RESTAURANT MANAGEMENT SYSTEM

### **Abstract**

A Restaurant Management System (RMS) is a software application designed to streamline and automate various operations within a restaurant. This mini project outlines the development of an RMS using Java for the frontend and a Database Management System (DBMS) for the backend. The system addresses key functionalities such as customer order management, inventory tracking, staff scheduling, reservations, and financial transactions. By integrating these components, the RMS aims to enhance operational efficiency, reduce manual errors, and provide better service to customers.

This project employs Java Swing for the user interface, offering a desktop application experience. The backend is implemented using a relational database, such as MySQL, to manage data persistence and ensure data integrity. The integration between the Java application and the database is facilitated through JDBC, with the potential use of Hibernate for Object-Relational Mapping (ORM).

The project begins with setting up the database, designing the frontend, and establishing the connection between the two. The business logic is then implemented to handle various operations, such as placing orders, managing inventory, scheduling staff shifts, and handling reservations. Finally, the system undergoes rigorous testing to ensure functionality and reliability.

By the end of this project, the resulting RMS will provide a robust platform for managing the complex operations of a restaurant, ultimately leading to improved management and customer satisfaction.

### **Implementation Plan**

#### 1.Requirements and Features

* **Customer Orders:**
  + Place, update, and track orders.
  + Generate bills and process payments.
* **Inventory Management:**
  + Track inventory levels.
  + Add or update inventory items.
* **Staff Scheduling:**
  + Manage staff shifts and schedules.
* **Reservations:**
  + Manage table reservations.
* **Financial Transactions:**
  + Track income and expenses.

#### 2.Technologies Used

* **Frontend:** Java (Swing for GUI)
* **Backend:** DBMS (MySQL)
* **Database Connectivity:** JDBC (Java Database Connectivity)

#### 3.Database Design

**Tables:**

* **Customers**: **customer\_id**, **name**, **contact\_info**
* **Orders**: **order\_id**, **customer\_id**, **order\_date**, **status**
* **Order\_Items**: **order\_item\_id**, **order\_id**, **menu\_item\_id**, **quantity**
* **Menu\_Items**: **menu\_item\_id**, **name**, **description**, **price**
* **Inventory**: **item\_id**, **name**, **quantity**, **price**
* **Staff**: **staff\_id**, **name**, **role**, **schedule**
* **Reservations**: **reservation\_id**, **customer\_id**, **reservation\_date**, **status**
* **Transactions**: **transaction\_id**, **order\_id**, **amount**, **transaction\_date**

#### 4. Implementation Steps

**Step 1: Setup the Database**

* Install and configure MySQL.
* Create the necessary tables and relationships.

**Step 2: Design the Frontend**

* Use Java Swing to design the UI.
* Implement forms and views for each functionality.

**Step 3: Connect Frontend to Backend**

* Use JDBC for database operations.
* Write code to handle CRUD operations for each entity.

**Step 4: Implement Business Logic**

* Handle order processing, inventory updates, staff scheduling, reservations, and financial transactions.

**Step 5: Testing and Debugging**

* Test all functionalities.
* Debug and fix issues.

