UNIVERSITY INSTITUTE OF COMPUTING

Subject Name: Database Management System

Subject Code: 24CAP-204

Project: Online Food Ordering System

Submitted by:

Vikita(24BCD10037)

Submitted to: MR. Suraj Parkash

Acknowledgment

We would like to express our sincere gratitude to all those who supported us in the successful completion of our project, *"Online Food Ordering System."*

First and foremost, we extend our heartfelt thanks to Mr. Suraj Parkash, our respected teacher and guide, for his valuable guidance, continuous support, and encouragement throughout the project. His insightful suggestions, expert knowledge, and constructive feedback greatly helped us in understanding the core concepts of Database Management Systems and in shaping this project effectively.

We are also thankful to Chandigarh University (UIC) for providing us with the required resources, facilities, and a motivating learning environment to carry out this work successfully.

This project has been a wonderful learning experience that helped us strengthen our understanding of database concepts such as data modeling, SQL queries, and relational database design. It also enhanced our skills in developing a real-world application that integrates efficient data handling and user-friendly functionality.

Aim

The primary aim of this project is to design, develop, and implement a comprehensive **Database Management System (DBMS)** for an **Online Food Ordering System**. This system is intended to serve as a centralized platform where:

- **Customers** can browse restaurant menus, place orders, and make secure online payments from anywhere at any time.
- **Restaurants** can efficiently manage their menus, track incoming orders, and update order statuses in real-time.
- **Administrators** can monitor system activity, maintain accurate records, and ensure data consistency across the platform.

The project focuses on achieving **efficiency, accuracy, and automation** in the food ordering and delivery workflow. By leveraging relational database concepts and SQL, the system ensures **data integrity, minimal redundancy, and fast retrieval**, thereby improving both operational efficiency and the overall user experience.

Objectives

1. **Relational Database Design**

- To create a robust **relational database schema** that accurately represents entities such as Customers, Restaurants, MenuItems, Orders, Payments, and their relationships.
- Ensure **data integrity** through primary and foreign key constraints.

2. **Automation of Food Ordering and Billing**

- To digitalize the traditional food ordering process and automate tasks such as **order placement, bill calculation, and payment tracking**, reducing manual errors.

3. **Data Normalization for Consistency**

To **normalize the database up to Third Normal Form (3NF)**, eliminating redundancy and maintaining consistency in the stored data.

4. **SQL Query Implementation**

- To practice **SQL queries** for creating tables, inserting and updating data, retrieving information, and generating reports.
- Implement complex queries using **JOIN, GROUP BY, HAVING, and VIEW** to retrieve meaningful insights from the database.

5. **Enhanced User Experience**

- To design a database that supports **quick and accurate data retrieval**, ensuring that customers receive up-to-date menu information and order status.
- Provide a framework for restaurants and administrators to make informed decisions efficiently.

6. **Scalability and Maintainability**

- To design the system in such a way that it can easily **scale** with additional restaurants, menu items, or customers, and can be **maintained** efficiently without affecting data consistency.

7. **Learning and Skill Development**

- To gain **hands-on experience** in database design, ER diagram creation, data normalization, and SQL-based data manipulation.
- To develop **problem-solving, analytical, and logical reasoning skills** through the design and implementation of a real-world application.

Tools and Technologies Used

- **Database:** MySQL – Stores and manages all customer, restaurant, order, and payment data efficiently.
- **Backend:** SQL / PL/SQL – Used for creating tables, inserting data, and writing queries and procedures.
- Front-end **(Optional):** HTML, CSS, PHP, Python Flask – Builds user interface for customers and admins.

- **ER Diagram Tool:** Draw.io / Lucid chart – Designs database structure and relationships visually.
- **Operating System:** Windows 10 / Linux – Supports development and database deployment.
- **Software Requirements:** MySQL Workbench, VS Code, SQL Plus – Tools for writing, testing, and managing SQL queries and project code

System Requirements

1. **Hardware Requirements:**

- **Processor:** Intel Core i3 or higher for smooth execution of database operations and web interface.
- **RAM:** Minimum 4 GB to handle queries, multiple applications, and server processes efficiently.
- **Hard Disk:** Minimum 500 GB to store database files, backups, and project-related resources.

