**Software Requirements Specification (SRS)**

**Project Title:** IBMS (Inventory and Billing Management System)

**Version:** 1.0

**Date:** July 20 2025

**Prepared By:** Jayakrishnan J

**Table of Contents**

1. Introduction  
   1.1. Purpose  
   1.2. Document Conventions  
   1.3. Intended Audience and Reading Suggestions  
   1.4. Project Scope  
   1.5. Definitions, Acronyms, and Abbreviations
2. Overall Description  
   2.1. Product Perspective  
   2.2. Product Functions  
   2.3. User Characteristics  
   2.4. Constraints  
   2.5. Assumptions and Dependencies
3. Specific Requirements  
   3.1. Functional Requirements  
   3.1.1. [Feature 1 Name]  
   3.1.2. [Feature 2 Name]  
   ...  
   3.2. Non-Functional Requirements  
   3.2.1. Performance Requirements  
   3.2.2. Security Requirements  
   3.2.3. Usability Requirements  
   3.3. External Interface Requirements  
   3.3.1. User Interfaces  
   3.3.2. Hardware Interfaces  
   3.3.3. Software Interfaces  
   3.3.4. Communications Interfaces  
   3.4. Data Model/Database Requirements (if applicable)
4. Problem Identification & Analysis  
   4.1. Problem Statement  
   4.2. Analysis of Existing Solutions/Literature Review  
   4.3. Proposed Approach and Innovation  
   4.4. Research Questions (if applicable)
5. System Architecture (High-Level Design)
6. Future Enhancements/Research Directions
7. References
8. Appendix (if any)

# 1. Introduction

## 1.1. Purpose

The purpose of this SRS document is to define the requirements for the Inventory and Billing Management System (IBMS) software system. This document serves as a comprehensive specification that outlines the functional and non-functional requirements, system architecture, and implementation guidelines for developing a web-based inventory and billing management solution for small to medium-sized businesses.

## 1.2. Document Conventions

This document follows the following conventions:

• Functional requirements are identified with "FR-XXX" format

• Non-functional requirements are identified with "NFR-XXX" format

• Headings use hierarchical numbering (1, 1.1, 1.1.1)

• Technical terms are defined in Section 1.5

## 1.3. Intended Audience and Reading Suggestions

This document is intended for the following audiences:

• Faculty: Review entire document with emphasis on Section 4 (Problem Analysis) for critical thinking assessment

## 1.4. Project Scope

The IBMS system scope encompasses the following capabilities:

• Inventory Management: Complete lifecycle management of product inventory including adding, editing, deleting, and tracking stock levels

• Customer Management: Comprehensive customer profile management with contact information and purchase history tracking

• Billing System: Automated bill generation with itemized details, transaction processing, and invoice management

• Automated Notifications: Real-time email and SMS alerts for low stock situations and critical business events

• Reporting: Basic reporting capabilities for inventory status, sales analysis, and business insights

• User Authentication: Secure login and user management system

• PDF Generation: Professional invoice generation in PDF format

The following features are explicitly out of scope:

• Barcode scanning and AI image recognition integration

## 1.5. Definitions, Acronyms, and Abbreviations

• IBMS: Inventory and Billing Management System - the complete software system described in this document

• API: Application Programming Interface - a set of rules and protocols for building software applications

• SMS: Short Message Service - text messaging service for mobile devices

• REST: Representational State Transfer - an architectural style for designing networked applications

• JWT: JSON Web Token - a compact, URL-safe means of representing claims between parties

• CRUD: Create, Read, Update, Delete - basic operations for persistent storage

• UI: User Interface - the visual and interactive elements of the software

• UX: User Experience - the overall experience of a person using the software

• DB: Database - organized collection of structured information stored electronically

• PDF: Portable Document Format - file format used for document exchange

# 2. Overall Description

## 2.1. Product Perspective

The IBMS is a standalone web application that operates independently without requiring integration with existing enterprise systems. The system is designed as a complete solution for small to medium-sized businesses that need to manage their inventory and billing operations efficiently.

