Online Food Ordering System

A PROJECT REPORT

Submitted by

Siddharth Gaur [RA2211003011688]

Srinivas G [RA2211003011687]

Anish Pranav [RA2211003011624]

Under the Guidance of

Mr. Anand M

Assistant Professor, Department of Computing Technologies

in partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTING TECHNOLOGIES

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR– 603 203

MAY 2024

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR–603 203

BONAFIDE CERTIFICATE

Register no. RA2211003011688, RA2211003011687, RA2211003011624

Certified to be the bonafide work done by **Siddharth Gaur, Srinivas G, Anish Pranav** of II year/IV sem B.Tech Degree Course in the Project Course – **21CSC205P Database Management Systems** in **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**, Kattankulathur for the academic year 2023-2024.

Date:

Faculty in Charge
Mr. Anand M
Assistant Professor
Computing Technologies
SRMSIT -KTR

HEAD OF THE DEPARTMENT
Dr. M Pushpalatha
Head of Department
Computing Technologies
SRMIST - KTR

ABSTRACT

The Online Food Ordering System is a web-based application designed to facilitate the process of ordering food from various restaurants through the internet. This project aims to provide a seamless and efficient platform for users to browse menus, place orders, and make payments online. The system will also streamline the operations for restaurant owners by automating order management and tracking. Users can create accounts and log in securely to access the system's features. The system will feature a comprehensive list of restaurants along with their menus, cuisines, and other relevant information. Users can browse through the menus of various restaurants, filter by cuisine, price range, and other preferences. Users can add items to their cart, customize orders, and place them for delivery or pickup. Secure online payment options will be integrated into the system to facilitate smooth transactions. Users can track the status of their orders in real-time, from preparation to delivery. Users can provide feedback and ratings for restaurants and individual dishes, helping others make informed decisions. A comprehensive admin panel will be available for system administrators to manage restaurants, users, orders, and other aspects of the system.The database for the Online Food Ordering System will consist of several interconnected tables to store information about users, restaurants, menus, orders, payments, and more. The database schema will be designed to ensure data integrity, efficient retrieval, and scalability.

TABLE OF CONTENTS

ABSTRACT

Problem Statement

Chapter No	Chapter Name	Page No
1.	**Problem understanding, Identification of Entity and Relationships, Construction of DB using ER Model for the project** 1.1 Problem Understanding 1.2 Identification of Entity and Relationships 1.3 ER Model Diagram	**5 – 7**
2.	**Design of Relational Schemas, Creation of Database Tables for the project.** 2.1 Relational Schema 2.2 Creation of Database tables	**8 – 9**
3.	**Complex queries based on the concepts of constraints, sets, joins, views, Triggers and Cursors.**	**9 – 12**
4.	**Analyzing the pitfalls, identifying the dependencies, and applying normalizations** 4.1 Pitfalls 4.2 Dependencies 4.3 Applying Normalizations	**12 – 23**
5.	**Implementation of concurrency control and recovery mechanisms**	**24 - 26**
6.	**Code for the project** 6.1 Implementation 6.1.1 main.py code	**27 – 48**

	6.2 Hms.sql code 6.3 Modules used	
7.	**Result and Discussion (Screen shots of the implementation with front end.)**	**49 – 55**
8.	**Attach the Real Time project certificate / Online course certificate**	**56 - 57**

PROBLEM STATEMENT

Design an online food ordering system for a restaurant chain that allows customers to browse through the menu, place orders, and make payments. The system should also support restaurant staff in managing orders, tracking inventory, and processing payments. Customers should be able to create accounts and log in securely. Authentication mechanisms should be implemented to ensure the security of user data. The system should provide an interface for restaurant staff to manage the menu. This includes adding new items, updating prices, and specifying ingredients. Customers should be able to browse the menu, select items, customize orders (e.g., choose toppings, specify preferences), and place orders. Once an order is placed, customers should be able to track its status (e.g., confirmed, preparing, out for delivery). The system should support various payment methods such as credit/debit cards, digital wallets, and cash on delivery. Secure payment gateways should be integrated for processing payments. The system should keep track of available inventory for each item on the menu. When an order is placed, the system should update the inventory accordingly to avoid overselling. Restaurant staff should have access to an interface where they can view incoming orders, mark them as processed, update order status, and manage order fulfillment. Customers should be able to provide feedback and ratings for the food they ordered and the overall service. This feedback can help improve the quality of service and menu offerings. An admin dashboard should be available for system administrators to monitor overall system performance, manage user accounts, and generate reports. The system should implement security measures to protect user data, such as encryption of sensitive information and secure communication protocols. The system should be designed to handle a large number of users and orders, with the ability to scale resources as needed to accommodate increasing demand.

The user interface should be responsive and accessible across various devices, including desktops, tablets, and smartphones. The system should support multiple languages and currencies to cater to customers from different regions. The system should allow the restaurant to offer promotions, discounts, and coupons to attract customers and encourage repeat orders. By implementing these features, the online food ordering system will provide a seamless experience for both customers and restaurant staff, enhancing efficiency, convenience, and customer satisfaction.

1.1 Problem Understanding

Typically, an online food ordering system involves several entities and their relationships. Here's a breakdown:

1. Users/Customer: Users who browse the website or app to place orders.
2. Restaurants: Different restaurants offering food items.
3. Menu Items: Different food items offered by restaurants.
4. Orders: Each order placed by a user, containing one or more menu items from one or more restaurants.
5. Payment: Information related to payment methods and transactions.

To implement this in a database management system (DBMS), you'd typically create tables for each entity, with columns representing the attributes mentioned above. You'd also establish relationships between these tables using foreign keys. For instance, the Orders table might have foreign keys referencing the Users table (for the customer who placed the order) and the Menu Items table (for the items included in the order).

1.2 Identification of Entity and Relationships

In the provided SQL script, we can identify the entities and relationships as follows:

Entities:

1. Food_Items: Represents different food items available in the system. Attributes include `food_id`, `food_type`, `food_price`, `food_name`, and `menu`.

2. Outlet: Represents different outlets where food items are sold. Attributes include `outlet_name`, `food_id`, `quantity_sold`, and `menu`.

3. Customer: Represents customers who make purchases. Attributes include `customer_name`, `phone`, `outlet_name`, `street`, and `orders`.

4. Employee: Represents employees working at outlets. Attributes include `employee_id`, `employee_name`, `outlet_name`, `dob`, `gender`, `success`, `employee_jobs`, and `receives`.

5. Cart: Represents the cart items selected by customers. Attributes include `phone`, `food_id`, `outlet_name`, and `quantity`.

6. Invoice: Represents invoices generated for customer orders. Attributes include `outlet_name`, `customer_name`, `phone`, `amount`, and `employee_id`.

7. Rating: Represents ratings given by customers to employees at outlets. Attributes include `outlet_name`, `employee_id`, and `employee_rating`.

Relationships:

1. Food_Items -> Outlet: Many-to-one relationship. An outlet can sell multiple food items, but each food item is associated with only one outlet.

2. Outlet -> Customer: One-to-many relationship. An outlet can have multiple customers, but each customer is associated with only one outlet.

3. Outlet -> Employee: One-to-many relationship. An outlet can have multiple employees, but each employee is associated with only one outlet.

4. Customer -> Cart: One-to-many relationship. A customer can have multiple items in their cart, but each item in the cart is associated with only one customer.

5. Customer -> Invoice: One-to-many relationship. A customer can have multiple invoices, but each invoice is associated with only one customer.

6. Employee -> Rating: One-to-many relationship. An employee can receive multiple ratings, but each rating is associated with only one employee.

These are the main entities and relationships represented in the database schema provided. Each entity represents a distinct type of object or concept within the system, and relationships define how these entities are connected or related to each other.