2. **Software Requirements:**

- **Database Server:** MySQL Database Server to manage and store all project data securely.
- **SQL Client Tool:** MySQL Workbench or Oracle SQL Developer for designing, querying, and managing the database.
- **Operating System:** Compatible with Windows, macOS, or Linux, ensuring flexibility in development and deployment.
- **Optional Tools:** VS Code or any code editor for writing scripts and frontend code, and web browsers to test the application interface.

Algorithm / Logic

1.Customer Registration and Login:

- The customer creates an account by providing details like name, email, phone number, and address.
- The system verifies credentials during login to ensure secure access.

2.Display Menu Items:

- The system fetches menu items from different restaurants and displays them with details such as item name, price, and restaurant.
- Customers can browse through categories or search for specific items.

3.Placing an Order:

- The customer selects desired food items and specifies quantity.
- The system calculates the **total amount** based on selected items and quantity.
- An **Order ID** is generated for tracking purposes.

4.Payment Processing:

- The customer enters payment details and chooses a payment mode (Cash, Card, UPI).
- Payment information is securely stored in the database with a status (Pending, Successful, Failed).

5.Order Status Update:

- The system updates the order status through different stages: **Pending → Preparing → Out for Delivery → Delivered**.
- Customers and restaurant staff can track the order in real-time.

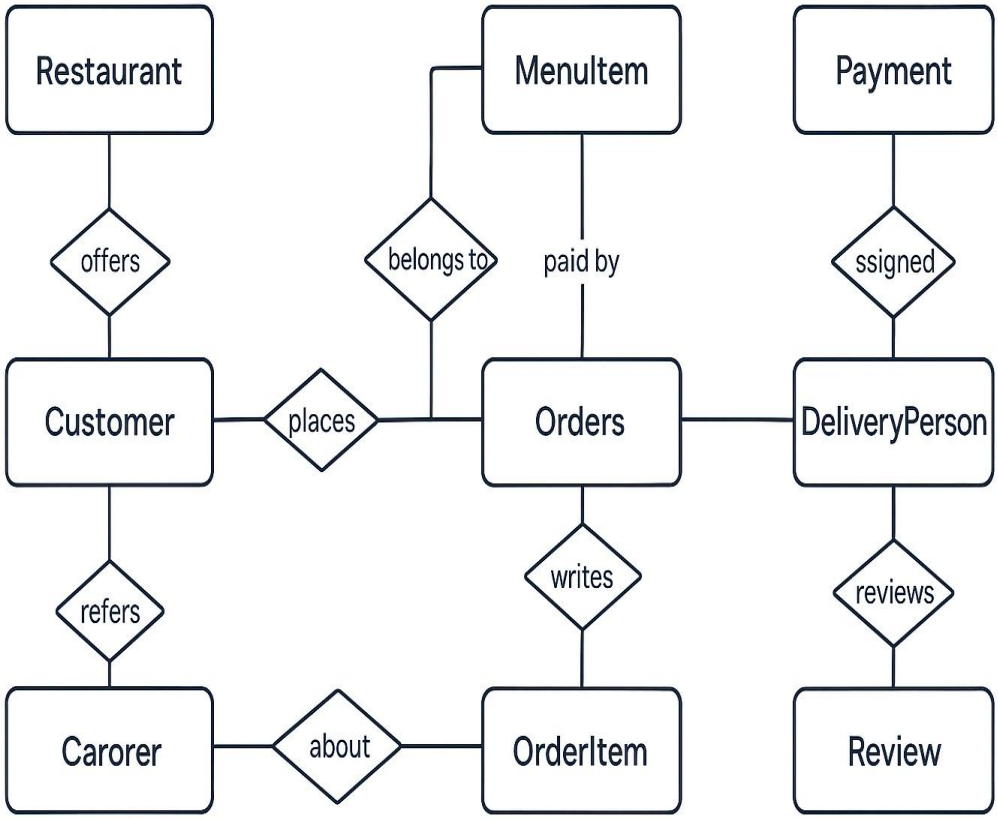
6.Admin Management:

- The admin can **view, add, update, or delete** records related to customers, restaurants, menu items, orders, and payments.
- SQL queries ensure **data integrity and efficient management** of the entire system.