The system architecture follows a modern client-server model with:

• Frontend: Next.js-based web application providing responsive user interface

• Backend: Django REST API providing business logic and data processing

• Database: PostgreSQL for reliable data storage and management

• External Services: Twilio integration for SMS notifications

## 2.2. Product Functions

The IBMS system provides the following major functions:

• Inventory Management: Complete lifecycle management of products including creation, modification, deletion, and real-time stock tracking with automatic updates during sales transactions

• Customer Management: Comprehensive customer database management including profile creation, contact information maintenance, and complete purchase history tracking with search and filtering capabilities

• Billing Operations: Automated bill generation with itemized product details, automatic total calculation, inventory deduction, and professional PDF invoice creation with customizable templates

• Stock Monitoring: Intelligent low stock detection with configurable thresholds, automatic alert generation, and status tracking to prevent stockouts and optimize inventory levels

• Notification System: Multi-channel alert system using email and SMS notifications for critical business events, with configurable notification preferences and delivery status tracking

• Data Reporting: Comprehensive reporting capabilities including inventory status reports, sales analysis, customer purchase history, and business performance metrics with export functionality

## 2.3. User Characteristics

The system is designed for the following user types:

• Primary Users - Small Business Owners/Managers:

- Technical Skills: Basic computer literacy, familiarity with web browsers

- Domain Knowledge: Understanding of inventory and billing concepts

- Usage Pattern: Daily system access for business operations and decision-making

• Secondary Users - Sales Staff/Inventory Clerks:

- Technical Skills: Basic computer literacy, ability to use web forms

- Domain Knowledge: Familiarity with product information and customer service

- Usage Pattern: Regular access for data entry and customer service

• Administrative Users - System Administrators:

- Technical Skills: Basic IT knowledge, understanding of web applications

- Domain Knowledge: Business process understanding

- Usage Pattern: Occasional access for system configuration and maintenance

## 2.4. Constraints

The following constraints affect the development and operation of the IBMS system:

Technical Constraints:

• Platform: Must be web-based application accessible through standard web browsers

• Browser Compatibility: Must support modern browsers (Chrome, Firefox, Safari, Edge)

• Database: Must use PostgreSQL for data storage and management

• Programming Languages: Must use Python (Django) for backend and JavaScript (Next.js) for frontend

• Deployment: Must be deployable on standard web hosting platforms

Operational Constraints:

• Internet Connectivity: Requires stable internet connection for SMS notifications and external API calls

• Single Location: Limited to single-location business operations

• User Capacity: Designed for small to medium-sized businesses with limited concurrent users

• Data Volume: Optimized for moderate data volumes typical of small businesses

Resource Constraints:

• Development Time: Limited development timeline requiring efficient implementation

• Budget: Cost-effective solution using open-source technologies

• Team Size: Small development team requiring manageable complexity

• Maintenance: Minimal ongoing maintenance requirements

## 2.5. Assumptions and Dependencies

The following assumptions are made during the requirements elicitation process:

• User Expertise: Users have basic computer literacy and can operate web browsers effectively

• Internet Connectivity: Business has reliable internet connectivity for system operation

• Twilio Service: Twilio account and credentials are available for SMS notification services

• Hardware: Standard computing devices with modern web browsers are available

• Business Process: Business follows standard inventory and billing processes

• Data Security: Basic security measures are sufficient for small business operations

The system has the following dependencies:

• Twilio API: External SMS notification service for alert delivery

• PostgreSQL Database: Reliable database system for data persistence

• Django Framework: Python web framework for backend development

• Next.js Framework: React-based framework for frontend development

• ReportLab Library: Python library for PDF generation

• Web Hosting: Platform for deploying the web application

# 3. Specific Requirements

## 3.1. Functional Requirements

This section details the specific functions the system must perform. Each requirement is uniquely identified and includes comprehensive specifications.

### 3.1.1. Inventory Management (FR-001)

Description: The system shall provide comprehensive inventory management capabilities allowing users to create, modify, delete, and track product information with real-time stock level monitoring.

Input: Product name , description , quantity , price , category

Processing: Validate input data, check for duplicate product names, store in database with timestamps, update inventory levels during transactions

Output: Confirmation message, updated inventory list, product details with current stock levels

Pre-conditions: User is authenticated and has appropriate permissions

Post-conditions: Product is successfully added/updated in database, inventory list is refreshed

Error Handling: Display validation errors for invalid input, prevent duplicate product names, handle database connection failures gracefully

### 3.1.2. Customer Management (FR-002)

Description: The system shall maintain comprehensive customer profiles including contact information, purchase history, and transaction records for effective customer relationship management.