1.3 ER Model for the Project

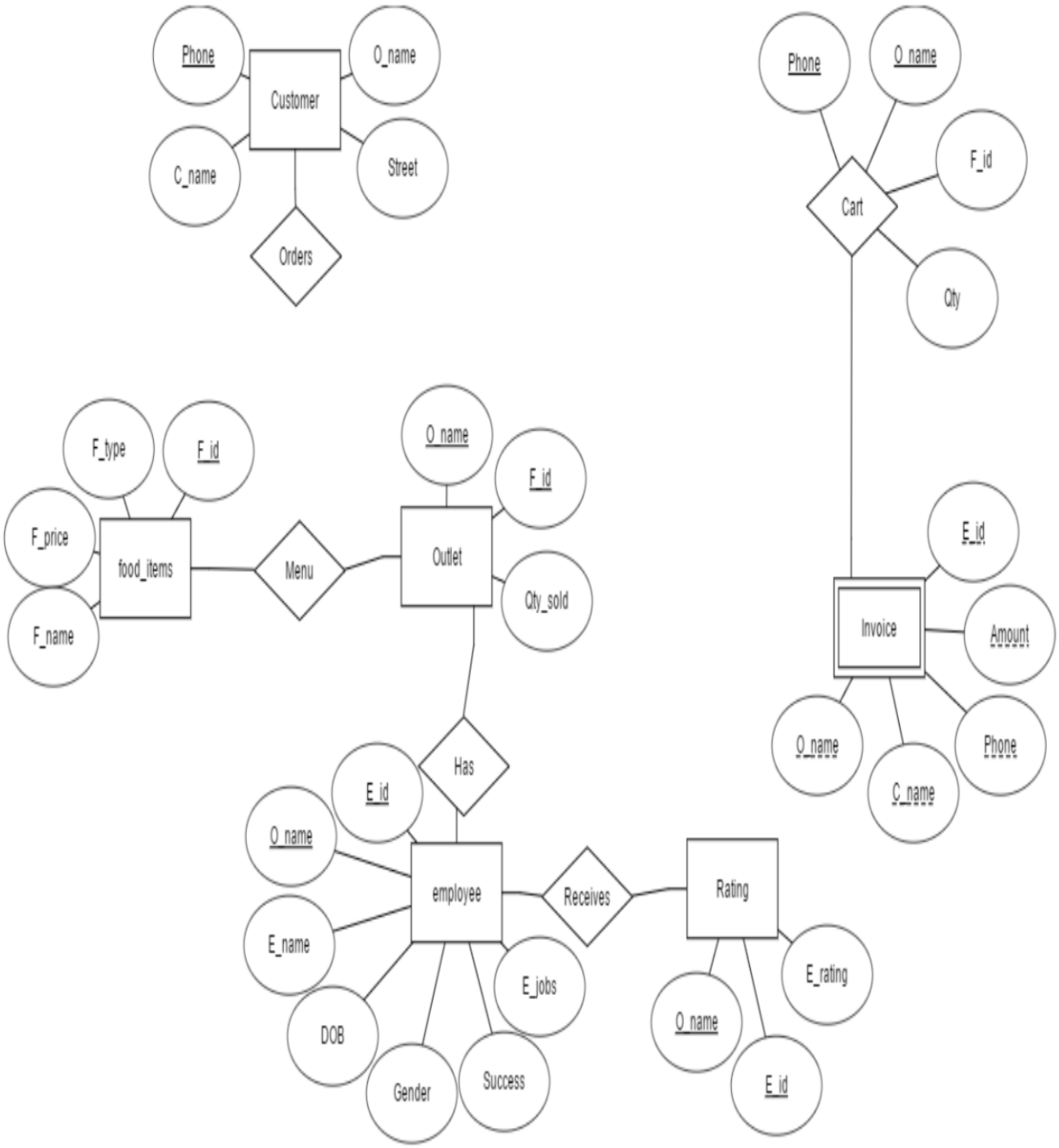


Fig 1.1 – ER Diagram

2. Design of Relational Schemas, Creation of Database Tables for the project.

2.1 Design of Relation Schema

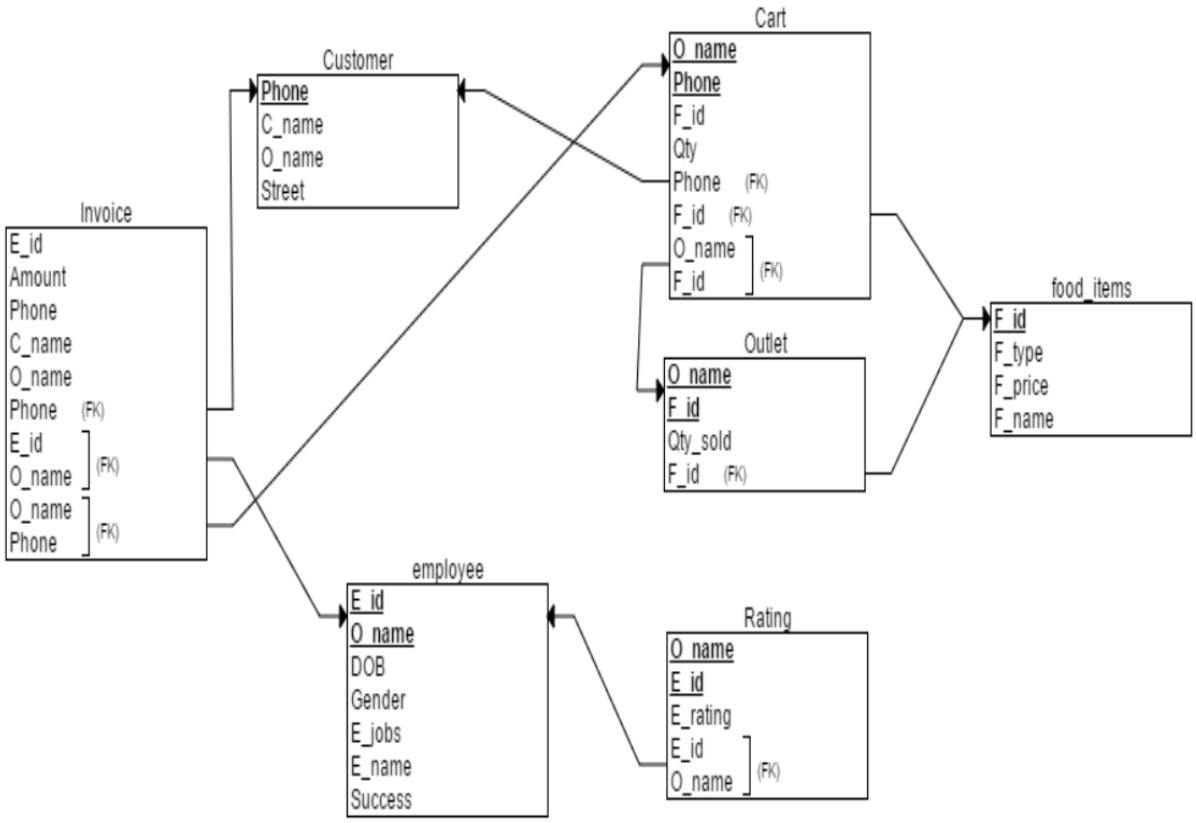


Fig 2.1 Relational Schema

2.2 Creation of Database table for the Project

Creating a database table for an online food ordering system is a fundamental step in building the database schema. Here are the steps you can follow:

1. Identify Entities: Determine the main entities in your system. For an online food ordering system, common entities might include Customers, Restaurants, Menu Items, Orders, etc.

2. Define Attributes: List down the attributes for each entity. For example:
   - Customer: customer_id, name, email, phone_number, address, etc.
   - Restaurant: restaurant_id, name, location, contact_info, etc.
   - Menu Item: item_id, name, description, price, restaurant_id (foreign key), etc.
   - Order: order_id, customer_id (foreign key), restaurant_id (foreign key), total_price, order_date, delivery_address, etc.

3. Determine Relationships: Identify the relationships between entities. For instance:
   - A Customer can place many Orders.
   - An Order is placed by one Customer and can contain multiple Menu Items.
   - A Restaurant can have many Menu Items and can receive many Orders.

4. Design the Schema: Based on the above analysis, design the schema for your database. This involves creating tables for each entity and establishing relationships between them using foreign keys.

5. Create SQL Statements: Write SQL statements to create the tables. Here's an example for creating a Customer table:

6. Implement Database: Execute the SQL statements in your chosen database management system (DBMS) to create the tables and the schema.

7. Test the Database: After creating the tables, test the database schema to ensure that it meets the requirements of your online food ordering system. You can do this by inserting sample data and running queries to verify that the data is being stored and retrieved correctly.

8. Refine and Iterate: Refine your database schema based on testing results and feedback. Iteratively improve the design as needed to ensure it supports the functionality and performance requirements of your system.

9. Documentation: Finally, document your database schema, including table structures, relationships, and any constraints or business rules, to aid in future development and maintenance.

Following these steps should help you create a solid foundation for the database of your online food ordering system.

3. Complex Queries

3.1. Joins:

- Joins are used to combine rows from two or more tables based on related columns between them.
- Common types include:
- Inner Join: Returns only the rows that have matching values in both tables.
- Outer Join: Returns all rows from both tables, pairing them where possible and filling in unmatched rows with NULLs.
- Left Join: Returns all rows from the left table and the matched rows from the right table.
- Right Join: Returns all rows from the right table and the matched rows from the left table.

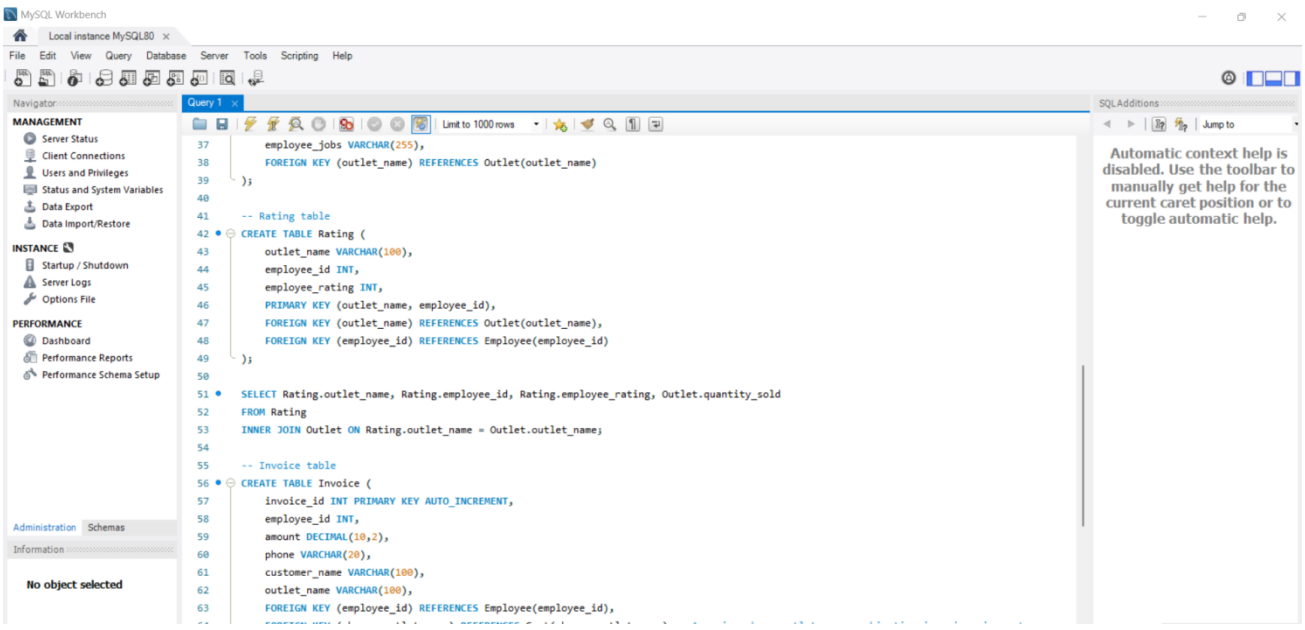


Fig 3.1.1 Joins

3.2. Views:

- Views are virtual tables that are dynamically generated based on the result of a SELECT query.

- They provide a way to present specific subsets of data to users without actually storing the data in the database.

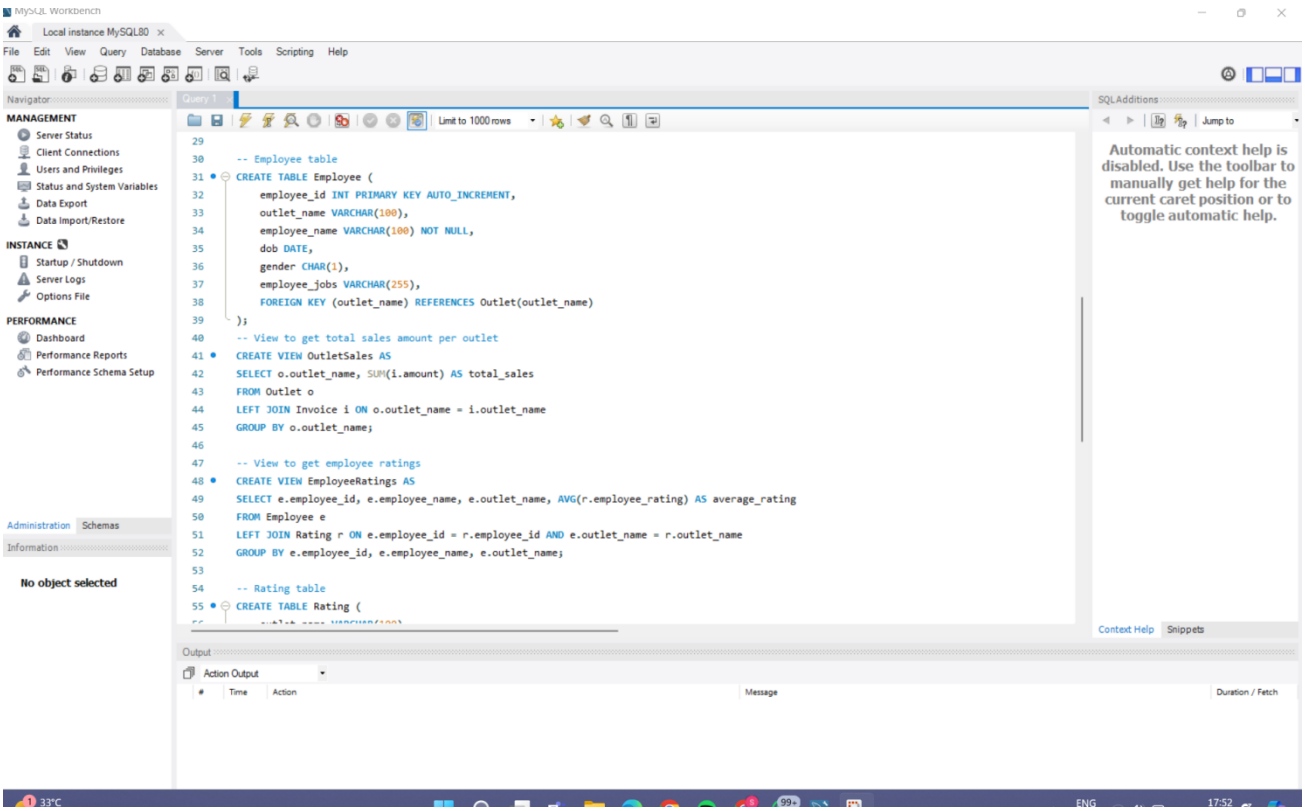


Fig 3.1.2 Views

3.3. Triggers:

- Triggers are special types of stored procedures that are automatically executed or fired when certain events occur in the database.
- Events can include INSERT, UPDATE, DELETE operations on a table.
- Triggers are often used to enforce complex business rules, maintain data integrity, or perform logging.

