FUCHSIUS

Mobile Point of Sale (POS) System

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to describe in detail the requirements of the Mobile Point of Sale (POS) System. The system is designed to enable businesses, particularly small and medium-sized enterprises (SMEs), to handle all essential retail operations directly through a mobile device.

The POS system will support sales transactions, inventory management, reporting, and customer relationship management, all within a user-friendly mobile application. It will also provide features such as barcode scanning, digital receipt generation, offline sales support, and integration with receipt printers.

By documenting the requirements, this SRS ensures a shared understanding between stakeholders (business owners, managers, developers, testers, and end-users). The ultimate goal is to deliver a secure, reliable, scalable, and easy-to-use mobile solution that improves operational efficiency and decision-making for retail businesses.

1.2 Scope

The Mobile POS System will provide the following functionalities:

- **Sales Transaction Processing:** Cashiers can search or scan items, apply discounts or promotions, and complete sales using various payment methods (cash, credit/debit card, QR code, or mobile wallets).
- **Barcode Scanning:** The application will utilize the mobile device's camera for fast and accurate scanning of barcodes or QR codes.
- **Inventory Management:** Administrators and managers will be able to add, update, and remove products from the system. Stock levels will automatically adjust after each transaction, and alerts will be generated for low-stock items.
- **Receipt Generation:** After each sale, the system will generate receipts that can be printed (via Bluetooth/USB printers), emailed, or sent via SMS/WhatsApp.
- **Customer Management:** Businesses can store and manage customer details, track purchase histories, and send targeted offers or promotions.

- **Reporting and Analytics:** The system will generate daily, weekly, and monthly reports. Reports can be filtered by user, date, or product and exported in PDF or Excel formats.
- **Offline Functionality:** In cases where internet connectivity is unavailable, the application will allow transactions to continue. Data will be synced automatically with the backend when connectivity is restored.

The system will be deployed as a mobile application (Android/iOS) and will connect to a cloud-based backend server and centralized database for data synchronization and reporting.

1.3 Definitions, Acronyms, and Abbreviations

- **POS (Point of Sale):** The system where transactions between the customer and the business are completed.
- **SKU (Stock Keeping Unit):** A unique identifier for each product.
- **Invoice:** A document detailing the products or services purchased, payment amount, and customer details.
- **RBAC (Role-Based Access Control):** A system where access to functionality is restricted based on user roles.
- **User:** Any individual interacting with the system, such as Admin, Cashier, or Manager.

2. Overall Description

2.1 Product Perspective

The Mobile POS System is an independent mobile application that integrates with backend services via secure APIs. The backend will be responsible for storing and processing data, generating reports, managing inventory, and handling user authentication.

The system must work in both **online** and **offline** modes. In online mode, data is transmitted in real-time to the backend server. In offline mode, sales and updates are temporarily stored locally on the device and synced with the server once internet connectivity is restored.

The system will also be compatible with external hardware such as barcode scanners and Bluetooth/USB receipt printers, ensuring flexibility for businesses.

2.2 User Classes and Characteristics

- **Admin:** Responsible for managing the entire system. Has access to all features including user account management, system configuration, reporting, and inventory control. Admins are typically IT staff or business owners.
- **Cashier:** Performs daily sales transactions. Cashiers require a simple, user-friendly interface that allows them to quickly search or scan products, complete transactions, and issue receipts. They have limited access to administrative functions.
- **Manager:** Oversees sales and inventory performance. Managers can generate and view reports, check stock levels, and monitor employee performance but may not necessarily perform sales transactions.

2.3 Operating Environment

- **Mobile Application:** Runs on Android 8.0+ and iOS 12+ devices.
- **Backend Framework:** Prisma ORM for database management, built on Node.js or equivalent.
- **Database:** MySQL / Firebase / MongoDB.
- **Hardware Support:** Compatibility with Bluetooth/USB receipt printers and mobile device cameras for barcode scanning.
- **Network Requirements:** Wi-Fi or mobile data for synchronization.

2.4 Constraints

- Minimum 2GB RAM required for smooth operation.
- Application must remain functional in low-bandwidth environments by supporting offline operations.
- Data must be encrypted both at rest (in the database) and in transit (via HTTPS).
- Mobile devices must support camera-based scanning.
- Integration with external SMS/email services depends on third-party providers.

3. Functional Requirements

3.1 User Authentication

- The system must allow secure login using email/password or PIN.
- Admins must be able to add, edit, and remove user accounts.
- Users should be able to reset their password via email or by requesting admin intervention.
- Authentication must be role-based to ensure different levels of system access.

3.2 Sales Transactions

- Users must be able to add products to the bill by scanning a barcode or manually searching.
- The system must calculate totals including taxes, discounts, and promotional offers.
- Multiple payment options should be supported, including cash, debit/credit card, QR payments, and mobile wallets.
- Receipts must be generated and provided to customers either in printed form or digitally.
- Completed transactions must be logged in the database for future reporting.

3.3 Inventory Management

- Admins and managers must be able to add, edit, and delete products. Each product must include SKU, name, category, price, stock level, and applicable taxes.
- The system should automatically update stock levels after each sale.
- Low-stock notifications should be sent to managers/admins when inventory falls below predefined thresholds.
- Support for multi-location inventory management should be considered in future versions.

3.4 Reporting

- The system must generate sales reports categorized by day, week, and month.
- Reports should be exportable in PDF and Excel formats for external use.
- Filtering and sorting options must be available by product, date range, user, or transaction type.
- Graphical visualizations (charts and graphs) should be included in the reports to provide insights.

3.5 Customer Management

- The system must allow businesses to add and maintain customer profiles including name, contact number, email, and purchase history.
- Businesses must be able to view customer transaction history for loyalty and analysis purposes.
- The system must support sending promotional offers, discounts, or receipts to customers via email or SMS.

3.6 Settings and Configuration

- Tax rates and discount rules must be configurable by the Admin.
- Stores must be able to customize their receipts with logos, store names, addresses, and other business details.
- The system should allow integration with various receipt printers.

3.7 Offline Mode

- Users must be able to complete sales transactions even without internet access.
- All offline transactions must be stored locally and automatically synchronized with the backend once connectivity is available.

4. Non-Functional Requirements

4.1 Performance

- Each sales transaction must be completed in under 2 seconds to ensure smooth customer experiences.
- Reports must be generated in under 5 seconds, even with large datasets.

4.2 Security

- All communication between mobile apps and the backend must be encrypted using HTTPS/TLS.
- User passwords must be hashed using bcrypt or argon2.
- Role-based access control must be enforced to restrict user permissions.
- Sensitive data (customer information, transaction details) must be encrypted in the database.

4.3 Usability

- The application must feature a simple and intuitive interface that can be used with minimal training.
- The system must be optimized for mobile screens, ensuring touch-friendly elements and fast navigation.
- The interface should be responsive and work equally well on smartphones and tablets.

4.4 Reliability and Availability

- The system must prevent data loss in case of unexpected crashes by auto-saving transactions.
- Offline data synchronization must ensure that no data is lost during reconnection.
- The backend must provide at least 99.5% uptime for availability.

4.5 Maintainability

- The codebase must follow modular programming principles for easier updates and maintenance.
- Documentation must be provided for developers, including API references and deployment instructions.

- The system should support version upgrades without affecting existing data.

5. Assumptions and Dependencies

- Devices must have functional cameras for barcode scanning.
- Receipt printer integration is dependent on device compatibility.
- SMS/email notifications depend on external providers such as SMTP servers or SMS gateways.
- Cloud synchronization assumes stable internet connectivity.