7.Optional Enhancements:

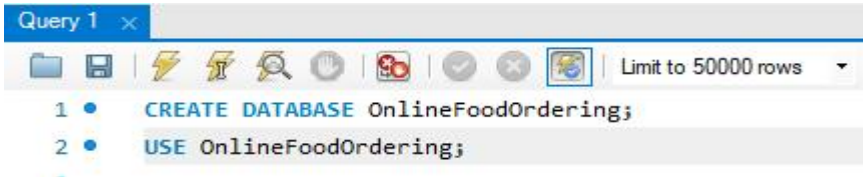
- The system can generate **reports** on total sales, pending orders, or revenue per restaurant.
- Notifications can be sent to customers when the order status changes.

ER Diagram



Code Overview and Implementation

Phase 1: Database Creation



Phase 2: Table Creation

```sql
CREATE TABLE Customer (
  CustomerID INT PRIMARY KEY AUTO_INCREMENT,
  Name VARCHAR(50),
  Email VARCHAR(50) UNIQUE,
  Phone VARCHAR(15),
  Address VARCHAR(100)
);

CREATE TABLE Restaurant (
  RestaurantID INT PRIMARY KEY AUTO_INCREMENT,
  Name VARCHAR(50),
  Location VARCHAR(100)
);

CREATE TABLE MenuItem (
  ItemID INT PRIMARY KEY AUTO_INCREMENT,
  ItemName VARCHAR(50),
  Price DECIMAL(8,2),
  RestaurantID INT,
  FOREIGN KEY (RestaurantID) REFERENCES Restaurant(RestaurantID)
);

CREATE TABLE Orders (
  OrderID INT PRIMARY KEY AUTO_INCREMENT,
  CustomerID INT,
  OrderDate DATE,
  TotalAmount DECIMAL(10,2),
  Status VARCHAR(20),
  FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)
);

CREATE TABLE OrderDetails (
  OrderDetailID INT PRIMARY KEY AUTO_INCREMENT,
  OrderID INT,
  ItemID INT,
  Quantity INT CHECK (Quantity > 0),
  FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
  FOREIGN KEY (ItemID) REFERENCES MenuItem(ItemID)
);

CREATE TABLE Payment (
  PaymentID INT PRIMARY KEY AUTO_INCREMENT,
  OrderID INT,
  Amount DECIMAL(10,2),
  Mode VARCHAR(20) CHECK (Mode IN ('Cash', 'Card', 'UPI')),
  PaymentStatus VARCHAR(20),
  FOREIGN KEY (OrderID) REFERENCES Orders(OrderID)
);
```

Phase 3: Data Insertion

```sql
INSERT INTO Customer (Name, Email, Phone, Address) VALUES
('Shivansh Sharma', 'Shivansh@gmail.com', '9876543210', 'Delhi'),
('Vikita Kumari', 'Vikita@gmail.com', '9998887777', 'Mumbai');

INSERT INTO Restaurant (Name, Location) VALUES
('Pizza Palace', 'Delhi'),
('Burger Hub', 'Mumbai');

INSERT INTO MenuItem (ItemName, Price, RestaurantID) VALUES
('Cheese Pizza', 250, 1),
('Veg Burger', 120, 2),
('Pasta', 180, 1);

INSERT INTO Orders (CustomerID, OrderDate, TotalAmount, Status) VALUES
(1, '2025-10-23', 430, 'Delivered'),
(2, '2025-10-23', 120, 'Pending');

INSERT INTO OrderDetails (OrderID, ItemID, Quantity) VALUES
(1, 1, 1),
(1, 3, 1),
(2, 2, 1);

INSERT INTO Payment (OrderID, Amount, Mode, PaymentStatus) VALUES
(1, 430, 'Card', 'Successful'),
(2, 120, 'UPI', 'Pending');
```

Phase 4: SQL Queries and Output

1 View all customers:

```sql
SELECT * FROM Customer;
```

Result Grid | Filter Rows: | Edit: | Export/I

CustomerID	Name	Email	Phone	Address
1	Shivansh Sharma	Shivansh@gmail.com	9876543210	Delhi
2	Vikita Kumari	Vikita@gmail.com	9998887777	Mumbai
NULL	NULL	NULL	NULL	NULL