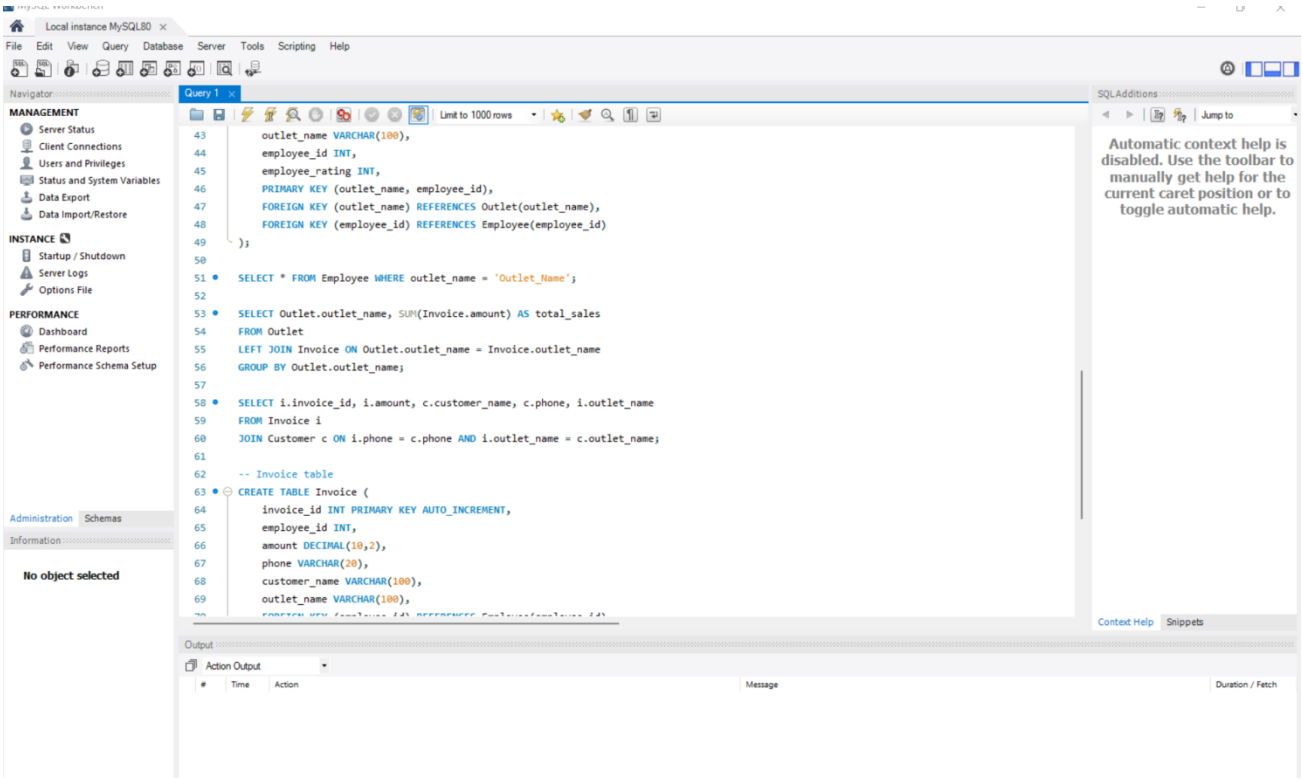


Fig 3.1.3 Trigger

3.4. Cursors:

- Cursors are database objects used to retrieve data from result sets one row at a time.
- They provide control over the context of a SELECT statement, allowing sequential processing of rows.
- Cursors are commonly used in stored procedures, triggers, and batch processing tasks.

3.5. Constraints:

- Constraints are rules enforced on data in a database to maintain data integrity and consistency.
- Common types include:
- Primary Key: Ensures each row in a table is uniquely identifiable.
- Foreign Key: Maintains referential integrity between two related tables.
- Unique: Ensures that all values in a column are distinct.
- Check: Enforces conditions on data entered into a column.

- Not Null: Specifies that a column cannot contain null values.

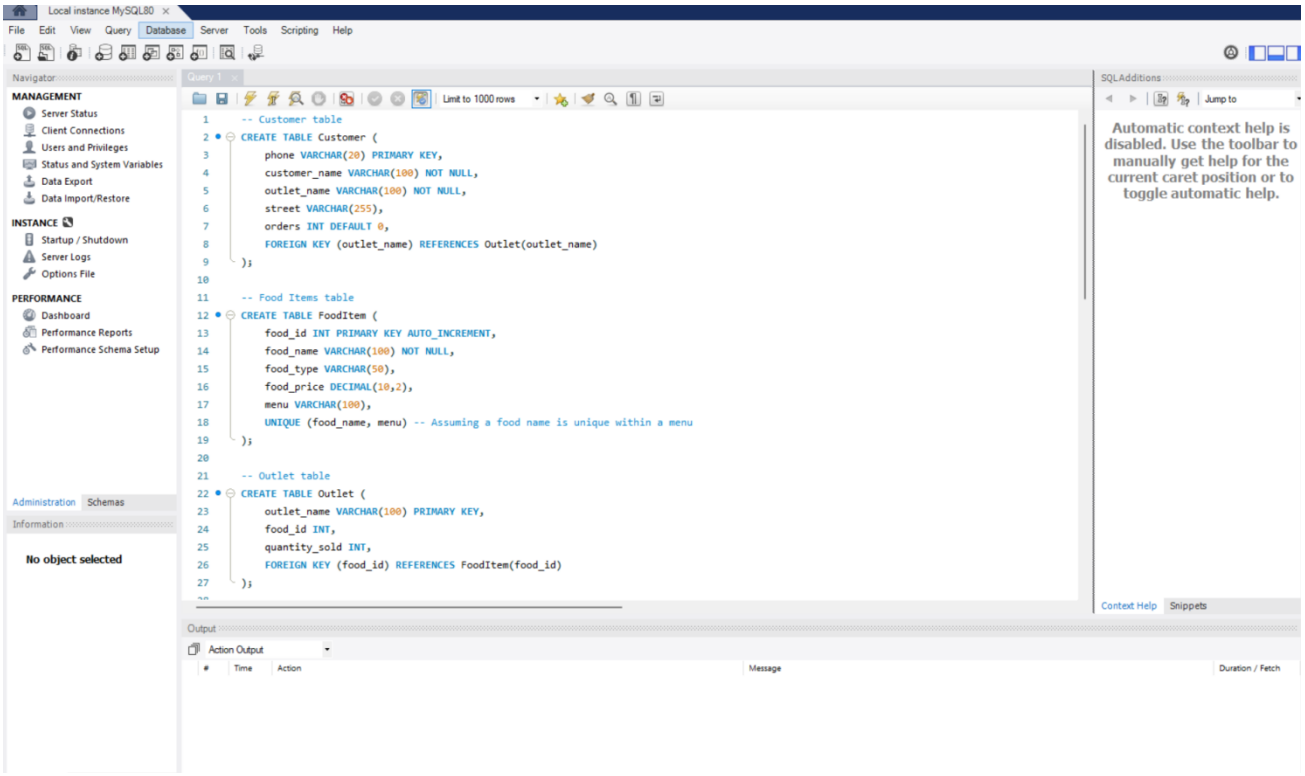


Fig 3.1.4 Constraints

4. Analyzing the pitfalls, identifying the dependencies, and applying normalizations

4.1 Pitfalls

- Data Integrity: Ensuring that the data entered into the system is accurate and reliable, especially when it comes to medical records and patient information.
- Security: Protecting sensitive patient information from unauthorized access or data breaches.
- Scalability: Designing the system to handle a growing number of patients, doctors, and administrative staff.
- Usability: Creating a user-friendly interface for both administrators and medical staff to efficiently manage hospital operations.
- Interoperability: Ensuring that the system can integrate with other healthcare systems and technologies to share data effectively.

4.2 Dependencies:

- Hardware: The system will depend on reliable hardware infrastructure to store and process data, including servers, networking equipment, and possibly medical devices.
- Software: Dependencies include the database management system (e.g., MySQL, PostgreSQL) and any other software used for the frontend or backend of the system.
- Regulatory Compliance: Compliance with healthcare regulations and standards (e.g., HIPAA in the United States) is critical and may require dependencies on specific software or protocols.
- Training and Support: Dependency on training programs and ongoing support for hospital staff to use the system effectively.

