INTRODUCTION

In today's fast-paced world, the need for streamlined and efficient business operations is more critical than ever. The Cafe Management System is designed to modernize the traditional management processes of a cafe, combining simplicity with robust functionality. This system utilizes Python as the programming backbone and MySQL as the database to create a seamless, user-friendly, and secure solution for cafe operations. Whether it's managing customers, processing orders, or analyzing sales, the Cafe Management System serves as a comprehensive tool to handle it all.

PURPOSE OF THE CAFE MANAGEMENT SYSTEM

The primary purpose of the Cafe Management System is to enhance operational efficiency, reduce manual errors, and improve the customer experience. By automating core tasks such as order placement, customer registration, menu management, and sales reporting, the system allows cafe staff to focus more on delivering quality service. Additionally, the system empowers administrators with analytical tools to track sales trends and manage inventory effectively.

KEY FEATURES AND FUNCTIONS

Admin Functions

1. Admin Login

- **Purpose**: Provides secure access to administrative functions.
- **Functionality**: Administrators log in using a predefined password to access advanced features such as customer management, order reviews, menu modifications, and sales reporting.

2. View All Customers

- **Purpose**: Enables administrators to monitor and manage customer data.
- **Functionality**: Displays a list of all registered customers, including their IDs, names, email addresses, and phone numbers.

3. View All Orders

- **Purpose**: Offers insight into the cafe's operational workflow by listing all orders.
- **Functionality**: Shows details of orders, such as order ID, customer ID, order date, and payment information.

4. Make Changes to the Menu

- **Purpose**: Simplifies menu management, enabling administrators to update offerings.
- **Functionality**: This feature is subdivided into three options:
    - **Add a Menu Item**: Add new dishes or drinks with details such as name, category, and price.
    - **Remove a Menu Item**: Delete outdated or unavailable items.
    - **Change Details of a Menu Item**: Update item details like name, category, or price.
- Additionally, the current menu is displayed for easy reference before making any changes.

5. View Sales Report

- **Purpose**: Provides a monthly breakdown of total sales.
- **Functionality**: Generates a report summarizing total revenue for each month, helping administrators identify trends and make informed business decisions.

Customer Functions

1. Customer Registration and Login

- **Purpose**: Facilitates customer onboarding and order placement.
- **Functionality**:
  - New customers can register by providing their name, phone number, and email.
  - Returning customers can log in using their unique customer ID.

2. Place Orders

- **Purpose**: Streamlines the ordering process for customers.
- **Functionality**:
  - Customers can browse the menu, select items, specify quantities, and complete payment via UPI.
  - The system records each order, linking it to the customer's ID for future reference.

3. **View Menu**

- **Purpose**: Allows customers to explore the cafe's offerings.
- **Functionality**: Displays the current menu, complete with item IDs, names, categories, and prices.

Benefits of the Cafe Management System

1. Time-Saving

The system automates many time-consuming manual tasks, such as registering customers, calculating order totals, and generating sales reports. This frees up staff to focus on customer service.

2. Improved Accuracy

By reducing manual input, the system minimizes the risk of human errors in tasks like order processing, data entry, and financial reporting.

3. Enhanced Customer Experience

Customers benefit from a seamless ordering process, access to a well-organized menu, and a quick registration process, which together enhance their overall experience at the cafe.

4. Actionable Insights

The sales report feature provides administrators with valuable insights into sales trends, helping them make informed decisions regarding inventory, staffing, and marketing strategies.

5. Scalability

As the cafe grows, the system can easily accommodate new menu items, additional customers, and increased order volumes, making it a scalable solution for expanding businesses.

