**SOFTWARE REQUIREMENT SPECIFICATION (SRS) DOCUMENT FOR COLLEGE CANTEEN MANAGEMENT SYSTEM**

**1. INTRODUCTION:**

**1.1 Purpose:**

**The purpose of this College Canteen Management System is to streamline the operations of the canteen and enhance the experience for both users (students, staff) and administrators. The system will provide an integrated platform for browsing available food items, placing orders, managing payments, and tracking inventory in real-time. By improving operational efficiency and ensuring quick decision-making, the system aims to provide a smoother, more efficient canteen experience.**

**1.2 Scope:**

**This system will support core functionalities such as:**

* **Order Management: Users can browse the menu, place orders, and check the status of their orders.**
* **Payment Processing: Supports multiple payment methods like cash, card, and digital wallets.**
* **Inventory Management: Tracks available stock, manages food item quantities, and sends alerts for low-stock items.**
* **Reporting: Provides administrators with detailed reports on sales, inventory, and order history.**

**2. SYSTEM DESCRIPTION:**

**The College Canteen Management System is designed for both users (students, staff) and administrators to interact with the system. Key features include:**

**2.1 Order Management:**

* **Browse Food Items: Users can view available food items, check their details like prices and descriptions.**
* **Place Order: Users can select food items, customize their order (e.g., quantity), and place the order.**
* **Order Status: Users can track the status of their orders (e.g., Pending, Completed).**
* **Order Update (Admin only): Admins can modify or cancel orders before processing.**

**2.2 Payment Processing:**

* **Generate Bill: The system calculates the total price of the selected items, applies taxes, and generates a final bill.**
* **Payment Methods: Supports payment through cash, debit/credit card, and digital wallets.**
* **Transaction History: Both users and admins can view the transaction history, including all completed payments.**

**2.3 Inventory Management:**

* **Track Inventory: The system automatically updates inventory when items are sold or restocked.**
* **Low Stock Alerts: When stock of a food item falls below the threshold, an alert is generated.**
* **Inventory Reports: Provides reports detailing the current stock, sales trends, and restocking needs.**

**2.4 Reporting and Analytics:**

* **Sales Report: Generates daily, weekly, and monthly sales reports, including revenue and item-wise sales.**
* **Inventory Report: Provides inventory summaries to track movement and optimize stock management.**

**3. SYSTEM REQUIREMENTS:**

**3.1 Functional Requirements:**

**3.1.1 Order Management:**

* **Place Order: Customers can select items, add them to the cart, and place the order.**
* **View Order: Displays the order details, including items, quantities, prices, and order status.**
* **Update Order: Admin can update orders to modify quantities or change items before finalizing.**

**3.1.2 Payment Processing:**

* **Generate Bill: Automatically calculates the order total, including taxes and discounts, and generates an invoice.**
* **Payment Methods: Supports various payment methods, including cash, card, and digital wallets.**
* **Transaction History: Maintains a log of all transactions for reference and accounting.**

**3.1.3 Inventory Management:**

* **Track Inventory: The system tracks and updates inventory levels as orders are placed and processed.**
* **Low Stock Alerts: Notifications are generated when an item's stock level is below a pre-defined threshold.**
* **Inventory Reports: Generates periodic inventory reports to monitor stock levels and trends.**

**3.1.4 Reporting and Analytics:**

* **Sales Report: Provides insights into sales trends, including revenue generated over various periods.**
* **Inventory Report: Offers insights into the stock levels and consumption trends for efficient stock management.**

**3.2 Non-Functional Requirements:**

**3.2.1 Performance Requirements:**

* **Transaction Speed: The system must process each order and payment transaction within 5 seconds.**
* **Simultaneous Transactions: The system should be able to handle up to 10 simultaneous transactions without performance degradation.**

**3.2.2 Usability:**

* **User Interface: The system must feature an intuitive user interface for both customers and staff, making navigation easy and efficient.**
* **Feedback: The system should provide instant feedback after actions (e.g., confirmation messages for orders, payments).**

**3.2.3 Reliability:**

* **Availability: The system should be available 24/7 with downtime only during maintenance windows.**
* **Error Handling: Graceful error handling with clear, actionable messages for users and staff (e.g., insufficient stock or invalid payment).**

**3.2.4 Security:**

* **Data Security: Payment information and personal details must be encrypted to protect customer privacy.**
* **Access Control: Role-based access control (RBAC) to limit admin access to sensitive functionalities like inventory management.**