4.3 Applying Normalization:

1) FOOD ITEMS TABLE

PREVIOUSLY THE CODE WAS-

```
CREATE TABLE food_items (
    food_id INT PRIMARY KEY,
    food_type VARCHAR(100),
    food_price DECIMAL(10, 2),
    food_name VARCHAR(100),
    menu VARCHAR(100)
);
```

```
CREATE TABLE food_items (
    food_id INT PRIMARY KEY,
    food_type VARCHAR(100),
    food_price DECIMAL(10, 2),
    food_name VARCHAR(100),
    menu VARCHAR(100)
);
INSERT INTO food_items (food_id, food_type, food_price, food_name, menu)
VALUES
    (1, 'Appetizer', 8.99, 'Mozzarella Sticks', 'Appetizers'),
    (2, 'Main Course', 15.99, 'Grilled Salmon', 'Seafood'),
    (3, 'Dessert', 6.49, 'Chocolate Lava Cake', 'Desserts'),
    (4, 'Beverage', 2.99, 'Soda', 'Beverages'),
    (5, 'Main Course', 12.49, 'Chicken Alfredo Pasta', 'Pasta');
select * from food_items;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

food_id	food_type	food_price	food_name	menu
1	Appetizer	8.99	Mozzarella Sticks	Appetizers
2	Main Course	15.99	Grilled Salmon	Seafood
3	Dessert	6.49	Chocolate Lava Cake	Desserts
4	Beverage	2.99	Soda	Beverages
5	Main Course	12.49	Chicken Alfredo Pasta	Pasta
NULL	NULL	NULL	NULL	NULL

FOR 2NF-

Looking at the current structure of the table:

```
-- Create the menu table
CREATE TABLE menu (
    menu_id INT PRIMARY KEY,
    menu_name VARCHAR(100)
);

-- Create the food_items table with a foreign key reference to menu_id
CREATE TABLE food_items (
    food_id INT PRIMARY KEY,
    food_type VARCHAR(100),
    food_price DECIMAL(10, 2),
```

```
    food_name VARCHAR(100),
    menu_id INT,
    FOREIGN KEY (menu_id) REFERENCES menu(menu_id)
);
```

```
INSERT INTO menu (menu_id, menu_name) VALUES
(1, 'Breakfast'),
(2, 'Lunch'),
(3, 'Dinner');

-- Insert values into the food_items table
INSERT INTO food_items (food_id, food_type, food_price, food_name, menu_id) VALUES
(1, 'Main Course', 10.99, 'Pasta Carbonara', 2),
(2, 'Main Course', 12.50, 'Grilled Salmon', 2),
(3, 'Appetizer', 6.99, 'Caesar Salad', 2),
(4, 'Dessert', 5.99, 'Tiramisu', 2),
(5, 'Main Course', 8.99, 'Eggs Benedict', 1),
(6, 'Main Course', 9.99, 'Pancakes', 1),
(7, 'Appetizer', 4.50, 'Fruit Bowl', 1);
select * from food_items;
select * from menu;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

food_id	food_type	food_price	food_name	menu_id
1	Main Course	10.99	Pasta Carbonara	2
2	Main Course	12.50	Grilled Salmon	2
3	Appetizer	6.99	Caesar Salad	2
4	Dessert	5.99	Tiramisu	2
5	Main Course	8.99	Eggs Benedict	1
6	Main Course	9.99	Pancakes	1
7	Appetizer	4.50	Fruit Bowl	1
NULL	NULL	NULL	NULL	NULL

It appears that all attributes (`food_type`, `food_price`, `food_name`, `menu`) are fully dependent on the `food_id`, which is the primary key. So, the table is already in the Second Normal Form (2NF). No further modification is needed.

FOR 4NF-

```
CREATE TABLE food_items (
    food_id INT PRIMARY KEY,
    food_type VARCHAR(100),
    food_price DECIMAL(10, 2),
    food_name VARCHAR(100),
    menu VARCHAR(100)
);
```

Limit to 1000 rows

```
    food_name VARCHAR(100),
    menu VARCHAR(100)
);
INSERT INTO food_items (food_id, food_type, food_price, food_name, menu)
VALUES
    (1, 'Appetizer', 8.99, 'Mozzarella Sticks', 'Appetizers'),
    (2, 'Main Course', 15.99, 'Grilled Salmon', 'Seafood'),
    (3, 'Dessert', 6.49, 'Chocolate Lava Cake', 'Desserts'),
    (4, 'Beverage', 2.99, 'Soda', 'Beverages'),
    (5, 'Main Course', 12.49, 'Chicken Alfredo Pasta', 'Pasta');
select * from food_items;
drop table food_items;
CREATE TABLE food_items (
    food_id INT PRIMARY KEY,
    food_type VARCHAR(100),
    food_price DECIMAL(10, 2),
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

food_id	food_type	food_price	food_name	menu
1	Appetizer	8.99	Mozzarella Sticks	Appetizers
2	Main Course	15.99	Grilled Salmon	Seafood
3	Dessert	6.49	Chocolate Lava Cake	Desserts
4	Beverage	2.99	Soda	Beverages
5	Main Course	12.49	Chicken Alfredo Pasta	Pasta
NULL	NULL	NULL	NULL	NULL

```
INSERT INTO menu (menu_id, menu_name) VALUES
(1, 'Breakfast'),
(2, 'Lunch'),
(3, 'Dinner');

-- Insert values into the food_items table
INSERT INTO food_items (food_id, food_type, food_price, food_name, menu_id) VALUES
(1, 'Main Course', 10.99, 'Pasta Carbonara', 2),
(2, 'Main Course', 12.50, 'Grilled Salmon', 2),
(3, 'Appetizer', 6.99, 'Caesar Salad', 2),
(4, 'Dessert', 5.99, 'Tiramisu', 2),
(5, 'Main Course', 8.99, 'Eggs Benedict', 1),
(6, 'Main Course', 9.99, 'Pancakes', 1),
(7, 'Appetizer', 4.50, 'Fruit Bowl', 1);
select * from food_items;
select * from menu;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

menu_id	menu_name
1	Breakfast
2	Lunch
3	Dinner
NULL	NULL

We can see that there are no multi-valued attributes in this table. Each column seems to be atomic and fully dependent on the primary key, `food_id`. Therefore, the table is already in the Fourth Normal Form (4NF).

If you have any specific requirements or additional attributes that could introduce multi-valued dependencies, please let me know, and I can assist you further in ensuring 4NF compliance.

2. OUTLET TABLE

PREVIOUSLY THE CODE WAS

```
CREATE TABLE outlet (
    outlet_name VARCHAR(100) PRIMARY KEY,
    food_id INT,
    quantity_sold INT,
    menu VARCHAR(100),
    FOREIGN KEY (food_id) REFERENCES food_items(food_id)
);
```

```
        ('Pizza Palace', 'John Doe', '123-456-7890', 25.98, 1),
        ('Burger Barn', 'Jane Smith', '987-654-3210', 26.97, 2),
        ('Taco Time', 'Michael Johnson', '555-555-5555', 5.00, 3),
        ('Sushi Spot', 'Emily Brown', '333-333-3333', 12.50, 4),
        ('Healthy Delights', 'Chris Wilson', '444-444-4444', 7.50, 5);

    -- Insert 5 values into the rating table
    INSERT INTO rating (outlet_name, employee_id, employee_rating)
    VALUES
        ('Pizza Palace', 1, 5),
        ('Burger Barn', 2, 4),
        ('Taco Time', 3, 3),
        ('Sushi Spot', 4, 5),
        ('Healthy Delights', 5, 4);
    select * from outlet;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wra

outlet_name	food_id	quantity_sold	menu
Burger Barn	2	40	Main Course
Healthy Delights	5	35	Salad
Pizza Palace	1	50	Main Course
Sushi Spot	4	20	Appetizer
Taco Time	3	30	Appetizer
NULL	NULL	NULL	NULL

1NF-

Here are the modified tables:

```
CREATE TABLE outlet (
    outlet_name VARCHAR(100) PRIMARY KEY,
    food_id INT,
    quantity_sold INT,
    FOREIGN KEY (food_id) REFERENCES food_items(food_id)
);

CREATE TABLE food_items (
    food_id INT PRIMARY KEY,
    food_name VARCHAR(100),
    menu VARCHAR(100)
);
```

In this structure:

- The "outlet" table contains information about each outlet, including the outlet name, the ID of the food item sold, and the quantity sold.
- The "food_items" table contains information about each food item, including its ID, name, and menu description.

This arrangement adheres to the first normal form as each table has atomic values in its columns, and there are no repeating groups.

```
13      food_id INT,
14      quantity_sold INT,
15      menu_item VARCHAR(100),
16      PRIMARY KEY (outlet_name, food_id),
17      FOREIGN KEY (food_id) REFERENCES food_items(food_id)
18  );
19
20  INSERT INTO outlet (outlet_name, food_id, quantity_sold, menu_item)
21  VALUES
22      ('Outlet1', 1, 50, 'Burger'),
23      ('Outlet1', 2, 30, 'Pizza'),
24      ('Outlet2', 1, 40, 'Burger'),
25      ('Outlet2', 3, 20, 'Salad');
26
27  select * from outlet
28
```

esult Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Conten

outlet_name	food_id	quantity_sold	menu_item
Outlet1	1	50	Burger
Outlet1	2	30	Pizza
Outlet2	1	40	Burger
Outlet2	3	20	Salad
NULL	NULL	NULL	NULL

utlet 12 ×

5NF-

To achieve fifth normal form (5NF), we need to ensure that every join dependency in the schema is implied by the candidate keys of the relation. We'll break down the given table into multiple tables to eliminate any possible join dependencies. Here's the refactored version:

1. Outlet Table (outlet_name, menu):
   - This table contains outlet information and the menu they offer.

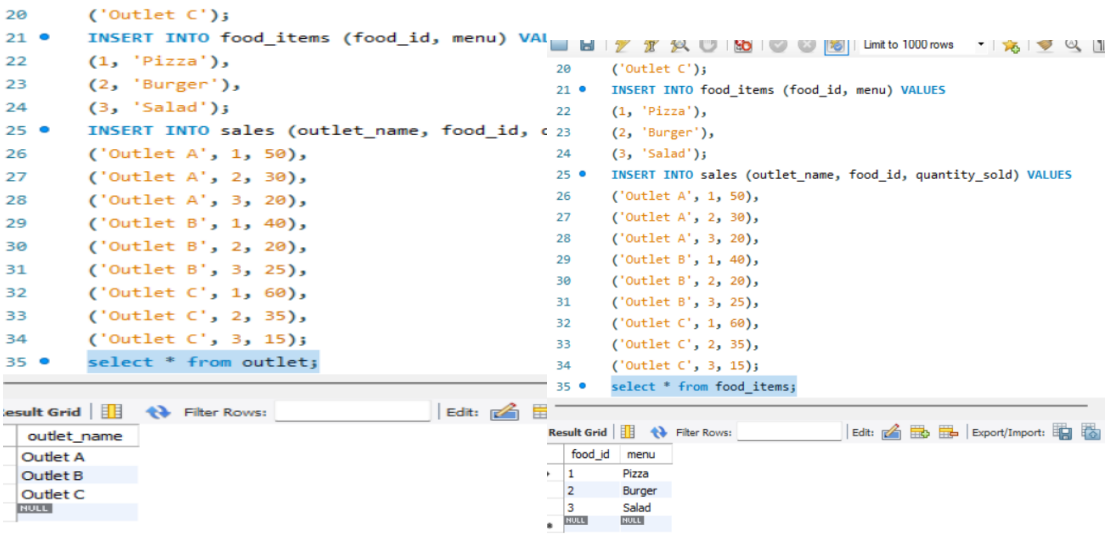
```
CREATE TABLE Outlet (
    outlet_name VARCHAR(100) PRIMARY KEY,
    menu VARCHAR(100)
);
```

2. Food Items Table (food_id, food_name):
   - This table contains information about each food item.

```
CREATE TABLE Food_Items (
    food_id INT PRIMARY KEY,
    food_name VARCHAR(100)
);
```

3. Sales Table (outlet_name, food_id, quantity_sold):**
   - This table tracks the sales of each food item at each outlet.