Input: Customer name , email address , phone number , address

Processing: Validate email format, check for duplicate email addresses, store customer data with creation timestamp, link to purchase history

Output: Customer profile, complete purchase history, customer search results

Pre-conditions: User is authenticated and has appropriate permissions

Post-conditions: Customer record is created/updated, purchase history is maintained

Error Handling: Prevent duplicate customer creation, validate email format, handle missing required fields

### 3.1.3. Bill Generation (FR-003)

Description: The system shall generate comprehensive bills for customer purchases with itemized details, automatic calculations, and inventory updates.

Input: Customer ID (integer), selected items array (inventory\_id, quantity, price), payment method (optional text)

Processing: Calculate line item totals, sum bill total, validate stock availability, update inventory levels, create bill record, generate PDF invoice

Output: Complete bill with itemized details, PDF invoice file, updated inventory levels, transaction confirmation

Pre-conditions: Valid customer exists, selected inventory items are available in sufficient quantities

Post-conditions: Bill is created, inventory is updated, PDF invoice is generated, transaction is recorded

Error Handling: Insufficient stock validation with rollback, transaction failure handling, PDF generation error management

### 3.1.4. Low Stock Alerts (FR-004)

Description: The system shall automatically monitor inventory levels and send notifications when stock falls below configurable thresholds.

Input: Current inventory levels (integer), threshold settings (integer), notification preferences (email/SMS)

Processing: Monitor stock levels continuously, compare against thresholds, trigger alerts when conditions are met, track alert status

Output: Email notifications, SMS alerts, alert status tracking, low stock reports

Pre-conditions: Notification settings are configured, Twilio credentials are valid, email settings are configured

Post-conditions: Alerts are sent successfully, alert status is tracked, low stock items are identified

Error Handling: Graceful handling of notification failures, retry mechanisms, fallback notification methods

### 3.1.5. PDF Invoice Generation (FR-005)

Description: The system shall generate professional PDF invoices for bills with customizable formatting and business branding.

Input: Bill data (bill\_id), customer information, item details, business information

Processing: Format data for PDF, apply business branding, generate PDF using ReportLab, create downloadable file

Output: Professional PDF invoice file, download link, invoice preview

Pre-conditions: Valid bill exists with complete data

Post-conditions: PDF file is generated, download link is available, invoice is properly formatted

Error Handling: Handle PDF generation errors, manage file storage issues, provide alternative formats if needed

## 3.2. Non-Functional Requirements

### 3.2.1. Performance Requirements

• NFR-001: All user interface interactions must have a system response time of less than three seconds;

• NFR-002: Database queries must finish in one second for routine operations;

• NFR-003: PDF generation must finish in five seconds for typical invoice sizes.   
• NFR-004: The system can accommodate up to 100 users at once without experiencing any performance issues;

• NFR-005: On typical internet connections, page loads should take less than two seconds.   
• NFR-006: Within 30 seconds of the trigger, SMS notifications must be sent   
• NFR-007: Within 60 seconds of the trigger, email notifications must be sent.

### 3.2.2. Security Requirements

• NFR-008: All API endpoints must have the appropriate authorization and authentication;

• NFR-009: HTTPS/TLS protocols must be used to encrypt sensitive data while it is in transit.  
• NFR-010: All credentials and sensitive configurations must use environment variables.   
  
• NFR-011: Input validation will stop cross-site scripting and SQL injection attacks.   
• NFR-012: Secure authentication techniques must be used for database connections.   
• NFR-014: For auditing purposes, every user action must be recorded.

### 3.2.3. Usability Requirements

•NFR-015: Users with a basic understanding of computers should find the interface easy to use;

•NFR-016: The system should provide concise, actionable error messages

•NFR-017: All application pages should have consistent navigation.   
•NFR-018: Real-time validation feedback will be provided by the forms;

•NFR-019: The system will be available on desktop and tablet computers.   
• NFR-020: The user interface must adhere to contemporary design standards.   
• NFR-021: Complex features will have access to help documentation.