### **Database Setup Example**

**CREATE DATABASE restaurant\_db;**

**USE restaurant\_db;**

**CREATE TABLE Customers (**

**customer\_id INT AUTO\_INCREMENT PRIMARY KEY,**

**name VARCHAR(100),**

**contact\_info VARCHAR(100)**

**);**

**CREATE TABLE Orders (**

**order\_id INT AUTO\_INCREMENT PRIMARY KEY,**

**customer\_id INT,**

**order\_date DATETIME,**

**status VARCHAR(20),**

**FOREIGN KEY (customer\_id) REFERENCES Customers(customer\_id)**

**);**

**CREATE TABLE Order\_Items (**

**order\_item\_id INT AUTO\_INCREMENT PRIMARY KEY,**

**order\_id INT,**

**menu\_item\_id INT,**

**quantity INT,**

**FOREIGN KEY (order\_id) REFERENCES Orders(order\_id)**

**);**

**CREATE TABLE Menu\_Items (**

**menu\_item\_id INT AUTO\_INCREMENT PRIMARY KEY,**

**name VARCHAR(100),**

**description TEXT,**

**price DECIMAL(10, 2)**

**);**

**CREATE TABLE Inventory (**

**item\_id INT AUTO\_INCREMENT PRIMARY KEY,**

**name VARCHAR(100),**

**quantity INT,**

**price DECIMAL(10, 2)**

**);**

**CREATE TABLE Staff (**

**staff\_id INT AUTO\_INCREMENT PRIMARY KEY,**

**name VARCHAR(100),**

**role VARCHAR(50),**

**schedule TEXT**

**);**

**CREATE TABLE Reservations (**

**reservation\_id INT AUTO\_INCREMENT PRIMARY KEY,**

**customer\_id INT,**

**reservation\_date DATETIME,**

**status VARCHAR(20),**

**FOREIGN KEY (customer\_id) REFERENCES Customers(customer\_id)**

**);**

**CREATE TABLE Transactions (**

**transaction\_id INT AUTO\_INCREMENT PRIMARY KEY,**

**order\_id INT,**

**amount DECIMAL(10, 2),**

**transaction\_date DATETIME,**

**FOREIGN KEY (order\_id) REFERENCES Orders(order\_id)**

**);**

### **Frontend Design Example**

**(Using Java Swing)**

**import javax.swing.\*;**

**import java.awt.\*;**

**public class RestaurantManagementSystem {**

**private JFrame frame;**

**public RestaurantManagementSystem() {**

**frame = new JFrame("Restaurant Management System");**

**frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);**

**frame.setSize(600, 400);**

**JMenuBar menuBar = new JMenuBar();**

**JMenu menu = new JMenu("Menu");**

**JMenuItem addMenuItem = new JMenuItem("Add Menu Item");**

**JMenuItem viewMenuItems = new JMenuItem("View Menu Items");**

**menu.add(addMenuItem);**

**menu.add(viewMenuItems);**

**JMenu orders = new JMenu("Orders");**

**JMenuItem newOrder = new JMenuItem("New Order");**

**JMenuItem viewOrders = new JMenuItem("View Orders");**

**orders.add(newOrder);**

**orders.add(viewOrders);**

**JMenu inventory = new JMenu("Inventory");**

**JMenuItem addInventoryItem = new JMenuItem("Add Inventory Item");**

**JMenuItem viewInventory = new JMenuItem("View Inventory");**

**inventory.add(addInventoryItem);**

**inventory.add(viewInventory);**

**JMenu reservations = new JMenu("Reservations");**

**JMenuItem addReservation = new JMenuItem("Add Reservation");**

**JMenuItem viewReservations = new JMenuItem("View Reservations");**

**reservations.add(addReservation);**

**reservations.add(viewReservations);**

**JMenu staff = new JMenu("Staff");**

**JMenuItem addStaff = new JMenuItem("Add Staff");**

**JMenuItem viewStaff = new JMenuItem("View Staff");**

**staff.add(addStaff);**

**staff.add(viewStaff);**

**JMenu transactions = new JMenu("Transactions");**

**JMenuItem viewTransactions = new JMenuItem("View Transactions");**

**transactions.add(viewTransactions);**

**menuBar.add(menu);**

**menuBar.add(orders);**

**menuBar.add(inventory);**

**menuBar.add(reservations);**

**menuBar.add(staff);**

**menuBar.add(transactions);**

**frame.setJMenuBar(menuBar);**

**frame.setVisible(true);**

**}**

**public static void main(String[] args) {**

**new RestaurantManagementSystem();**

**}**

**}**

### **Database Connection Example**

**import java.sql.Connection;**

**import java.sql.DriverManager;**

**import java.sql.SQLException;**

**public class DatabaseConnection {**

**private static final String URL = "jdbc:mysql://localhost:3306/restaurant\_db";**

**private static final String USER = "root";**

**private static final String PASSWORD = "password";**

**public static Connection getConnection() throws SQLException {**

**return DriverManager.getConnection(URL, USER, PASSWORD);**

**}**

**}**

### **Business Logic Example**

**(Adding a Menu Item)**

**import java.sql.Connection;**

**import java.sql.PreparedStatement;**

**import java.sql.SQLException;**

**public class MenuItemDAO {**

**public void addMenuItem(String name, String description, double price) throws SQLException {**

**String sql = "INSERT INTO Menu\_Items (name, description, price) VALUES (?, ?, ?)";**

**try (Connection conn = DatabaseConnection.getConnection();**

**PreparedStatement stmt = conn.prepareStatement(sql)) {**

**stmt.setString(1, name);**

**stmt.setString(2, description);**

**stmt.setDouble(3, price);**

**stmt.executeUpdate();**

**}**

**}**

**}**

### **Conclusion**

This mini project outlines the creation of a Restaurant Management System using Java for the frontend and a DBMS like MySQL for the backend. The system incorporates functionalities crucial for efficient restaurant management, and by following the implementation steps, it can be expanded and customized based on specific needs. The project aims to provide a foundation for developing a robust, user-friendly RMS that enhances operational efficiency and customer service.