```
CREATE TABLE Sales (
    outlet_name VARCHAR(100),
    food_id INT,
    quantity_sold INT,
    PRIMARY KEY (outlet_name, food_id),
    FOREIGN KEY (outlet_name) REFERENCES Outlet(outlet_name),
    FOREIGN KEY (food_id) REFERENCES Food_Items(food_id)
);
```



```
20      ('Outlet C');
21  •   INSERT INTO food_items (food_id, menu) VALUES
22      (1, 'Pizza'),
23      (2, 'Burger'),
24      (3, 'Salad');
25  •   INSERT INTO sales (outlet_name, food_id, quantity_sold) VALUES
26      ('Outlet A', 1, 50),
27      ('Outlet A', 2, 30),
28      ('Outlet A', 3, 20),
29      ('Outlet B', 1, 40),
30      ('Outlet B', 2, 20),
31      ('Outlet B', 3, 25),
32      ('Outlet C', 1, 60),
33      ('Outlet C', 2, 35),
34      ('Outlet C', 3, 15);
35  •   select * from sales;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell C

outlet_name	food_id	quantity_sold
Outlet A	1	50
Outlet A	2	30
Outlet A	3	20
Outlet B	1	40
Outlet B	2	20
Outlet B	3	25
Outlet C	1	60
Outlet C	2	35
Outlet C	3	15

sales 3

By splitting the original table into these three tables, we ensure that each table represents a single entity and there are no redundant or derived attributes. This design adheres to 5NF and provides a clear structure for managing outlet information, food items, and sales data.

3. CUSTOMER TABLE

PREVIOUSLY THE CODE WAS-

```
CREATE TABLE customer (
    customer_name VARCHAR(100),
    phone VARCHAR(15) PRIMARY KEY,
    outlet_name VARCHAR(100),
    street VARCHAR(255),
    orders INT,
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name)
);
```

Limit to 1000 rows

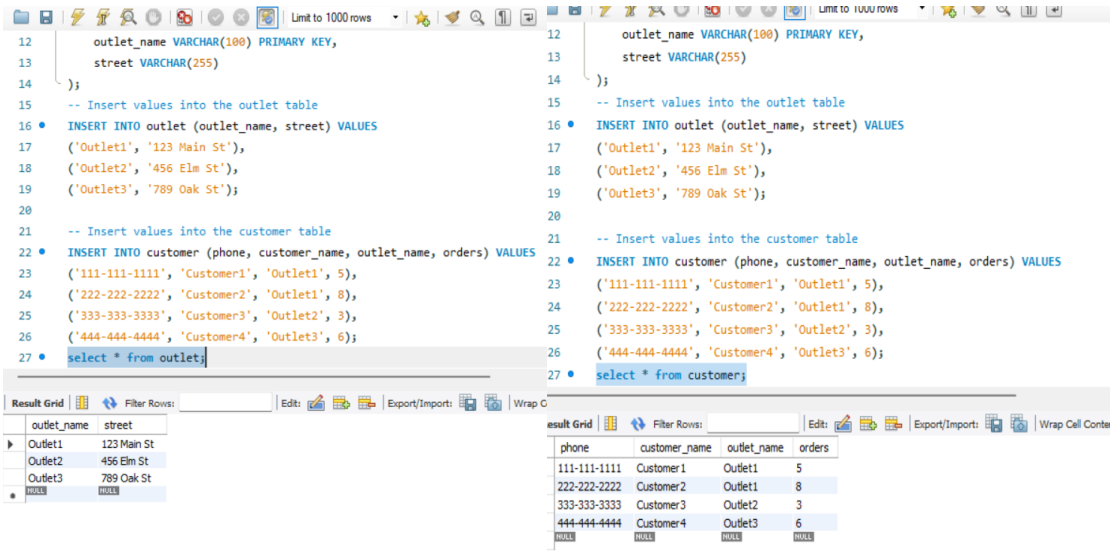
```
41      -- Insert values into invoice table
42  •   INSERT INTO invoice (outlet_name, customer_name, phone, amount, employee_id) VALUES
43      ('Outlet A', 'John Doe', '1234567890', 31.98, 1),
44      ('Outlet B', 'Jane Smith', '9876543210', 8.50, 2),
45      ('Outlet C', 'Alice Johnson', '5551234567', 20.97, 3),
46      ('Outlet D', 'Bob Brown', '4447890123', 5.98, 4),
47      ('Outlet E', 'Emma Davis', '2223334444', 18.99, 5);
48
49      -- Insert values into rating table
50  •   INSERT INTO rating (outlet_name, employee_id, employee_rating) VALUES
51      ('Outlet A', 1, 4),
52      ('Outlet B', 2, 5),
53      ('Outlet C', 3, 4),
54      ('Outlet D', 4, 3),
55      ('Outlet E', 5, 5);
56  •   select * from customer;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

customer_name	phone	outlet_name	street	orders
John Doe	1234567890	Outlet A	123 Main St	5
Emma Davis	2223334444	Outlet E	202 Maple St	6
Bob Brown	4447890123	Outlet D	101 Pine St	4
Alice Johnson	5551234567	Outlet C	789 Oak St	2
Jane Smith	9876543210	Outlet B	456 Elm St	3
NULL	NULL	NULL	NULL	NULL

3NF-

```
-- Customer Table
CREATE TABLE customer (
    phone VARCHAR(15) PRIMARY KEY,
    customer_name VARCHAR(100),
    outlet_name VARCHAR(100),
    orders INT,
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name)
);

-- Outlet Table
CREATE TABLE outlet (
    outlet_name VARCHAR(100) PRIMARY KEY,
    street VARCHAR(255)
);
```



4. EMPLOYEE TABLE

PREVIOUSLY THE CODE WAS-

```
CREATE TABLE employee (
    employee_id INT PRIMARY KEY,
    employee_name VARCHAR(100),
    outlet_name VARCHAR(100),
    dob DATE,
    gender CHAR(1),
    success VARCHAR(50),
    employee_jobs VARCHAR(100),
    receives DECIMAL(10, 2),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name)
);
```

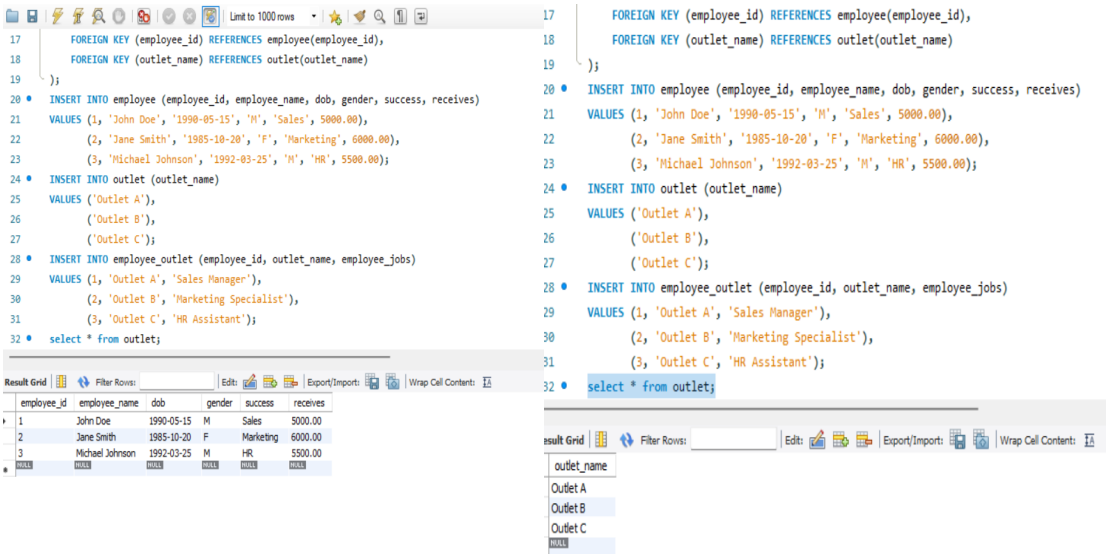
```
124     VALUES
125         ('Pizza Palace', 'John Doe', '123-456-7890', 25.98, 1),
126         ('Burger Barn', 'Jane Smith', '987-654-3210', 26.97, 2),
127         ('Taco Time', 'Michael Johnson', '555-555-5555', 5.00, 3),
128         ('Sushi Spot', 'Emily Brown', '333-333-3333', 12.50, 4),
129         ('Healthy Delights', 'Chris Wilson', '444-444-4444', 7.50, 5);
130
131     -- Insert 5 values into the rating table
132     INSERT INTO rating (outlet_name, employee_id, employee_rating)
133     VALUES
134         ('Pizza Palace', 1, 5),
135         ('Burger Barn', 2, 4),
136         ('Taco Time', 3, 3),
137         ('Sushi Spot', 4, 5),
138         ('Healthy Delights', 5, 4);
139     select * from employee;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

employee_id	employee_name	outlet_name	dob	gender	success	employee_jobs	receives
1	Alice Johnson	Pizza Palace	1990-05-15	F	Employee of the Month	Chef	1500.00
2	Bob Smith	Burger Barn	1988-09-20	M	Top Sales Performer	Cashier	1200.00
3	Charlie Brown	Taco Time	1995-03-10	M	Most Improved Employee	Server	1000.00
4	Diana Wilson	Sushi Spot	1992-11-25	F	Outstanding Customer Service	Chef	1400.00
5	Emma Davis	Healthy Delights	1993-07-08	F	Employee of the Month	Salad Preparer	1300.00
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

5NF-

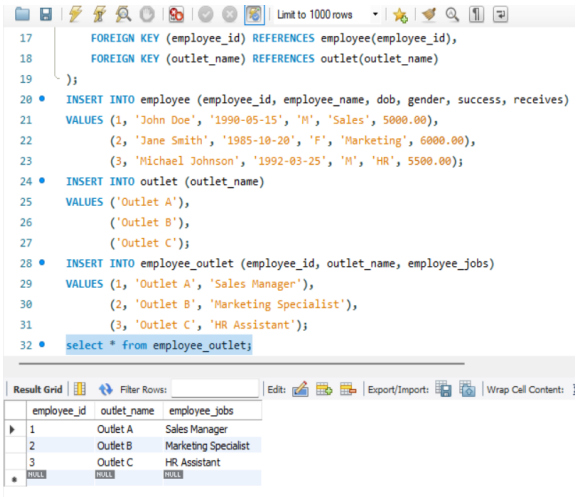
```
CREATE TABLE employee (
    employee_id INT PRIMARY KEY,
    employee_name VARCHAR(100),
    dob DATE,
    gender CHAR(1),
    success VARCHAR(50),
    receives DECIMAL(10, 2)
);
CREATE TABLE outlet (
    outlet_name VARCHAR(100) PRIMARY KEY
);
CREATE TABLE employee_outlet (
    employee_id INT,
    outlet_name VARCHAR(100),
    employee_jobs VARCHAR(100),
    PRIMARY KEY (employee_id, outlet_name),
    FOREIGN KEY (employee_id) REFERENCES employee(employee_id),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name)
);
```



```
    FOREIGN KEY (employee_id) REFERENCES employee(employee_id),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name)
);
INSERT INTO employee (employee_id, employee_name, dob, gender, success, receives)
VALUES (1, 'John Doe', '1990-05-15', 'M', 'Sales', 5000.00),
       (2, 'Jane Smith', '1985-10-20', 'F', 'Marketing', 6000.00),
       (3, 'Michael Johnson', '1992-03-25', 'M', 'HR', 5500.00);
INSERT INTO outlet (outlet_name)
VALUES ('Outlet A'),
       ('Outlet B'),
       ('Outlet C');
INSERT INTO employee_outlet (employee_id, outlet_name, employee_jobs)
VALUES (1, 'Outlet A', 'Sales Manager'),
       (2, 'Outlet B', 'Marketing Specialist'),
       (3, 'Outlet C', 'HR Assistant');
select * from outlet;
```