## 3.3. External Interface Requirements

### 3.3.1. User Interfaces

The system shall provide the following user interface components:

• Web-based interface using Next.js framework with responsive design

• Form-based interactions for data entry and management

• Dashboard with key metrics and quick access to common functions

• Search and filter capabilities for inventory and customer data

• Breadcrumb navigation for easy page traversal

• Responsive design supporting desktop (1920x1080) and tablet (768x1024) resolutions

### 3.3.2. Hardware Interfaces

The system has the following hardware interface requirements:

• Standard web browsers (Chrome, Firefox, Safari, Edge) on desktop and tablet devices

• Minimum screen resolution of 1024x768 pixels

• Internet connectivity for web access and external service communication

### 3.3.3. Software Interfaces

The system interfaces with the following software components:

• Django REST API for backend communication and business logic

• PostgreSQL database for data persistence and management

• Twilio API for SMS notification delivery

• ReportLab library for PDF generation and formatting

• Django CORS headers for cross-origin resource sharing

• Next.js framework for frontend development and routing

### 3.3.4. Communications Interfaces

The system uses the following communication protocols:

• HTTP/HTTPS for web communication and API requests

• RESTful API design patterns for backend communication

• JSON data format for API request and response payloads

• SMTP for email notification delivery

• Twilio REST API for SMS message delivery

• TCP/IP for database connections

# 4. Problem Identification & Analysis

## 4.1. Problem Statement

Small businesses face significant challenges in managing their inventory and billing operations efficiently. The current manual or semi-automated approaches lead to:

• Inefficient stock management resulting in stockouts or overstock situations that impact cash flow and customer satisfaction

• Time-consuming manual billing procedures that reduce productivity and increase human error rates

• Lack of automated notifications for critical business events such as low stock levels, leading to missed opportunities and operational inefficiencies

• Difficulty in maintaining accurate customer records and purchase history, limiting the ability to provide personalized service

• Limited reporting capabilities that prevent data-driven business decisions and strategic planning

• High costs associated with enterprise-level solutions that are often too complex for small business needs

• Inconsistent data management across different business processes, leading to data integrity issues

## 4.2. Analysis of Existing Solutions/Literature Review

A comprehensive analysis of existing solutions reveals several approaches to inventory and billing management:

Enterprise Solutions:

• QuickBooks: Comprehensive accounting software with inventory features, but high cost.

• Oracle NetSuite: Cloud-based ERP with strong inventory management, but high cost

Small Business Solutions:

• Shopify: E-commerce site with basic inventory features but not suitable for traditional retail

Identified Gaps and Limitations:  
• Fewer customization options for particular business needs;

• Outdated interfaces and poor user experience in many solutions;

• Absence of contemporary web-based architecture in reasonably priced solutions;

• Inadequate attention to the needs and workflows of small businesses

## 4.3. Proposed Approach and Innovation

The proposed approach addresses the identified gaps through innovative use of modern technologies and focused design:

Technical Approach:

• Django + Next.js, two contemporary web technologies, provide a stable, expandable, and maintainable architecture.

• An extensible codebase and clear concern separation are achieved through the use of RESTful API design.

• PostgreSQL database for data storage that complies with ACID

• Responsive design for a contemporary user experience and accessibility across multiple devices

Innovative Features:

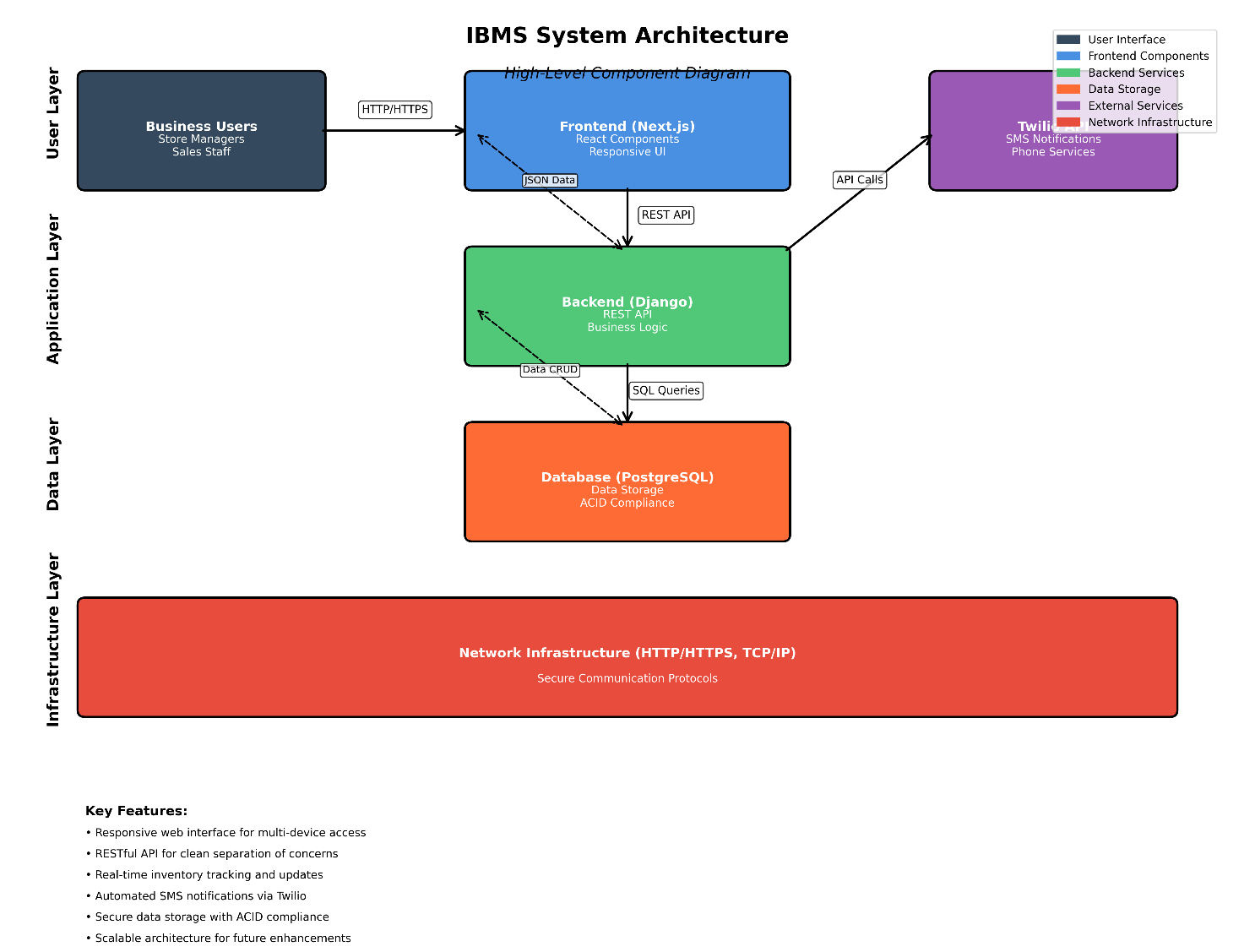
• Automated threshold monitoring with adjustable alert settings;

• Integrated SMS notifications for real-time low stock alerts via Twilio API

• Transaction rollback capabilities and real-time inventory updates throughout the billing process   
  
• Expert creation of PDF invoices using editable templates Scalable architecture that can expand to meet business needs; a contemporary, user-friendly interface created especially for non-technical users

• AI image recognition and a barcode reader for stock updates

# 5. System Architecture (High-Level Design)



# 6. Future Enhancements

Possible enhancements include:

• Support for multiple locations for companies with several physical stores;

• Predictive modeling and advanced analytics and business intelligence

• Developing mobile applications for iOS and Android platforms;

• Integration with e-commerce platforms for online sales management;

# AI Tool Usage Disclosure

This Software Requirements Specification document was primarily developed by the student, Jayakrishnan j(Reg No: 24pmc122) AI tools like cursor and chat-gpt were utilized to assist with brainstorming initial concepts for user roles, non-functional requirements, and for proofreading grammatical errors. All core analysis, detailed requirements, and critical decision-making reflect the original work and understanding of the author.