SOURCE CODE

```
import mysql.connector
from datetime import datetime
while True:
    try:
        connection = mysql.connector.connect (
            host="localhost",
            database="cafe_central",
            user="root",
            passwd="root")
        if connection.is_connected():
            print("Connected to MySQL database")
    except mysql.connector.Error as e:
        print("Error while connecting to MySQL", e)
        break

    print("\nWelcome to Cafe Management System")
    role = input("Login as (admin/customer) or type 'exit' to quit: ").strip().lower()

    if role == 'admin':
        password = input("Enter admin password: ")
        if password == "Admin442":
            print("Welcome, Admin!")
            while True:
                print("\nAdmin Menu")
                print("1. View all customers")
```

```
            print("2. View all orders")
            print("3. Make changes to the menu")
            print("4. View sales report")
            print("5. Logout")
            choice = input("Select an option: ")

            if choice == '1':
                cursor = connection.cursor(dictionary=True)
                cursor.execute("SELECT * FROM customers")
                customers = cursor.fetchall()
                cursor.close()

                print("\n--- Customers ---")
                for customer in customers:
                    print(f"ID: {customer['id']}, Name: {customer['name']}, Email: {customer['email']}, Phone:
{customer['phone']}")
                print("----------------")

            elif choice == '2':
                cursor = connection.cursor(dictionary=True)
                cursor.execute("SELECT * FROM orders")
                orders = cursor.fetchall()
                cursor.close()

                print("\n--- Orders ---")
                for order in orders:
                    print(f"Order ID: {order['id']}, Customer ID: {order['customer_id']}, Date:
{order['date']}, UPI: {order['upi_number']}")
                print("----------------")
```

```
        elif choice == '3':
            while True:
                # Display the current menu
                print("\n--- Current Menu ---")
                cursor = connection.cursor(dictionary=True)
                cursor.execute("SELECT * FROM menu")
                menu_items = cursor.fetchall()
                cursor.close()

                for item in menu_items:
                    print(f"{item['id']}. {item['category']} - {item['name']} - ${item['price']}")
                print("----------------------\n")

                # Show menu modification options
                print("Menu Modification Options:")
                print("1. Add a menu item")
                print("2. Remove a menu item")
                print("3. Change details of a menu item")
                print("4. Go back to Admin Menu")

                submenu_choice = input("Select an option: ")

                if submenu_choice == '1':
                    name = input("Enter item name: ")
                    category = input("Enter category: ")
                    price = float(input("Enter price: "))

                    cursor = connection.cursor()
                    cursor.execute("INSERT INTO menu (name, category, price) VALUES (%s, %s,
%s)", (name, category, price))
```

```
                connection.commit()
                cursor.close()
                print("Menu item added successfully!")

            elif submenu_choice == '2':
                try:
                    item_id = int(input("Enter the ID of the item to remove: "))
                    cursor = connection.cursor()
                    cursor.execute("DELETE FROM menu WHERE id = %s", (item_id,))
                    connection.commit()
                    cursor.close()
                    print("Menu item removed successfully!")
                except ValueError:
                    print("Invalid ID. Please enter a number.")

            elif submenu_choice == '3':
                try:
                    item_id = int(input("Enter the ID of the item to modify: "))
                    cursor = connection.cursor(dictionary=True)
                    cursor.execute("SELECT * FROM menu WHERE id = %s", (item_id,))
                    item = cursor.fetchone()
                    cursor.close()

                    if item:
                        print(f"Current Details: {item['name']} - {item['category']} -
${item['price']}")
                        new_name = input(f"Enter new name (leave blank to keep
'{item['name']}'): ") or item['name']
                        new_category = input(f"Enter new category (leave blank to keep
'{item['category']}'): ") or item['category']
```

```
                        new_price = input(f"Enter new price (leave blank to keep
'${item['price']}'): ")
                        new_price = float(new_price) if new_price else item['price']

                        cursor = connection.cursor()
                        cursor.execute(
                            "UPDATE menu SET name = %s, category = %s, price = %s WHERE id =
%s",