employee_id	employee_name	dob	gender	success	receives
1	John Doe	1990-05-15	M	Sales	5000.00
2	Jane Smith	1985-10-20	F	Marketing	6000.00
3	Michael Johnson	1992-03-25	M	HR	5500.00

outlet_name
Outlet A
Outlet B
Outlet C

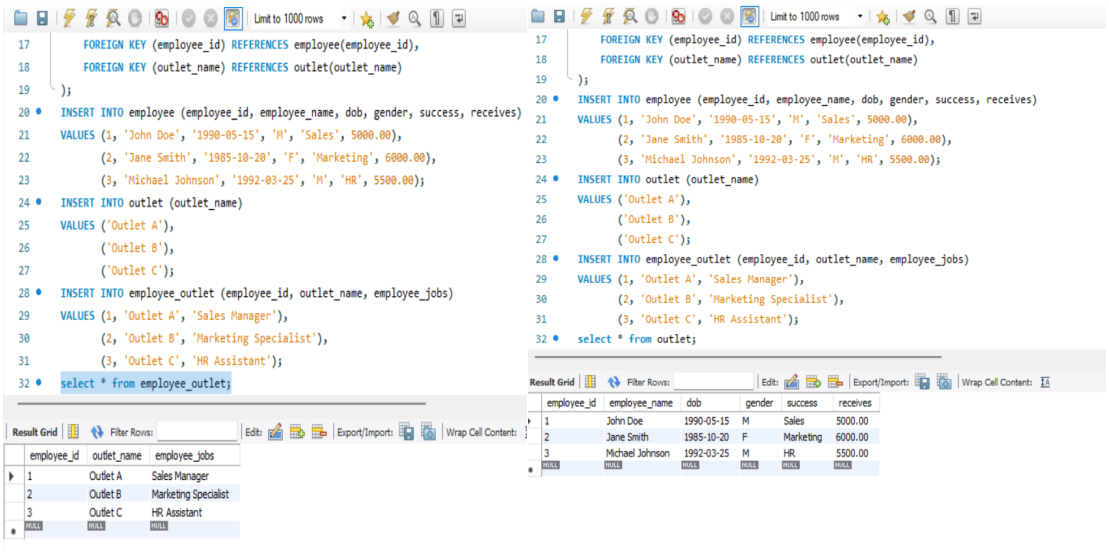


```
    FOREIGN KEY (employee_id) REFERENCES employee(employee_id),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name)
);
INSERT INTO employee (employee_id, employee_name, dob, gender, success, receives)
VALUES (1, 'John Doe', '1990-05-15', 'M', 'Sales', 5000.00),
       (2, 'Jane Smith', '1985-10-20', 'F', 'Marketing', 6000.00),
       (3, 'Michael Johnson', '1992-03-25', 'M', 'HR', 5500.00);
INSERT INTO outlet (outlet_name)
VALUES ('Outlet A'),
       ('Outlet B'),
       ('Outlet C');
INSERT INTO employee_outlet (employee_id, outlet_name, employee_jobs)
VALUES (1, 'Outlet A', 'Sales Manager'),
       (2, 'Outlet B', 'Marketing Specialist'),
       (3, 'Outlet C', 'HR Assistant');
select * from employee_outlet;
```

employee_id	outlet_name	employee_jobs
1	Outlet A	Sales Manager
2	Outlet B	Marketing Specialist
3	Outlet C	HR Assistant

4NF-

```
CREATE TABLE employee (
    employee_id INT PRIMARY KEY,
    employee_name VARCHAR(100),
    dob DATE,
    gender CHAR(1),
    success VARCHAR(50),
    employee_jobs VARCHAR(100),
    receives DECIMAL(10, 2)
);

CREATE TABLE outlet_employee_relation (
    employee_id INT PRIMARY KEY,
    outlet_name VARCHAR(100),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name)
);
```



```
    FOREIGN KEY (employee_id) REFERENCES employee(employee_id),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name)
);
INSERT INTO employee (employee_id, employee_name, dob, gender, success, receives)
VALUES (1, 'John Doe', '1990-05-15', 'M', 'Sales', 5000.00),
       (2, 'Jane Smith', '1985-10-20', 'F', 'Marketing', 6000.00),
       (3, 'Michael Johnson', '1992-03-25', 'M', 'HR', 5500.00);
INSERT INTO outlet (outlet_name)
VALUES ('Outlet A'),
       ('Outlet B'),
       ('Outlet C');
INSERT INTO employee_outlet (employee_id, outlet_name, employee_jobs)
VALUES (1, 'Outlet A', 'Sales Manager'),
       (2, 'Outlet B', 'Marketing Specialist'),
       (3, 'Outlet C', 'HR Assistant');
select * from employee_outlet;
```

employee_id	outlet_name	employee_jobs
1	Outlet A	Sales Manager
2	Outlet B	Marketing Specialist
3	Outlet C	HR Assistant

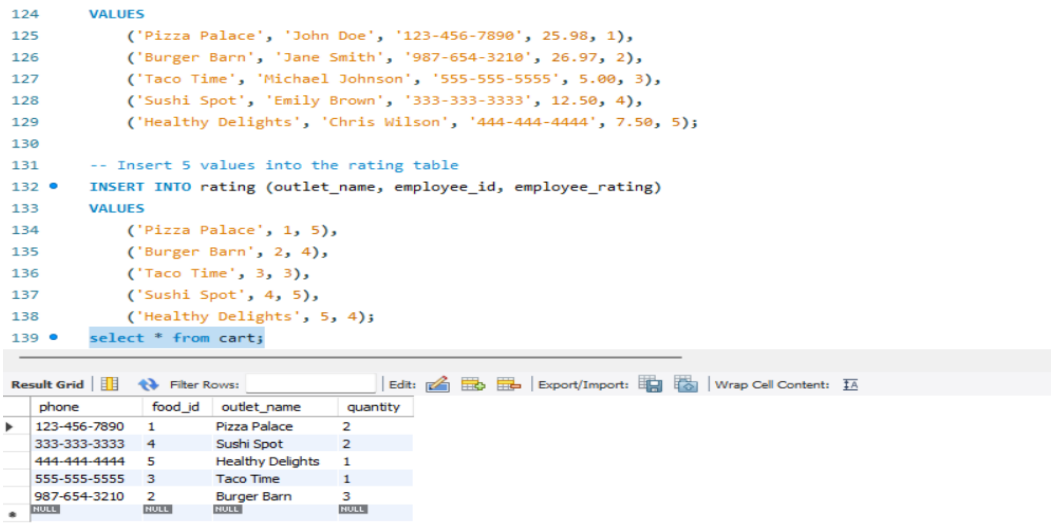
```
    FOREIGN KEY (employee_id) REFERENCES employee(employee_id),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name)
);
INSERT INTO employee (employee_id, employee_name, dob, gender, success, receives)
VALUES (1, 'John Doe', '1990-05-15', 'M', 'Sales', 5000.00),
       (2, 'Jane Smith', '1985-10-20', 'F', 'Marketing', 6000.00),
       (3, 'Michael Johnson', '1992-03-25', 'M', 'HR', 5500.00);
INSERT INTO outlet (outlet_name)
VALUES ('Outlet A'),
       ('Outlet B'),
       ('Outlet C');
INSERT INTO employee_outlet (employee_id, outlet_name, employee_jobs)
VALUES (1, 'Outlet A', 'Sales Manager'),
       (2, 'Outlet B', 'Marketing Specialist'),
       (3, 'Outlet C', 'HR Assistant');
select * from outlet;
```

employee_id	employee_name	dob	gender	success	receives
1	John Doe	1990-05-15	M	Sales	5000.00
2	Jane Smith	1985-10-20	F	Marketing	6000.00
3	Michael Johnson	1992-03-25	M	HR	5500.00

5. CART TABLE

ORIGINAL CODE-

```
CREATE TABLE cart (
    phone VARCHAR(15),
    food_id INT,
    outlet_name VARCHAR(100),
    quantity INT,
    PRIMARY KEY (phone, food_id, outlet_name),
    FOREIGN KEY (phone) REFERENCES customer(phone),
    FOREIGN KEY (food_id) REFERENCES food_items(food_id),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name)
);
```



```
    VALUES
        ('Pizza Palace', 'John Doe', '123-456-7890', 25.98, 1),
        ('Burger Barn', 'Jane Smith', '987-654-3210', 26.97, 2),
        ('Taco Time', 'Michael Johnson', '555-555-5555', 5.00, 3),
        ('Sushi Spot', 'Emily Brown', '333-333-3333', 12.50, 4),
        ('Healthy Delights', 'Chris Wilson', '444-444-4444', 7.50, 5);

    -- Insert 5 values into the rating table
    INSERT INTO rating (outlet_name, employee_id, employee_rating)
    VALUES
        ('Pizza Palace', 1, 5),
        ('Burger Barn', 2, 4),
        ('Taco Time', 3, 3),
        ('Sushi Spot', 4, 5),
        ('Healthy Delights', 5, 4);
    select * from cart;
```

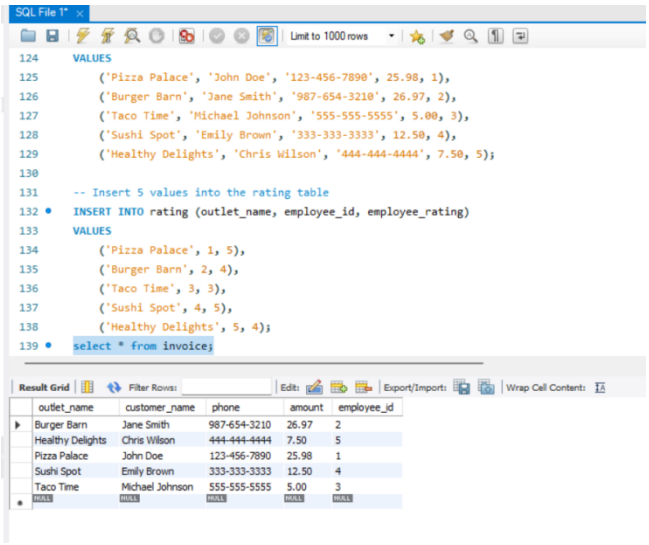
phone	food_id	outlet_name	quantity
123-456-7890	1	Pizza Palace	2
333-333-3333	4	Sushi Spot	2
444-444-4444	5	Healthy Delights	1
555-555-5555	3	Taco Time	1
987-654-3210	2	Burger Barn	3