**4. INTERFACE REQUIREMENTS:**

**4.1 User Interface:**

* **Customer Interface: A touchscreen-based or mobile app interface where users can browse the menu, place orders, and make payments.**
* **Admin Interface: A terminal-based interface for administrators to manage orders, inventory, payments, and reports.**

**4.2 Hardware Interface:**

* **POS Terminal: Standard POS hardware such as receipt printers, barcode scanners, and card readers.**
* **System Compatibility: The system will work with commonly available POS peripherals, ensuring flexibility and scalability.**

**5. TECH STACK:**

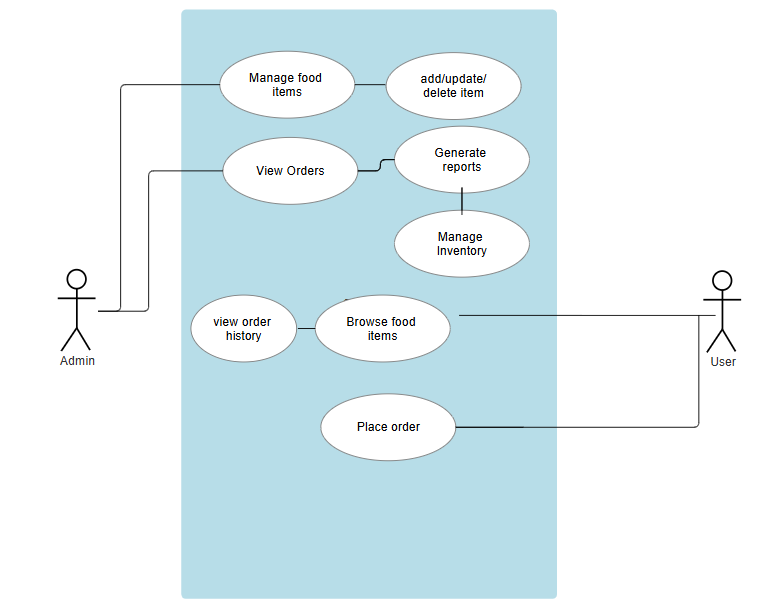
**The College Canteen Management System will be developed using the following technologies:**

* **Java: For implementing the core features with object-oriented programming principles.**
* **JDBC with MySQL: For database operations (CRUD) on orders, payments, inventory, and user data.**
* **Java I/O: For handling file operations and generating reports such as transaction history and sales reports.**
* **Exception Handling: To gracefully handle errors (e.g., invalid payments, low stock) and recover smoothly.**
* **Multithreading and Synchronization: To ensure concurrency for multiple users and transactions, without affecting data integrity.**
* **Git: For version control and collaboration among the development team.**

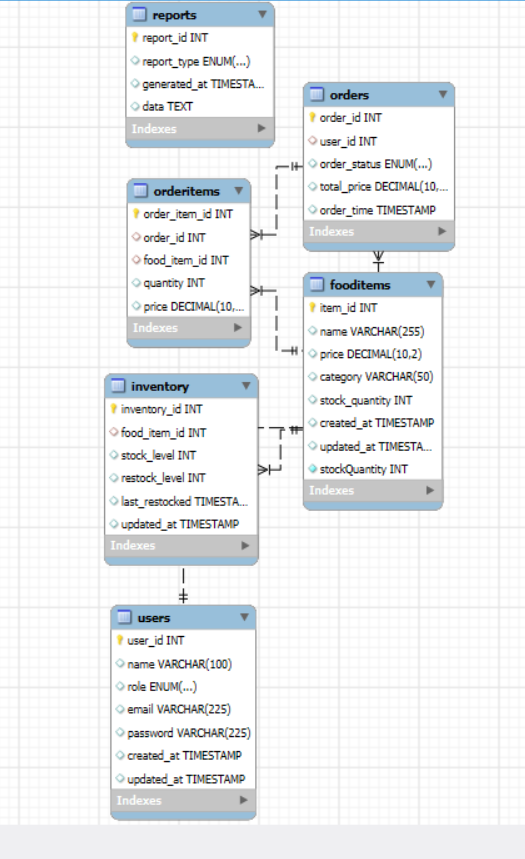
**6. DIAGRAMS**

**6.1 Use case Diagram:**

**COLLEGE CANTEEN MANAGEMENT SYSTEM**

****

**6.2 UML Diagram:**



**6.3 Class Diagram:**