                            (new_name, new_category, new_price, item_id)
                        )
                        connection.commit()
                        cursor.close()
                        print("Menu item updated successfully!")
                    else:
                        print("Item not found.")
                except ValueError:
                    print("Invalid ID. Please enter a number.")

            elif submenu_choice == '4':
                print("Returning to Admin Menu.")
                break
            else:
                print("Invalid option. Please try again.")

        elif choice == '4':
            cursor = connection.cursor(dictionary=True)
            cursor.execute("""
                SELECT
                    MONTH(o.date) AS month,
                    SUM(oi.total_price) AS total_sales
```

```
                FROM
                    order_items oi
                JOIN
                    orders o ON oi.order_id = o.id
                GROUP BY
                    MONTH(o.date)
            """)
            sales_report = cursor.fetchall()
            cursor.close()

            print("\n--- Monthly Sales Report ---")
            for report in sales_report:
                print(f"Month: {report['month']}, Total Sales: ${report['total_sales']}")
            print("----------------------------\n")

        elif choice == '5':
            print("Logging out of Admin menu.")
            break
        else:
            print("Invalid option")

elif role == 'customer':
    first_time = input("Is this your first time? (yes/no): ").strip().lower()
    if first_time == 'yes':
        name = input("Enter your name: ")
        phone = input("Enter your phone number: ")
        email = input("Enter your email: ")
        cursor = connection.cursor()
        cursor.execute("INSERT INTO customers (name, phone, email) VALUES (%s, %s, %s)",
(name, phone, email))
```

```
    connection.commit()
    customer_id = cursor.lastrowid
    cursor.close()
    print(f"Registration successful! Your customer ID is {customer_id}")

customer_id = input("Enter your customer ID: ")
cursor = connection.cursor(dictionary=True)
cursor.execute("SELECT * FROM customers WHERE id = %s", (customer_id,))
customer = cursor.fetchone()
cursor.close()

if customer:
    print(f"Welcome back, {customer['name']}!")
    while True:
        # Display menu
        print("\nCustomer Menu")
        print("1. View Menu")
        print("2. Place Order")
        print("3. Logout")
        choice = input("Select an option: ")

        if choice == '1':
            cursor = connection.cursor(dictionary=True)
            cursor.execute("SELECT * FROM menu")
            menu_items = cursor.fetchall()
            cursor.close()

            print("\n--- Menu ---")
            for item in menu_items:
```

```
            print(f"{item['id']}. {item['category']} - {item['name']} - ${item['price']}")
        print("----------------\n")

    elif choice == '2':
        cursor = connection.cursor(dictionary=True)
        cursor.execute("SELECT * FROM menu")
        menu_items = cursor.fetchall()
        cursor.close()

        order_items = []
        while True:
            item_id = input("Enter the item ID to order (or type 'done' to finish): ")
            if item_id.lower() == 'done':
                break
            try:
                item_id = int(item_id)
                quantity = int(input("Enter quantity: "))
                selected_item = next((item for item in menu_items if item['id'] == item_id),
None)
                if selected_item:
                    order_items.append((item_id, quantity, selected_item['price'] * quantity))
                else:
                    print("Invalid item ID.")
            except ValueError:
                print("Please enter valid numbers.")

        if not order_items:
            print("No items ordered.")
        else:
            upi_number = input("Enter your UPI number for payment: ")
```

```
                # Insert order details into the database
                cursor = connection.cursor()
                cursor.execute("INSERT INTO orders (customer_id, date, upi_number) VALUES
(%s, %s, %s)",
                               (customer_id, datetime.now(), upi_number))
                order_id = cursor.lastrowid
                for item_id, quantity, total_price in order_items:
                    cursor.execute("INSERT INTO order_items (order_id, item_id, quantity,
total_price) VALUES (%s, %s, %s, %s)",
                                   (order_id, item_id, quantity, total_price))
                connection.commit()
                cursor.close()
                print(f"Order placed successfully! Your order ID is {order_id}.")
            elif choice == '3':
                print("Logging out of Customer menu.")
                break
            else:
                print("Invalid option.")
        else:
            print("Invalid customer ID")

    elif role == 'exit':
        print("Thank you for using Cafe Management System.")
        break
    else:
        print("Invalid role. Please try again.")
```