2 Show all orders with customer names:

```
80 •  SELECT O.OrderID, C.Name AS CustomerName, O.OrderDate, O.TotalAmount, O.Status
81    FROM Orders O
82    JOIN Customer C ON O.CustomerID = C.CustomerID;
83
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

OrderID	CustomerName	OrderDate	TotalAmount	Status
1	Shivansh Sharma	2025-10-23	430.00	Delivered
2	Vikita Kumari	2025-10-23	120.00	Pending

3 Display items ordered by a specific customer:

```
83 •  SELECT C.Name, M.ItemName, OD.Quantity
84    FROM OrderDetails OD
85    JOIN Orders O ON OD.OrderID = O.OrderID
86    JOIN Customer C ON O.CustomerID = C.CustomerID
87    JOIN MenuItem M ON OD.ItemID = M.ItemID
88    WHERE C.Name = 'Shivansh Sharma';
89
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Name	ItemName	Quantity
Shivansh Sharma	Cheese Pizza	1
Shivansh Sharma	Pasta	1

4 Find total revenue of all delivered orders:

```
89 •  SELECT SUM(TotalAmount) AS Total_Revenue
90    FROM Orders
91    WHERE Status = 'Delivered';
92
```

Result Grid | Filter Rows: | Export:

Total_Revenue
430.00

5 Create a view for pending payments:

```
92 •  CREATE VIEW PendingPayments AS
93    SELECT C.Name, O.OrderID, P.Amount, P.PaymentStatus
94    FROM Payment P
95    JOIN Orders O ON P.OrderID = O.OrderID
96    JOIN Customer C ON O.CustomerID = C.CustomerID
97    WHERE P.PaymentStatus = 'Pending';
```

Output

Action Output

#	Time	Action	Message
21	22:17:01	describe table Customer	1 row(s) returned
22	22:17:27	describe table Orders	1 row(s) returned
23	22:18:08	SELECT * FROM Customer LIMIT 0, 50000	2 row(s) returned
24	22:18:21	SELECT O.OrderID, C.Name AS CustomerName, O.OrderDate, O.TotalAmount, O.Status FR...	2 row(s) returned
25	22:19:03	SELECT C.Name, M.ItemName, OD.Quantity FROM OrderDetails OD JOIN Orders O ON OD...	2 row(s) returned
26	22:19:40	SELECT SUM(TotalAmount) AS Total_Revenue FROM Orders WHERE Status = 'Delivered' ...	1 row(s) returned
27	22:20:08	CREATE VIEW PendingPayments AS SELECT C.Name, O.OrderID, P.Amount, P.PaymentSt...	0 row(s) affected

Conclusion

The project successfully demonstrates the design and implementation of a **fully functional Database Management System (DBMS)** for an **Online Food Ordering System**, highlighting the practical application of database concepts in real-world scenarios. Key achievements include:

- **Efficient Data Management:** The system stores and organizes information about customers, restaurants, menu items, orders, and payments, allowing quick and accurate retrieval of data.
- **Automation of Operations:** The workflow for order placement, billing, and payment tracking has been automated, reducing human intervention and errors.
- **Data Integrity and Consistency:** Through **primary and foreign keys, constraints, and normalization**, the system ensures that data is reliable, non-redundant, and easy to maintain.
- **Enhanced User and Admin Experience:** Customers benefit from a streamlined ordering experience, while restaurants and administrators can easily manage operations, track orders, and monitor revenue.
- **Practical SQL Application:** The project provided **hands-on experience with SQL operations**, including DDL, DML, DQL, JOINs, and Views, reinforcing theoretical knowledge with practical application.
- **Future Scalability:** The database design supports potential future enhancements, such as adding more restaurants, payment modes, customer loyalty programs, or integrating analytics for business insights.

Overall, the project bridges **theoretical knowledge of DBMS** with **practical application**, preparing students to handle real-world database challenges effectively. It serves as a foundation for advanced projects in **web-based applications, e-commerce, and data-driven platforms**.

References

- *Database System Concepts* — Abraham Silberschatz, Henry F. Korth.
- TutorialsPoint, W3Schools — SQL Query References.
- MySQL Documentation.
- Draw.io / Lucidchart — ER Diagram Tools.
- **Stack Overflow**: Community-driven Q&A for solving real-world SQL and database design problems.