**System Requirement Software for Hotel management**

### Introduction

* **Purpose**: The purpose of this document is to define the functional and non-functional requirements for a restaurant management system that will streamline operations such as table reservation, order management, billing, and inventory control.
* **Scope**: This system will cater to restaurants of various sizes, aiming to automate and simplify management processes to ensure efficient operations and superior customer service.
* **Definitions, Acronyms, and Abbreviations**:
  + SRS: Software Requirements Specification
  + POS: Point of Sale
  + GUI: Graphical User Interface

### 2. Overall Description

* **Product Perspective**: This restaurant management system will be a standalone application that integrates multiple functionalities to manage restaurant operations. It will interface with payment gateways, inventory systems, and customer relationship management (CRM) tools.
* **Operating Environment**: The system will operate on Windows, macOS, and Linux platforms, and support web-based access through modern web browsers.
* **User Characteristics**: Users will include restaurant staff such as waiters, managers, chefs, and administrators.
* **Constraints**: The system must comply with local and international regulations regarding data privacy and food safety standards.

### 3. Functional Requirements

* **Table Reservation Management**:
  + Allow customers to reserve tables online or through a mobile app.
  + Manage table availability and reservation schedules.
* **Order Management**:
  + Enable waitstaff to take orders via mobile devices or tablets.
  + Facilitate real-time order tracking and status updates.
* **Menu Management**:
  + Allow for easy updates to menu items, descriptions, and prices.
  + Manage special offers and promotions.
* **Billing and Payment**:
  + Generate itemized bills for customers.
  + Process payments via credit cards, debit cards, and mobile payment options.
* **Inventory Management**:
  + Track inventory levels and generate alerts for low stock items.
  + Manage supplier information and order history.
* **Reporting and Analytics**:
  + Generate sales, inventory, and customer satisfaction reports.
  + Provide data analytics for decision-making.

### 4. Non-Functional Requirements

* **Performance Requirements**:
  + The system should handle up to 300 concurrent users without performance issues.
* **Security Requirements**:
  + Implement data encryption and secure login mechanisms.
  + Conduct regular security audits and updates.
* **Usability Requirements**:
  + Ensure an intuitive and user-friendly interface.
  + Minimize the need for extensive user training.
* **Availability and Reliability Requirements**:
  + Ensure the system has 99.9% uptime.
* **Maintainability and Scalability Requirements**:
  + Design the system to be easily maintainable and scalable for future growth.

### 5. Interface Requirements

* **User Interfaces**:
  + Provide a graphical user interface (GUI) for restaurant staff.
  + Offer a web interface for online reservations and orders.
* **Hardware Interfaces**:
  + Compatibility with standard restaurant hardware like POS terminals, printers, and barcode scanners.
* **Software Interfaces**:
  + Integration with third-party payment gateways and inventory systems.
* **Communication Interfaces**:
  + Support for standard communication protocols such as HTTP, HTTPS, and FTP.

### 6. Other Requirements

* **Data Migration and Conversion**:
  + Offer tools for migrating data from existing systems.
* **Training and Documentation**:
  + Provide user manuals, training sessions, and support documentation.
* **Legal and Regulatory Requirements**:
  + Ensure compliance with GDPR, PCI-DSS, and other relevant regulations.

### 7. Appendices

* **Glossary**:
  + Definitions of terms used in the document.
* **Use Case Diagrams**:
  + Visual representations of system use cases and actors.
* **Data Flow Diagrams**:
  + Diagrams showing data flow within the system.
* **Entity-Relationship Diagrams**:
  + Diagrams depicting database structure and relationships.
* **User Interface Mockups**:
  + Prototypes and mockups of the user interfaces.