DATABASE CREATION CODES

Creating the cafe database

```
CREATE DATABASE cafe_central;
```

Creating the ‘customers’ table

```
CREATE TABLE customers (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    phone VARCHAR(15) NOT NULL,
    email VARCHAR(255) NOT NULL UNIQUE,
    created_at TIMESTAMP DEFAULT
CURRENT_TIMESTAMP);
```

Creating the ‘menu’ table

```
CREATE TABLE menu (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    category VARCHAR(100) NOT NULL,
    price DECIMAL(10, 2) NOT NULL);
```

Creating the ‘orders’ table

```
CREATE TABLE orders (
    id INT AUTO_INCREMENT PRIMARY KEY,
    customer_id INT NOT NULL,
    date DATETIME DEFAULT CURRENT_TIMESTAMP,
    upi_number VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES
customers(id) ON DELETE CASCADE);
```

Creating the ‘order_items’ table

```
CREATE TABLE order_items (
    id INT AUTO_INCREMENT PRIMARY KEY,
    order_id INT NOT NULL,
    item_id INT NOT NULL,
    quantity INT NOT NULL,
    total_price DECIMAL(10, 2) NOT NULL,
    FOREIGN KEY (order_id) REFERENCES orders(id) ON
DELETE CASCADE,
    FOREIGN KEY (item_id) REFERENCES menu(id) ON
DELETE CASCADE);
```

DATABASE TABLES

Using the database and displaying the tables

```
mysql> USE cafe_central;
Database changed
mysql> SHOW TABLES;
+------------------------+
| Tables_in_cafe_central |
+------------------------+
| customers              |
| menu                   |
| order_items            |
| orders                 |
+------------------------+
4 rows in set (0.05 sec)
```

DATA IN 'CUSTOMERS' TABLE

```
mysql> select * from customers;
+----+---------------+------------+--------------------+---------------------+
| id | name          | phone      | email              | created_at          |
+----+---------------+------------+--------------------+---------------------+
|  1 | David Miller  | 6543210987 | david@google.com   | 2025-01-15 21:30:28 |
|  2 | Emma Wilson   | 5432109876 | emma@hotmail.com   | 2025-01-15 21:30:28 |
|  3 | Franklin Harris | 4321098765 | franklin@google.com | 2025-01-15 21:30:28 |
|  4 | Grace Lee     | 3210987654 | grace@yahoo.com    | 2025-01-15 21:30:28 |
|  5 | Helen Adams   | 2109876543 | helen@egoogle.com  | 2025-01-15 21:30:28 |
+----+---------------+------------+--------------------+---------------------+
5 rows in set (0.00 sec)
```

DATA IN 'MENU' TABLE

```
mysql> select * from menu;
+----+-----------+-----------+-------+
| id | name      | category  | price |
+----+-----------+-----------+-------+
|  1 | Mocha     | Beverage  |  4.25 |
|  2 | Croissant | Snack     |  3.50 |
|  3 | Brownie   | Dessert   |  4.00 |
|  4 | Iced Tea  | Beverage  |  3.00 |
|  5 | Pancakes  | Breakfast |  6.00 |
+----+-----------+-----------+-------+
5 rows in set (0.01 sec)
```

DATA IN ‘ORDERS’ TABLE

```
mysql> select * from orders;
+----+-------------+---------------------+------------+
| id | customer_id | date                | upi_number |
+----+-------------+---------------------+------------+
|  1 |           5 | 2025-01-16 15:49:19 | 9467135746 |
|  2 |           1 | 2025-01-16 15:50:17 | 7648563149 |
|  3 |           3 | 2025-01-16 15:51:22 | 6475981348 |
|  4 |           2 | 2025-01-16 15:52:03 | 6748951223 |
|  5 |           4 | 2025-01-16 15:53:10 | 7846821674 |
+----+-------------+---------------------+------------+
5 rows in set (0.00 sec)
```

DATA IN ‘ORDER_ITEMS’ TABLE

```
mysql> select * from order_items;
+----+----------+---------+----------+-------------+
| id | order_id | item_id | quantity | total_price |
+----+----------+---------+----------+-------------+
|  1 |        1 |       2 |        1 |        3.50 |
|  2 |        2 |       5 |        1 |        6.00 |
|  3 |        3 |       2 |        1 |        3.50 |
|  4 |        4 |       4 |        1 |        3.00 |
|  5 |        5 |       2 |        1 |        3.50 |
+----+----------+---------+----------+-------------+
5 rows in set (0.00 sec)
```

