Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology
(Deemed to be University Estd. U/s 3 of UGC Act, 1956)
School of Computing
B.Tech. – Computer Science an Engineering
VTR USE2821–(CBCS)

Academic Year:2025– 2026

SDG-4:Quality Education

Course Code : 10211CS207

Course Name: Database Management Systems

Slot No : S2L3

DBMS PROJECT REPORT

Title: Online Learning Platform Management System (OLPMS)

Submitted by:

VTU NUMBER	REGISTRATION NUMBER	STUDENT NAME
VTU29692	24UECS1503	GADILLI NAGA SAI
VTU29785	24UECS0922	SIVANESWARI.S
VTU29809	24UECS1158	C.UJWALA
VTU29890	24UECS0427	SANJANA.B
VTU29558	24UECS0216	MEKAPOTHULA SIVA SANKARA VARA PRASAD

Under the guidance of:

Dr V Senthil kumar

INDEX. Pgno.

1. Introduction

In today's digital era, online food delivery platforms have revolutionized the way people order and enjoy their meals. Instead of visiting restaurants physically, customers can conveniently browse menus, place orders, and get food delivered to their doorstep through mobile or web applications. The Online Food Delivery Management System (OFDMS) aims to provide a seamless, efficient, and user-friendly platform for managing restaurant data, customer orders, delivery tracking, and payments.

The system acts as a bridge between customers, restaurants, and delivery partners. It simplifies the ordering process, reduces manual errors, and enhances the overall dining experience. This project demonstrates the use of relational database design and NoSQL integration to manage structured and unstructured data efficiently. The solution ensures real-time updates, secure transactions, and scalable performance for large volumes of orders.

The Online Food Delivery Management System contributes significantly to business automation in the food industry. By integrating database management with web-based ordering and tracking, the system provides transparency, reliability, and convenience. Customers can explore menus, customize their meals, and make digital payments, while restaurants can manage their menus, receive live order notifications, and monitor sales.

2. Problem Statement

The traditional method of ordering food involves either dining at restaurants or placing orders through phone calls, which is often time-consuming and inefficient. Managing customer records, tracking multiple orders, and maintaining delivery information manually can lead to errors and miscommunication. Additionally, restaurants struggle with inconsistent order data, delayed delivery updates, and limited customer engagement.

Customers face issues like inaccurate order status, lack of menu information, and payment delays. Delivery partners, on the other hand, often encounter route management difficulties and lack of real-time order tracking. Restaurants also find it challenging to manage peak-hour traffic and customer feedback without an integrated digital system.

The Online Food Delivery Management System addresses these challenges by offering a centralized platform where customers can place orders online, restaurants can manage menus and orders efficiently, and delivery partners can track and complete deliveries with real-time updates. It ensures accuracy, convenience, and speed for all parties involved in the process.

3. Objectives

- To design a relational database for managing customers, restaurants, food items, and orders.
- To ensure data integrity through the use of primary and foreign keys.
- To provide efficient order tracking, billing, and delivery management.
- To enable data retrieval through optimized SQL queries.
- To integrate MongoDB for storing unstructured data such as food images and customer reviews.
- To ensure scalability and security in handling high transaction volumes.

4. System Requirements

Hardware Requirements

Processor: Intel i5 or higher

RAM: Minimum 8 GB

Hard Disk: 250 GB or more

Software Requirements

Operating System: Windows 10 / Linux

Database: Oracle 12c or higher

Front-End: Java / Web Interface

NoSQL Database: MongoDB 6.0

Tools: Oracle SQL Developer, MongoDB Compass

5. System Analysis and Design

The system identifies the following major entities and relationships:

Customer

Restaurant

Menu_Item

Order

Order_Detail

Delivery_Partner

Payment Each order connects a customer to a restaurant and contains one or more food items. Delivery details and payment status are also linked to each order record.

6. ER Diagram (Conceptual Design)

Entities and Attributes

Customer (Customer_ID, Name, Email, Phone, Address)

Restaurant (Restaurant_ID, Name, Location, Contact, Rating)

Menu_Item (Item_ID, Restaurant_ID, Item_Name, Price, Category)

Order (Order_ID, Customer_ID, Restaurant_ID, Order_Date, Total_Amount, Status)

Order_Detail (Order_ID, Item_ID, Quantity)

Delivery_Partner (Partner_ID, Name, Phone, Vehicle_No)

Payment (Payment_ID, Order_ID, Mode, Amount, Payment_Status)

Relationships

A Customer can place multiple Orders.

Each Order is linked to one Restaurant.

Each Order includes multiple Menu Items (many-to-many via Order_Detail).

Each Order is assigned to one Delivery Partner.

Each Order has one Payment record.