6. INVOICE TABLE

ORIGINAL CODE-

```
CREATE TABLE invoice (
```

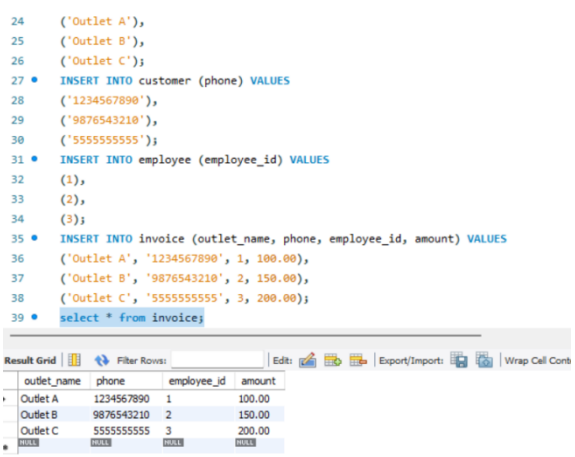
```
    outlet_name VARCHAR(100),
    customer_name VARCHAR(100),
    phone VARCHAR(15),
    amount DECIMAL(10, 2),
    employee_id INT,
    PRIMARY KEY (outlet_name, phone, employee_id),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name),
    FOREIGN KEY (phone) REFERENCES customer(phone),
    FOREIGN KEY (employee_id) REFERENCES employee(employee_id)
);
```

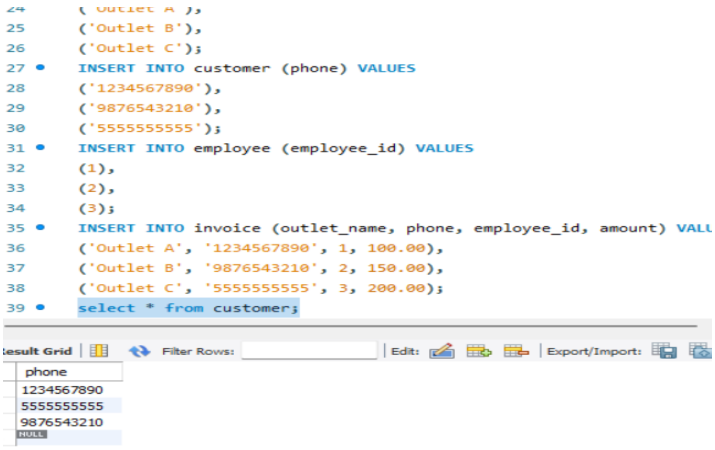


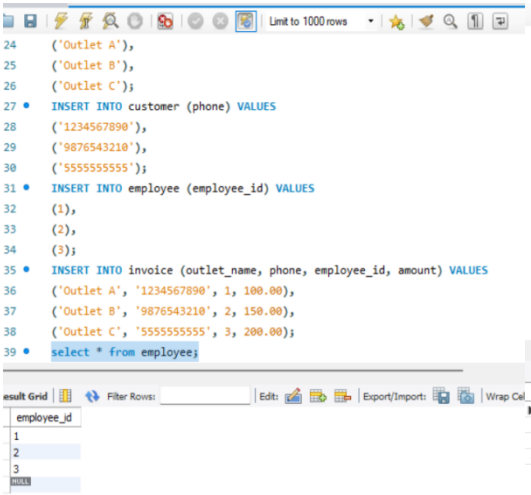
3NF-

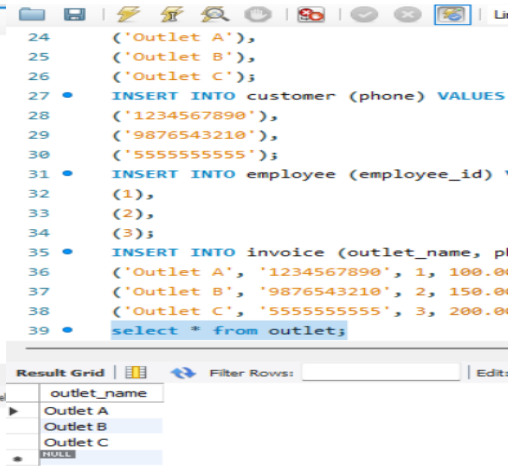
```
CREATE TABLE invoice (
    outlet_name VARCHAR(100),
    phone VARCHAR(15),
    employee_id INT,
    amount DECIMAL(10, 2),
    PRIMARY KEY (outlet_name, phone, employee_id),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name),
    FOREIGN KEY (phone) REFERENCES customer(phone),
    FOREIGN KEY (employee_id) REFERENCES employee(employee_id)
);

CREATE TABLE outlet (
    outlet_name VARCHAR(100) PRIMARY KEY
    -- Other outlet attributes here
);

CREATE TABLE customer (
    phone VARCHAR(15) PRIMARY KEY
    -- Other customer attributes here
);

CREATE TABLE employee (
    employee_id INT PRIMARY KEY
```

```
    -- Other employee attributes here
);
```









7. RATING TABLE

ORIGINAL CODE-

```
CREATE TABLE rating (
    outlet_name VARCHAR(100),
    employee_id INT,
    employee_rating INT,
    PRIMARY KEY (outlet_name, employee_id),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name),
    FOREIGN KEY (employee_id) REFERENCES employee(employee_id)
);
```

```
124     VALUES
125         ('Pizza Palace', 'John Doe', '123-456-7890', 25.98, 1),
126         ('Burger Barn', 'Jane Smith', '987-654-3210', 26.97, 2),
127         ('Taco Time', 'Michael Johnson', '555-555-5555', 5.00, 3),
128         ('Sushi Spot', 'Emily Brown', '333-333-3333', 12.50, 4),
129         ('Healthy Delights', 'Chris Wilson', '444-444-4444', 7.50, 5);
130
131     -- Insert 5 values into the rating table
132 •   INSERT INTO rating (outlet_name, employee_id, employee_rating)
133     VALUES
134         ('Pizza Palace', 1, 5),
135         ('Burger Barn', 2, 4),
136         ('Taco Time', 3, 3),
137         ('Sushi Spot', 4, 5),
138         ('Healthy Delights', 5, 4);
139 •   select * from rating;
```

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

outlet_name	employee_id	employee_rating
Burger Barn	2	4
Healthy Delights	5	4
Pizza Palace	1	5
Sushi Spot	4	5
Taco Time	3	3
NULL	NULL	NULL

1NF-

```
CREATE TABLE rating (
    rating_id INT PRIMARY KEY AUTO_INCREMENT,
    outlet_name VARCHAR(100),
    employee_id INT,
    employee_rating INT,
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name),
    FOREIGN KEY (employee_id) REFERENCES employee(employee_id)
);
```

```
124     VALUES
125         ('Pizza Palace', 'John Doe', '123-456-7890', 25.98, 1),
126         ('Burger Barn', 'Jane Smith', '987-654-3210', 26.97, 2),
127         ('Taco Time', 'Michael Johnson', '555-555-5555', 5.00, 3),
128         ('Sushi Spot', 'Emily Brown', '333-333-3333', 12.50, 4),
129         ('Healthy Delights', 'Chris Wilson', '444-444-4444', 7.50, 5);
130
131     -- Insert 5 values into the rating table
132 •   INSERT INTO rating (outlet_name, employee_id, employee_rating)
133     VALUES
134         ('Pizza Palace', 1, 5),
135         ('Burger Barn', 2, 4),
136         ('Taco Time', 3, 3),
137         ('Sushi Spot', 4, 5),
138         ('Healthy Delights', 5, 4);
139 •   select * from rating;
```

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

outlet_name	employee_id	employee_rating
Burger Barn	2	4
Healthy Delights	5	4
Pizza Palace	1	5
Sushi Spot	4	5
Taco Time	3	3
NULL	NULL	NULL

5) Implementation of concurrency control and recovery mechanisms

1)Lock-Based Concurrency Control (LBCC):

BEGIN TRANSACTION;

LOCK TABLE patients IN EXCLUSIVE MODE;

UPDATE patients SET diagnosis = 'New Diagnosis' WHERE patient_id = 123;

COMMIT;

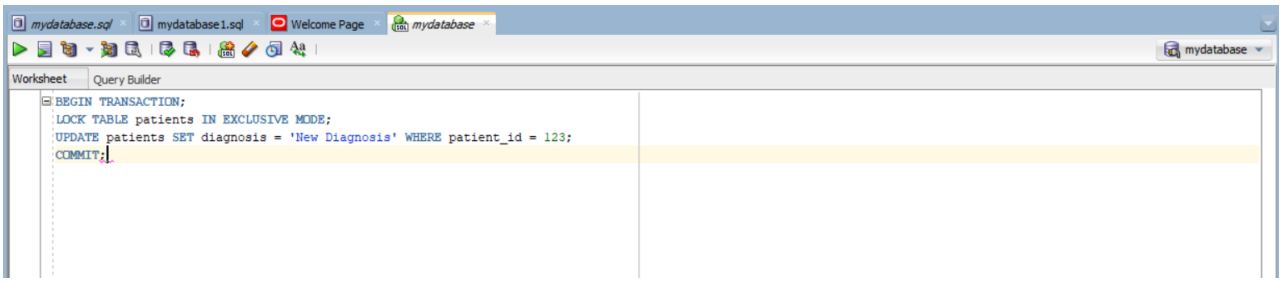


Fig 5.1 Lock-Based Concurrency Control

2) Timestamp-based concurrency control

BEGIN TRANSACTION;

SELECT MAX(W_TS) INTO @max_w_ts FROM patients WHERE patient_id = 123;

IF (@max_w_ts < current_transaction_timestamp) THEN

  UPDATE patients SET diagnosis = 'New Diagnosis', W_TS = current_transaction_timestamp WHERE patient_id = 123;

ELSE

  -- Handle timestamp conflict (e.g., rollback or retry)

END IF;

COMMIT;

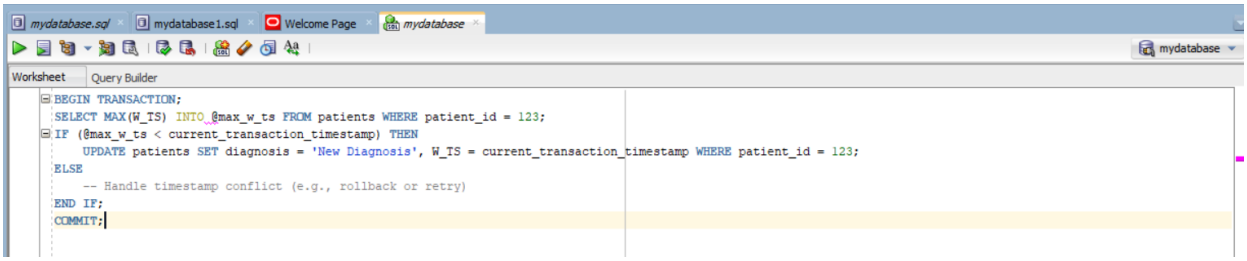


Fig 5.2 Timestamp-based concurrency control

3)Optimistic Concurrency Control (OCC):

BEGIN TRANSACTION;

-- Assume local copies of data are maintained in application code

UPDATE patients SET diagnosis = 'New Diagnosis' WHERE patient_id = 123;

-- Check for conflicts and handle them (e.g., rollback or retry)

COMMIT;

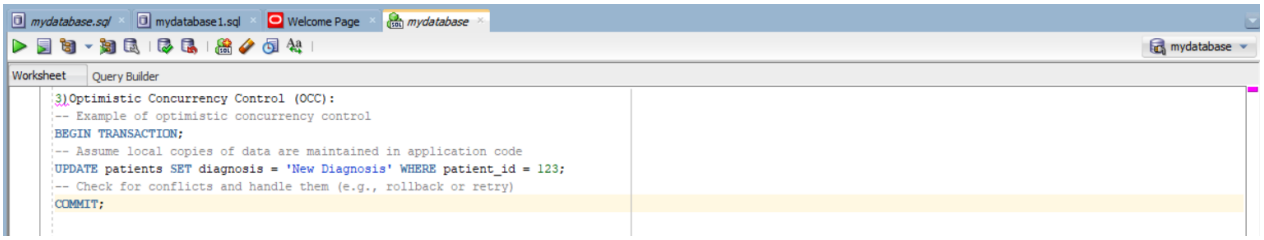


Fig 5.3 Optimistic Concurrency Control

6) Code for the Project

6.1 Implementation of python

Python, coupled with Flask for web development and SQLAlchemy for database management, powers the hospital management system efficiently. Utilizing Flask routes, SQLAlchemy models, and Flask-Login for user authentication, the system manages users, patients, doctors, and bookings seamlessly. HTML templates render dynamic pages, while form handling captures user input for processing and storage. Flash messages provide immediate feedback, enhancing user experience.