OUTPUTS

MAIN MENU

```
Welcome to Cafe Management System
Login as (admin/customer) or type 'exit' to quit: admin
Enter admin password: Admin442
Welcome, Admin!
```

ADMIN MENU

```
Admin Menu
1. View all customers
2. View all orders
3. Make changes to the menu
4. View sales report
5. Logout
Select an option:
```

VIEWING ALL CUSTOMERS

```
Admin Menu
1. View all customers
2. View all orders
3. Make changes to the menu
4. View sales report
5. Logout
Select an option: 1

--- Customers ---
ID: 1, Name: David Miller, Email: david@google.com, Phone: 6543210987
ID: 2, Name: Emma Wilson, Email: emma@hotmail.com, Phone: 5432109876
ID: 3, Name: Franklin Harris, Email: franklin@google.com, Phone: 4321098765
ID: 4, Name: Grace Lee, Email: grace@yahoo.com, Phone: 3210987654
ID: 5, Name: Helen Adams, Email: helen@egoogle.com, Phone: 2109876543
----------------
```

VIEWING ALL ORDERS

```
Admin Menu
1. View all customers
2. View all orders
3. Make changes to the menu
4. View sales report
5. Logout
Select an option: 2

--- Orders ---
Order ID: 1, Customer ID: 5, Date: 2025-01-16 15:49:19, UPI: 9467135746
Order ID: 2, Customer ID: 1, Date: 2025-01-16 15:50:17, UPI: 7648563149
Order ID: 3, Customer ID: 3, Date: 2025-01-16 15:51:22, UPI: 6475981348
Order ID: 4, Customer ID: 2, Date: 2025-01-16 15:52:03, UPI: 6748951223
Order ID: 5, Customer ID: 4, Date: 2025-01-16 15:53:10, UPI: 7846821674
----------------
```

MAKING CHANGES TO THE MENU

```
Admin Menu
1. View all customers
2. View all orders
3. Make changes to the menu
4. View sales report
5. Logout
Select an option: 3

--- Current Menu ---
1. Beverage - Mocha - $4.25
2. Snack - Croissant - $3.50
3. Dessert - Brownie - $4.00
4. Beverage - Iced Tea - $3.00
5. Breakfast - Pancakes - $6.00
---------------------

Menu Modification Options:
1. Add a menu item
2. Remove a menu item
3. Change details of a menu item
4. Go back to Admin Menu
Select an option:
```

ADDING A MENU ITEM

```
Admin Menu
1. View all customers
2. View all orders
3. Make changes to the menu
4. View sales report
5. Logout
Select an option: 3

--- Current Menu ---
1. Beverage - Mocha - $4.25
2. Snack - Croissant - $3.50
3. Dessert - Brownie - $4.00
4. Beverage - Iced Tea - $3.00
5. Breakfast - Pancakes - $6.00
----------------------

Menu Modification Options:
1. Add a menu item
2. Remove a menu item
3. Change details of a menu item
4. Go back to Admin Menu
Select an option: 1
Enter item name: Coffee
Enter category: Beverage
Enter price: 3.75
Menu item added successfully!

--- Current Menu ---
1. Beverage - Mocha - $4.25
2. Snack - Croissant - $3.50
3. Dessert - Brownie - $4.00
4. Beverage - Iced Tea - $3.00
5. Breakfast - Pancakes - $6.00
6. Beverage - Coffee - $3.75
----------------------
```

REMOVING A MENU ITEM

```
Menu Modification Options:
1. Add a menu item
2. Remove a menu item
3. Change details of a menu item
4. Go back to Admin Menu
Select an option: 2
Enter the ID of the item to remove: 6
Menu item removed successfully!

--- Current Menu ---
1. Beverage - Mocha - $4.25
2. Snack - Croissant - $3.50
3. Dessert - Brownie - $4.00
4. Beverage - Iced Tea - $3.00
5. Breakfast - Pancakes - $6.00
----------------------
```