7. Schema Design (Oracle)

```
CREATE TABLE Customer (
  Customer_ID NUMBER PRIMARY KEY,
  Name VARCHAR2(50),
  Email VARCHAR2(50),
  Phone VARCHAR2(15),
  Address VARCHAR2(100)
);

CREATE TABLE Restaurant (
  Restaurant_ID NUMBER PRIMARY KEY,
  Name VARCHAR2(50),
  Location VARCHAR2(50),
  Contact VARCHAR2(15),
  Rating NUMBER(2,1)
);

CREATE TABLE Menu_Item (
  Item_ID NUMBER PRIMARY KEY,
  Restaurant_ID NUMBER REFERENCES Restaurant(Restaurant_ID),
```

```
  Item_Name VARCHAR2(50),
  Price NUMBER(6,2),
  Category VARCHAR2(30)
);

CREATE TABLE Order_Table (
  Order_ID NUMBER PRIMARY KEY,
  Customer_ID NUMBER REFERENCES Customer(Customer_ID),
  Restaurant_ID NUMBER REFERENCES Restaurant(Restaurant_ID),
  Order_Date DATE,
  Total_Amount NUMBER(8,2),
  Status VARCHAR2(20)
);

CREATE TABLE Order_Detail (
  Order_ID NUMBER REFERENCES Order_Table(Order_ID),
  Item_ID NUMBER REFERENCES Menu_Item(Item_ID),
  Quantity NUMBER,
  PRIMARY KEY (Order_ID, Item_ID)
);
```

```
CREATE TABLE Delivery_Partner (
  Partner_ID NUMBER PRIMARY KEY,
  Name VARCHAR2(50),
  Phone VARCHAR2(15),
  Vehicle_No VARCHAR2(20)
);

CREATE TABLE Payment (
  Payment_ID NUMBER PRIMARY KEY,
  Order_ID NUMBER REFERENCES Order_Table(Order_ID),
  Mode VARCHAR2(20),
  Amount NUMBER(8,2),
  Payment_Status VARCHAR2(20)
);
```

8. Normalization

The database design follows normalization up to Third Normal Form (3NF):

1NF: Each table has atomic values and a unique primary key.

2NF: All non-key attributes depend on the full primary key.

3NF: There are no transitive dependencies among non-key attributes.

This ensures data consistency, removes redundancy, and maintains referential integrity.

9. Implementation (SQL Queries)

INSERT INTO Customer VALUES (1, ‘Ravi Kumar’, ‘ravi@gmail.com’, ‘9876543210’, ‘Chennai’);

INSERT INTO Restaurant VALUES (101, ‘Spice Hub’, ‘Chennai’, ‘0442456789’, 4.3);

INSERT INTO Menu_Item VALUES (1001, 101, ‘Paneer Butter Masala’, 220, ‘Main Course’);

INSERT INTO Order_Table VALUES (5001, 1, 101, SYSDATE, 440, ‘Delivered’);

INSERT INTO Order_Detail VALUES (5001, 1001, 2);

INSERT INTO Delivery_Partner VALUES (301, ‘Arun’, ‘9789098776’, ‘TN09C2345’);

INSERT INTO Payment VALUES (9001, 5001, ‘UPI’, 440, ‘Completed’);

Sample Query:

SELECT c.Name, r.Name AS Restaurant, o.Total_Amount, o.Status

FROM Customer c

JOIN Order_Table o ON c.Customer_ID = o.Customer_ID

JOIN Restaurant r ON o.Restaurant_ID = r.Restaurant_ID;

10. Input and Output

Sample Input

INSERT INTO Customer VALUES (2, 'Priya Sharma', 'priya@gmail.com', '9823445566', 'Coimbatore');

INSERT INTO Menu_Item VALUES (1002, 101, 'Veg Biryani', 180, 'Main Course');

Expected Output

Customer Name Restaurant Total Amount Status

Ravi Kumar Spice Hub440 Delivered

11. Integration with MongoDB (NoSQL)

MongoDB stores unstructured data such as food images, reviews, and delivery logs.

Database Name: FoodDeliveryDB

Collections and Example Documents:

1. customers

```
{
  "_id": 1,
  "customer_id": "C001",
  "name": "Ravi Kumar",
  "email": "ravi@gmail.com",
  "address": "Chennai",
  "orders": ["O5001", "O5002"]
}
```

2. restaurants

```
{
  “_id”: 1,
  “restaurant_id”: “R101”,
  “name”: “Spice Hub”,
  “menu”: [
    {“item”: “Paneer Butter Masala”, “price”: 220},
    {“item”: “Veg Biryani”, “price”: 180}
```

]

}

3. reviews

{

“_id”: 1,

“order_id”: “O5001”,

“rating”: 5,

“comment”: “Delicious food and fast delivery!”

}

4. delivery_logs

{

“_id”: 1,

“partner_id”: “P301”,

“order_id”: “O5001”,

“status”: “Delivered”,

“timestamp”: “2025-10-26T19:00:00Z”

}

Example Queries

Db.customers.find({}, {name:1, email:1});

Db.reviews.find({rating: {$gte: 4}});

Db.delivery_logs.updateOne({order_id: “O5001”}, {$set: {status: “Delivered”}});

12. Results and Discussion

The Online Food Delivery Management System successfully automates the process of ordering, tracking, and delivering food. It connects customers, restaurants, and delivery partners through a unified platform. The system ensures real-time data synchronization, minimizes manual errors, and improves customer satisfaction.

Testing confirms that the database structure efficiently handles multiple concurrent orders and queries. The integration with MongoDB enhances scalability and supports multimedia storage. Performance evaluations show faster order retrieval and accurate billing. Overall, the system improves operational efficiency, reduces delivery delays, and strengthens customer-restaurant relationships.

13. Conclusion

The Online Food Delivery Management System provides a comprehensive solution for automating the food ordering process. By combining Oracle for structured data and MongoDB for unstructured data, the system achieves both reliability and flexibility. It enhances user convenience, optimizes restaurant operations, and ensures secure transactions. The project demonstrates effective use of database principles, normalization, and real-world implementation of SQL and NoSQL integration.

In conclusion, the system contributes to the digital transformation of the food industry by enabling fast, transparent, and customer-centric services.

14. References

1. Oracle Database Documentation – Oracle Corporation

2. MongoDB Official Documentation

3. Silberschatz, Korth, Sudarshan – Database System Concepts

4. Raghu Ramakrishnan – Database Management Systems

5. TutorialsPoint – SQL Basics and Examples