1. Restaurant Management System
2. System Requirements
   * Functional Requirements
   * Non - Functional Requirements
   * Software Requirements
3. System Design
   * Architecture of the System
   * Subsystem Decomposition
   * Low-Level Design
   * Database Design

**Restaurant Management System**

A restaurant management system is a software that helps the restaurant industry streamline their food business operations. It provides a complete set of features including Point of Sales (POS) system, payment processing, table reservations, inventory management , accounting and employee management.

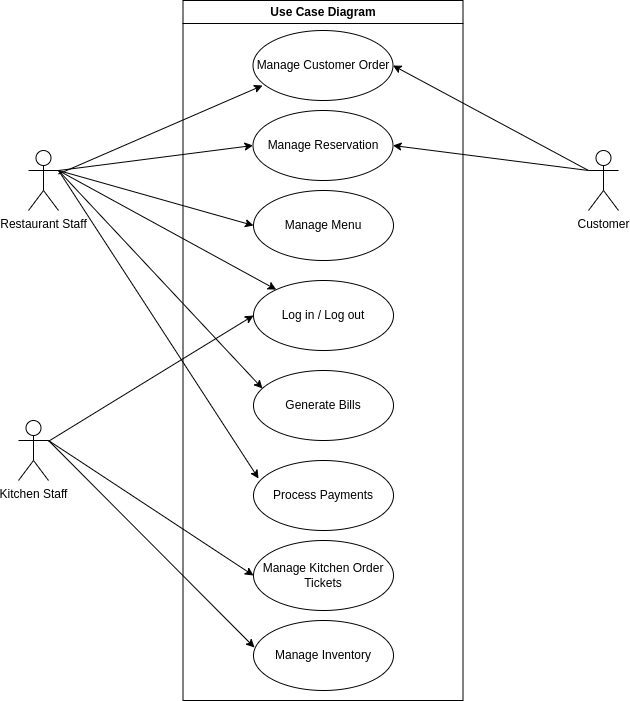
**System Requirements**

The system we will be designing will provide the most important features that every restaurant management system must offer. However, do note that a restaurant management system can have other sub-systems for accounting, employee managment, etc.

**Functional Requirements**

The restaurant management system should at least include the following features,

1. Allow to add and remove item to order
2. Allow to add and remove table reservation
3. Allow to add and remove menu items
4. Generate bills
5. Manage payment of bills
6. Generate kitchen order tickets

A better understanding of the functional requirements can be gained from the use-case diagram below.  


**Non - Functional Requirements**

**Usability**  
The system should provide an interactive user-friendly interface that is easily understandable for all users.

**Availability**  
The System should be available at least during the restaurant operating hours and must be recovered within an hour or less if it fails. The system should respond to the requests within two seconds or less.

**Dependability**  
The system should provide consistent performance with easy tracking of records and updating of records.

**Maintainability**  
The software should be easily maintainable and adding new features and making changes to the software must be as simple as possible.

**Security**  
Only authorized users must be able to access the system and view and modify the data.

**Software Requirements**

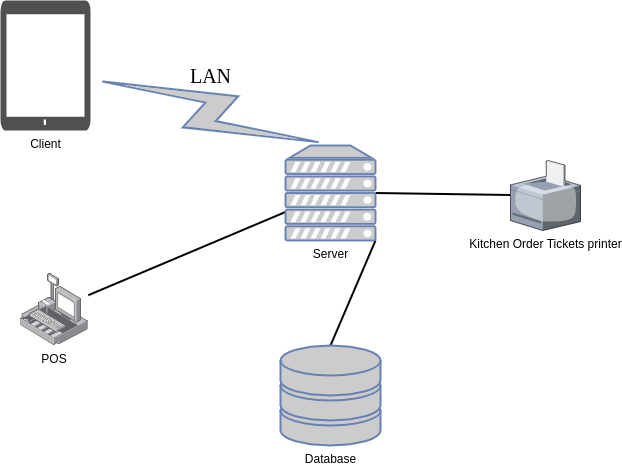
1. A server running Windows Server/Linux OS.
2. A backend language such as like Java, Python to process the orders.
3. Front-end frameworks like Angular/React/Vue for the user interface.
4. Relational DBMS such as MySQL, PostgreSQL.

**System Design**

With our understanding of the functional and non-functional requirements of our system, we will now look at the system architecture, decomposition, and database design.

**Architecture of the System**

The restaurant management system follows a simple 3 tier client/server architecture. The client can use web browsers on a tablet to access the system (restaurant menu) through the local area network of the restaurant using the HTTPS protocol.