CHANGING THE DETAILS OF A MENU ITEM

(For e.g., changing the price of an item)

```
Menu Modification Options:
1. Add a menu item
2. Remove a menu item
3. Change details of a menu item
4. Go back to Admin Menu
Select an option: 3
Enter the ID of the item to modify: 5
Current Details: Pancakes - Breakfast - $6.00
Enter new name (leave blank to keep 'Pancakes'):
Enter new category (leave blank to keep 'Breakfast'):
Enter new price (leave blank to keep '$6.00'): 10.00
Menu item updated successfully!

--- Current Menu ---
1. Beverage - Mocha - $4.25
2. Snack - Croissant - $3.50
3. Dessert - Brownie - $4.00
4. Beverage - Iced Tea - $3.00
5. Breakfast - Pancakes - $10.00
----------------------
```

VIEWING THE SALES REPORT

```
Admin Menu
1. View all customers
2. View all orders
3. Make changes to the menu
4. View sales report
5. Logout
Select an option: 4

--- Monthly Sales Report ---
Month: 1, Total Sales: $19.50
----------------------------
```

CUSTOMER REGISTRATION

```
Welcome to Cafe Management System
Login as (admin/customer) or type 'exit' to quit: customer
Is this your first time? (yes/no): yes
Enter your name: Mohak Goswami
Enter your phone number: 8789457183
Enter your email: mohakgoswami@gmail.com
Registration successful! Your customer ID is 6
```

CUSTOMER MENU

```
Customer Menu
1. View Menu
2. Place Order
3. Logout
Select an option:
```

VIEWING MENU ITEM

```
Customer Menu
1. View Menu
2. Place Order
3. Logout
Select an option: 1

--- Menu ---
1. Beverage - Mocha - $4.25
2. Snack - Croissant - $3.50
3. Dessert - Brownie - $4.00
4. Beverage - Iced Tea - $3.00
5. Breakfast - Pancakes - $10.00
----------------
```

PLACING AN ORDER

```
Customer Menu
1. View Menu
2. Place Order
3. Logout
Select an option: 1

--- Menu ---
1. Beverage - Mocha - $4.25
2. Snack - Croissant - $3.50
3. Dessert - Brownie - $4.00
4. Beverage - Iced Tea - $3.00
5. Breakfast - Pancakes - $10.00
----------------


Customer Menu
1. View Menu
2. Place Order
3. Logout
Select an option: 2
Enter the item ID to order (or type 'done' to finish): 4
Enter quantity: 1
Enter the item ID to order (or type 'done' to finish): done
Enter your UPI number for payment: 8764527961
Order placed successfully! Your order ID is 8.
```

CONCLUSION

The **‘Cafe Management System’** streamlines administrative and customer-facing chores, representing a revolutionary way to modernising a cafe's operations. The system overcomes the drawbacks of conventional cafe administration techniques by incorporating strong features like menu updates, real-time sales reporting, and customer and order management. Its reliance on MySQL as the database and Python for programming guarantees a scalable, safe, and smooth framework. When combined, these technologies provide an effective solution that meets the various needs of administrators and clients, increasing operational effectiveness and elevating the general client experience.

To sum up, the Cafe Management System is a complete solution that enables cafes to prosper in a market that is becoming more and more competitive. It is more than just a digital tool. By encouraging openness, effectiveness, and client happiness, the system lays the foundation for sustained growth and success.

REFERENCES

1. **Python Documentation**: Official Python documentation for understanding language constructs and database connectivity. https://docs.python.org.

2. **MySQL Documentation**: Comprehensive reference for MySQL commands and database management. https://dev.mysql.com/doc.

3. **Stack Overflow**: Forum discussions on debugging and optimization techniques for database-driven applications. https://stackoverflow.com.

4. **W3Schools**: Tutorials on SQL and Python for handling database operations. https://w3schools.com.

5. **MySQL Connector Library**: Official library documentation for connecting Python to MySQL. https://pypi.org/project/mysql-connector-python.

6. **Kaggle Datasets**: Reference for structuring tables and adding meaningful sample data. https://kaggle.com.

7. **Cafe Management Case Studies**: Real-world examples of digitizing cafe operations for inspiration.