock levels.

3.3.1 Create Stock Request

- **Input**: Stock request details (product ID, supplier ID, quantity, request date, status).

- **Processing**: Saves the stock request to the StockRequests table.
- **Output**: Confirmation of stock request creation.

3.3.2 Handle Stock Request

- **Input**: Stock request details (e.g., status, quantity).
- **Processing**: Updates stock levels based on stock requests and records changes in ProductHistory.
- **Output**: Confirmation of stock update and stock request handling.

3.4 Product History Tracking

Description: The system will log all changes in product quantities and maintain a history of changes.

3.4.1 Record Product History

- **Input**: Product history details (product ID, quantity changed, change date, change type, status).
- **Processing**: Saves the product history entry in the ProductHistory table.
- **Output**: Confirmation of history entry.

---

4. Non-Functional Requirements

4.1 Performance

- The system should be able to process inventory updates and queries within 2 seconds.
- Database queries should be optimized for quick retrieval.

4.2 Reliability

- Data integrity should be maintained throughout CRUD operations.
- The system should handle concurrent access and ensure database consistency.

4.3 Usability

- User-friendly console interface for ease of interaction.
- Clear prompts and messages for users to guide them through actions.

4.4 Security

- Sensitive data (e.g., supplier contact info) should be stored securely.

- Prevent unauthorized access to database configurations.

4.5 Portability

- The system should be portable across machines running Java and MySQL.
- It should support multiple platforms (Windows, Linux, MacOS).

---

5. System Models

5.1 ER Diagram

The ER diagram represents entities like Product, Supplier, StockRequest, and ProductHistory, along with their relationships and attributes.

5.2 Use Case Diagram

The use case diagram includes:

- **Actors**: Warehouse Manager, Warehouse Staff.
- **Use Cases**: Add/View Product, Add/View Supplier, Update Stock, Handle Stock Request.

---

6. Appendix

6.1 Assumptions

- All users have basic knowledge of console-based applications.
- Network connectivity is reliable for database transactions.

6.2 Database Schema (Optional)

- Tables for Products, Suppliers, StockRequests, ProductHistory, along with primary and foreign keys for relational mapping.

6.3 Acronyms

- **CRUD**: Create, Read, Update, Delete.
- **DB**: Database.