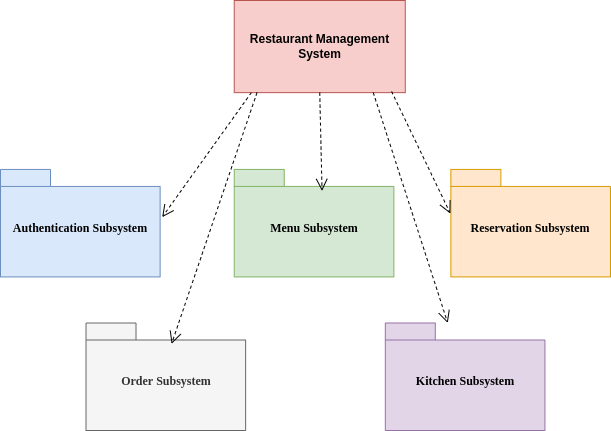


The middle tier which includes the server presents the website to the user and controls the business logic. It controls the interactions between the application and the user. The server also send the user orders to both the Point of Sales (POS) system and Kitchen Order Tickets printers. Common web server technology used here can be Apache, Nginx, etc.

The data tier maintains the application's data such as order data, menu data, reservation data, etc. It stores these data in a relational database management system (RDBMS) like PostgreSQL. The client tier interacts with the server to make requests and retrieve data from the database. It then displays to the user the data retrieved from the server.

**Subsystem Decomposition**

Decomposing the system into smaller units called subsystems will help reduce the complexity of the system. Subsystems are just packages holding related classes. Our restaurant management system is also decomposed into subsystems as follows. The major subsystems are 'Authentication', 'Menu', 'Reservation', 'Order', and 'Kitchen' systems.



The Authentication subsystem authenticates a user to grant access based on the role of the user.

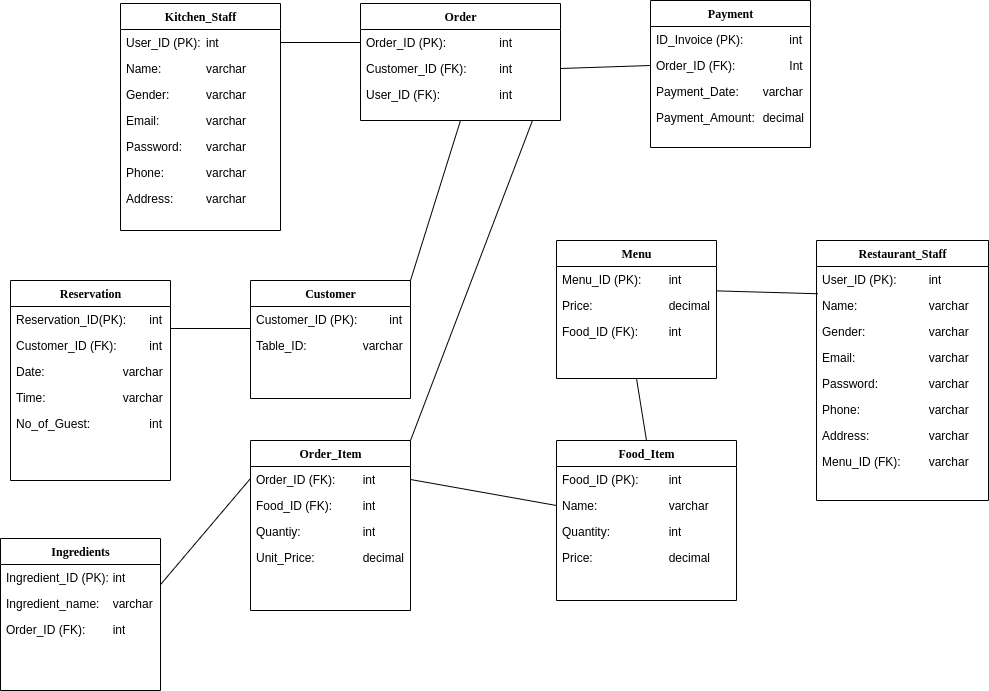
The Menu subsystem generates the restaurant menu, which involves assigning a price to each item.

The Reservation system facilitates the booking of table by the customer and handle the payment of the booking fee in the event of a no-show.

The Order subsystem registers the user selection from the menu. It compile the order for the Point of Sales system for billing. In addition, it send the data to the Kitchen subsystem for processing.

The Kitchen subsystem deals with printing of the Kitchen Order Tickets and adjusting the inventory.

**Database Design**

A restaurant management system software needs to store data about the Order, Menu, Reservations, etc. Therefore, we have identified the major tables that will be implemented on the selected RDBMS.  


The above database diagram shows the schema for the restaurant management software database.

Generally, there are three types of relationships in a relational database system. These are one-to-one, one-to-many, and many-to-many relationships. The system under consideration has one-to-many and many-to-many relationships.

The Reservation, Order and Menu tables store data of the reservations, orders and menu respectively. Each Order is associated with one or many order items. Each order item has a food item and ingredients associated with it. The Menu table acts as relation between the Restaurant\_Staff and Food\_Item tables. The Payment table stores the details about a specific order.