UE22CS351A: Software Engineering

Implementation Document

Project Name: Restaurant Management System

PRANEEL PATEL(PES1UG22CS437) &

PRIYANSHU CHAUDHARY (PES1UG22CS455)

H-Section, 5TH Semester

BTech-CSE

CODE

```
import mysql.connector



# Connect to MySQL

mydb = mysql.connector.connect(

    host="localhost",

    user="root",

    password="root"

)



mycursor = mydb.cursor()



# Create the restaurant database

mycursor.execute("CREATE DATABASE IF NOT EXISTS restaurant_db")

mycursor.execute("USE restaurant_db")



# Create the login table for admin

mycursor.execute("""

    CREATE TABLE IF NOT EXISTS admin_login (

        username VARCHAR(50) NOT NULL,
```

```
        password VARCHAR(50) NOT NULL,
        PRIMARY KEY (username)
    )
""")

# Create the menu table
mycursor.execute("""
    CREATE TABLE IF NOT EXISTS menu (
        item_id INT AUTO_INCREMENT NOT NULL,
        item_name VARCHAR(50) NOT NULL,
        price DECIMAL(5, 2) NOT NULL,
        available BOOLEAN NOT NULL DEFAULT TRUE,
        PRIMARY KEY (item_id)
    )
""")

# Create the ingredients table
mycursor.execute("""
    CREATE TABLE IF NOT EXISTS ingredients (
```

```
        ingredient_id INT AUTO_INCREMENT NOT NULL,
        ingredient_name VARCHAR(50) NOT NULL,
        quantity DECIMAL(10, 2) NOT NULL,
        unit VARCHAR(10) NOT NULL,
        PRIMARY KEY (ingredient_id)
    )
""")

# Create the orders table
mycursor.execute("""
    CREATE TABLE IF NOT EXISTS orders (
        order_id INT AUTO_INCREMENT NOT NULL,
        customer_name VARCHAR(50) NOT NULL,
        item_id INT NOT NULL,
        quantity INT NOT NULL,
        status VARCHAR(20) NOT NULL DEFAULT 'Placed',
        PRIMARY KEY (order_id),
        FOREIGN KEY (item_id) REFERENCES menu(item_id)
    )
```

```
""")



# Create the transactions table

mycursor.execute("""

    CREATE TABLE IF NOT EXISTS transactions (

        transaction_id INT AUTO_INCREMENT NOT NULL,

        transaction_date DATE NOT NULL,

        order_id INT NOT NULL,

        total_amount DECIMAL(7, 2) NOT NULL,

        PRIMARY KEY (transaction_id),

        FOREIGN KEY (order_id) REFERENCES orders(order_id)

    )

""")



# Create the employees table

mycursor.execute("""

    CREATE TABLE IF NOT EXISTS employees (

        employee_id INT AUTO_INCREMENT NOT NULL,

        name VARCHAR(50) NOT NULL,
```

```
        role VARCHAR(50) NOT NULL,

        PRIMARY KEY (employee_id)

    )

""")



mydb.commit()
```

Implementation Document: Restaurant Management System

1. Introduction

This system manages restaurant operations, including order processing, menu management, billing, and inventory tracking for ingredients. The system will have two user roles:

- **Admin (Restaurant Manager): Manages menu, inventory, and employees.**
- **Customer: Places orders and views available menu items.**

2. System Features

Admin:

- **Add or remove menu items.**
- **Update prices for menu items.**
- **Track ingredient inventory.**
- **View sales transactions.**
- **Manage employee details.**

Customer:

- **Browse and order food from the menu.**
- **View order details.**
- **Pay the bill.**

3. Database Structure

The system uses a MySQL database with the following tables:

1. **`admin_login`**: Stores administrator credentials.

2. **`menu`**: Stores menu items.
3. **`ingredients`**: Stores details of ingredients for menu items.
4. **`orders`**: Logs customer orders.
5. **`transactions`**: Records details of sales.
6. **`employees`**: Stores employee information.

Table Structures:

admin_login Table:

- username: Admin username (Primary Key).
- password: Admin password.

menu Table:

- item_id: Unique item ID (Primary Key).
- item_name: Name of the menu item.
- price: Price of the item.
- available: Availability status.

ingredients Table:

- ingredient_id: Unique ID for each ingredient (Primary Key).
- ingredient_name: Name of the ingredient.
- quantity: Quantity available.
- unit: Measurement unit (e.g., grams, liters).

orders Table:

- order_id: Unique order ID (Primary Key).
- customer_name: Name of the customer.
- item_id: References menu items (Foreign Key).
- quantity: Quantity of the ordered item.
- status: Status of the order (e.g., placed, completed).

transactions Table:

- transaction_id: Unique ID for each transaction (Primary Key).
- transaction_date: Date of transaction.
- order_id: References orders (Foreign Key).
- total_amount: Total amount paid.

employees Table:

- employee_id: Unique employee ID (Primary Key).
- name: Employee name.
- role: Role of the employee (e.g., waiter, chef).

4. Implementation Details

4.1. Database Connection and Setup

The system connects to MySQL using mysql.connector. The following tables are created during setup: admin_login, menu, ingredients, orders, transactions, employees.

4.2. Admin Functionalities

- **Login:**
  Admin can log in using a username and password.
- **Add/Remove Menu Items:**
  Admin can add new menu items or remove existing ones.
- **Update Prices:**
  Admin can update prices for menu items.
- **Manage Inventory:**
  Admin can manage the quantity of ingredients required for the menu.
- **View Sales Transactions:**
  Admin can view transaction history with details such as order ID, item names, and total sales.
- **Manage Employees:**
  Admin can add, update, or remove employee records.

4.3. Customer Functionalities

- **Place Order:**
  Customers can select items from the menu and place an order.
- **View Order:**
  Customers can view the details of their orders.
- **Payment:**
  After placing an order, customers can proceed with payment.

5. Transaction Management

The system logs each transaction when a customer places an order. The log includes:

- **Date and time of the transaction.**
- **Total amount.**
- **Order details.**

6. Error Handling and Constraints

1. **Unique Item IDs: Each menu item and ingredient has a unique**
2. **Ingredient Availability:** Orders can only be placed if sufficient ingredients are available.

7. Improvements and Future Features

- **Employee Management:** Assign different roles (chef, waiter) to employees with specific permissions.
- **Reports**: Generate daily or weekly sales and inventory reports.
- **Reservation System**: Implement a table reservation system for the restaurant.

8. Conclusion

The Restaurant Management System efficiently handles menu management, order processing, inventory tracking, and employee management using a MySQL database. Admins can manage menus, track ingredients, view transactions, and handle staff, while customers can place orders and pay seamlessly. The system is scalable, with future enhancements like employee roles, reporting, and reservations, making it a valuable tool for optimizing restaurant operations and improving customer service.