Software Requirements Specification (SRS) Document

Project Name: Inventory Management System (IMS)

Version: 1.0

Author: Mohamed Hany Mosa

Date: [1/4/2025]

---

1. Introduction

1.1 Purpose

The purpose of the Inventory Management System (IMS) is to help businesses efficiently track and manage their inventory, sales, purchases, suppliers, and customers. The system will provide role-based access to employees, ensuring secure and efficient inventory operations.

1.2 Scope

IMS is a desktop app built using C# (.NET Framework) and SQL Server.

The system will allow users to:

Manage inventory (add, update, delete, and track products).

Record sales and purchases with transaction details.

Generate reports on stock levels, sales trends, and purchases.

Manage employees, users, customers and suppliers efficiently.

Ensure security with authentication and role-based authorization.

The system does not include features like online ordering, barcode scanning, or multi-branch inventory tracking in the initial version but can be expanded later.

1.3 Intended Users

Admin: Full control over the system (manages users, inventory, and reports).

Inventory Manager: Manages stock levels, purchases, and suppliers.

Cashier: Records sales transactions and manages customers.

1.4 Assumptions & Constraints

The system will run on Windows OS.

Database will be SQL Server.

Users must have basic computer skills.

The system will work in a single-location business (multi-location support can be added later).

---

2. Overall Description

2.1 System Perspective

The IMS follows a 3-tier architecture:

Presentation Layer (UI) – Windows Forms for user interaction.

Business Logic Layer – Manages validation, processing, and calculations.

Data Access Layer – Handles database interactions using ADO.NET.

2.2 System Functions

User Authentication & Authorization (Login, Role-based Access).

Product Management (CRUD operations on products and product batches).

Sales Management (Process sales, generate invoices).

Purchase Management (Record supplier transactions).

Customer & Supplier Management (Store contact and transaction history).

Reporting (Generate stock reports, sales reports).

2.3 User Characteristics

Admin: Advanced user with full control.

Inventory Manager: Moderate technical knowledge to manage products and stock.

Cashier: Basic computer skills to handle sales.

2.4 Dependencies

.NET Framework for application development.

SQL Server for database storage.

Git for version control.

---

3. Functional Requirements

3.1 User Management

Users must log in with a username and password.

Role-based access:

Admin can create/update/delete users.

Inventory Manager can manage products, suppliers, and purchases.

Cashier can handle sales and customers.

3.2 Product Management

Users can add, update, delete, and search for products.

Each product must have name, category, price, quantity, supplier, etc.

The system should alert when stock is low.

3.3 Sales Management

Users can create a new sale, select products, and generate invoices.

The system should update stock levels automatically after a sale.

Sales details should be stored with timestamps for reporting.

3.4 Purchase Management

Users can record purchases from suppliers.

Stock levels should increase automatically when a purchase is recorded.

3.5 Customer & Supplier Management

Users can store customer details (name, phone, address, purchase history).

Users can store supplier details (company name, contact info, past transactions).

3.6 Reporting & Analytics

Generate reports for:

Sales (Daily, Weekly, Monthly)

Stock Levels (Low stock, Expired products)

Purchase History

3.7 Security & Access Control

Passwords must be hashed before storage.

Users can reset their password securely.

Only authorized users can access specific modules(forms).

---

4. Non-Functional Requirements

4.1 Performance

The system should be able to handle 1,000+ products efficiently.

Sales processing should not exceed 2 seconds per transaction.

4.2 Security

Use role-based access to restrict features.

Implement data encryption for sensitive data (passwords).

4.3 Usability

The system should have a simple and intuitive UI.

Tooltips and error messages should guide users.

4.4 Scalability

The system should allow future upgrades, such as:

Multi-branch inventory management.

Integration with barcode scanners.

4.5 Backup & Recovery

The system should support database backups for data protection.