6.1.1 main.py

```
-- Create the food_items table first since it's referenced in the outlet and
cart tables
CREATE TABLE food_items (
    food_id INT PRIMARY KEY,
    food_type VARCHAR(100),
    food_price DECIMAL(10, 2),
    food_name VARCHAR(100),
    menu VARCHAR(100)
);

-- Create the outlet table, referencing food_items
CREATE TABLE outlet (
    outlet_name VARCHAR(100) PRIMARY KEY,
    food_id INT,
    quantity_sold INT,
    menu VARCHAR(100),
    FOREIGN KEY (food_id) REFERENCES food_items(food_id)
);

-- Create the customer table, referencing outlet
CREATE TABLE customer (
    customer_name VARCHAR(100),
    phone VARCHAR(15) PRIMARY KEY,
    outlet_name VARCHAR(100),
    street VARCHAR(255),
    orders INT,
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name)
```

```
);

-- Create the employee table, referencing outlet
CREATE TABLE employee (
    employee_id INT PRIMARY KEY,
    employee_name VARCHAR(100),
    outlet_name VARCHAR(100),
    dob DATE,
    gender CHAR(1),
    success VARCHAR(50),
    employee_jobs VARCHAR(100),
    receives DECIMAL(10, 2),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name)
);

-- Create the cart table, referencing customer, food_items, and outlet
CREATE TABLE cart (
    phone VARCHAR(15),
    food_id INT,
    outlet_name VARCHAR(100),
    quantity INT,
    PRIMARY KEY (phone, food_id, outlet_name),
    FOREIGN KEY (phone) REFERENCES customer(phone),
    FOREIGN KEY (food_id) REFERENCES food_items(food_id),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name)
);

-- Create the invoice table, referencing customer, outlet, and employee
CREATE TABLE invoice (
```

```
    outlet_name VARCHAR(100),
    customer_name VARCHAR(100),
    phone VARCHAR(15),
    amount DECIMAL(10, 2),
    employee_id INT,
    PRIMARY KEY (outlet_name, phone, employee_id),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name),
    FOREIGN KEY (phone) REFERENCES customer(phone),
    FOREIGN KEY (employee_id) REFERENCES employee(employee_id)
);

-- Create the rating table, referencing outlet and employee
CREATE TABLE rating (
    outlet_name VARCHAR(100),
    employee_id INT,
    employee_rating INT,
    PRIMARY KEY (outlet_name, employee_id),
    FOREIGN KEY (outlet_name) REFERENCES outlet(outlet_name),
    FOREIGN KEY (employee_id) REFERENCES employee(employee_id)
);

-- Inserting data into the food_items table
INSERT INTO food_items (food_id, food_type, food_price, food_name,
menu) VALUES
(1, 'Main Course', 15.99, 'Spaghetti Bolognese', 'Main Menu'),
(2, 'Appetizer', 8.50, 'Caesar Salad', 'Main Menu'),
(3, 'Dessert', 6.99, 'Chocolate Lava Cake', 'Desserts Menu'),
(4, 'Beverage', 2.99, 'Iced Tea', 'Beverages Menu'),
(5, 'Main Course', 18.99, 'Grilled Salmon', 'Main Menu');
```

```
-- Inserting data into the outlet table
INSERT INTO outlet (outlet_name, food_id, quantity_sold, menu)
VALUES
('Outlet A', 1, 50, 'Main Menu'),
('Outlet B', 2, 30, 'Main Menu'),
('Outlet C', 3, 20, 'Desserts Menu'),
('Outlet D', 4, 40, 'Beverages Menu'),
('Outlet E', 5, 60, 'Main Menu');

-- Inserting data into the customer table
INSERT INTO customer (customer_name, phone, outlet_name, street,
orders) VALUES
('John Doe', '1234567890', 'Outlet A', '123 Main St', 5),
('Jane Smith', '9876543210', 'Outlet B', '456 Elm St', 3),
('Alice Johnson', '5551234567', 'Outlet C', '789 Oak St', 2),
('Bob Brown', '4447890123', 'Outlet D', '101 Pine St', 4),
('Emma Davis', '2223334444', 'Outlet E', '202 Maple St', 6);

-- Inserting data into the employee table
INSERT INTO employee (employee_id, employee_name, outlet_name,
dob, gender, success, employee_jobs, receives) VALUES
(1, 'Michael Scott', 'Outlet A', '1980-05-15', 'M', 'Manager', 'Cook',
2000.00),
(2, 'Pam Beesly', 'Outlet B', '1985-11-30', 'F', 'Assistant Manager', 'Server',
1500.00),
(3, 'Jim Halpert', 'Outlet C', '1982-07-15', 'M', 'Chef', 'Bartender',
1800.00),
(4, 'Dwight Schrute', 'Outlet D', '1978-03-20', 'M', 'Cashier', 'Delivery',
```

```
1600.00),
(5, 'Angela Martin', 'Outlet E', '1987-01-05', 'F', 'Server', 'Cleaner',
1400.00);

-- Inserting data into the cart table
INSERT INTO cart (phone, food_id, outlet_name, quantity) VALUES
('1234567890', 1, 'Outlet A', 2),
('9876543210', 2, 'Outlet B', 1),
('5551234567', 3, 'Outlet C', 3),
('4447890123', 4, 'Outlet D', 2),
('2223334444', 5, 'Outlet E', 1);

-- Inserting data into the invoice table
INSERT INTO invoice (outlet_name, customer_name, phone, amount,
employee_id) VALUES
('Outlet A', 'John Doe', '1234567890', 31.98, 1),
('Outlet B', 'Jane Smith', '9876543210', 8.50, 2),
('Outlet C', 'Alice Johnson', '5551234567', 20.97, 3),
('Outlet D', 'Bob Brown', '4447890123', 5.98, 4),
('Outlet E', 'Emma Davis', '2223334444', 18.99, 5);

-- Inserting data into the rating table
INSERT INTO rating (outlet_name, employee_id, employee_rating)
VALUES
('Outlet A', 1, 4),
('Outlet B', 2, 5),
('Outlet C', 3, 4),
('Outlet D', 4, 3),
('Outlet E', 5, 5);
```

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.mi
n.css" integrity="sha384-
9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+Nc
Pb1dKGj7Sk" crossorigin="anonymous">
    <link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.8.2/css/all.css"
integrity="sha384-
oS3vJWv+0UjzBfQzYUhtDYW+Pj2yciDJxpsK1OYPAYjqT085Qq/1cq5F
LXAZQ7Ay" crossorigin="anonymous">

    <title>About Us</title>
    <link rel = "icon" href ="img/logo.jpg" type = "image/x-icon">
    <!-- Google Fonts -->
    <link
href="https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,40
0i,600,600i,700,700i|Roboto:300,300i,400,400i,500,500i,700,700i|Poppins:30
0,300i,400,400i,500,500i,600,600i,700,700i" rel="stylesheet">
```

```
  <!-- Vendor CSS Files -->
  <link href="assets/vendor/bootstrap/css/bootstrap.min.css"
rel="stylesheet">
  <link href="assets/vendor/icofont/icofont.min.css" rel="stylesheet">
  <link href="assets/vendor/animate.css/animate.min.css"
rel="stylesheet">

  <!-- Template Main CSS File -->
  <link href="assets/css/style.css" rel="stylesheet">

 </head>
 <body>
 <?php include 'partials/_dbconnect.php';?>
 <?php include 'partials/_nav.php';?>

   <!-- ======= Hero Section ======= -->
 <section id="hero">
  <div class="hero-container">
   <div id="heroCarousel" class="carousel slide carousel-fade" data-
ride="carousel">
     <ol class="carousel-indicators" id="hero-carousel-indicators"></ol>
     <div class="carousel-inner" role="listbox">
      <!-- Slide 1 -->
      <div class="carousel-item active">
       <div class="carousel-background"><img
src="assets/img/slide/slide-1.jpg" alt=""></div>
       <div class="carousel-container">
        <div class="carousel-content">
         <h2 class="animate__animated
```

```
animate__fadeInDown">Welcome to <span>Pizza World</span></h2>
            <a href="index.php" class="btn-get-started animate__animated
animate__fadeInUp scrollto">Get Started</a>
          </div>
        </div>
      </div>
      <!-- Slide 2 -->
      <div class="carousel-item">
        <div class="carousel-background"><img
src="assets/img/slide/slide-2.jpg" alt=""></div>
        <div class="carousel-container">
          <div class="carousel-content">
            <h2 class="animate__animated animate__fadeInDown mb-
0">Our Mission</h2>
            <p class="animate__animated animate__fadeInUp">To be
number one</p>
            <a href="index.php" class="btn-get-started animate__animated
animate__fadeInUp scrollto">Get Started</a>
          </div>
        </div>
      </div>
      <!-- Slide 3 -->
      <div class="carousel-item">
        <div class="carousel-background"><img
src="assets/img/slide/slide-3.jpg" alt=""></div>
        <div class="carousel-container">
          <div class="carousel-content">
            <h2 class="animate__animated animate__fadeInDown mb-
0">Parmar Darshan Kiritbhai</h2><p>CE084    <a
```

```
href="https://github.com/darshankparmar"
target="_blank">@darshankparmar</a></p>
            <a href="index.php" class="btn-get-started animate__animated
animate__fadeInUp scrollto">Get Started</a>
          </div>
        </div>
      </div>
    </div>

    <a class="carousel-control-prev" href="#heroCarousel"
role="button" data-slide="prev">
      <span class="carousel-control-prev-icon icofont-thin-double-left"
aria-hidden="true"></span>
      <span class="sr-only">Previous</span>
    </a>

    <a class="carousel-control-next" href="#heroCarousel"
role="button" data-slide="next">
      <span class="carousel-control-next-icon icofont-thin-double-right"
aria-hidden="true"></span>
      <span class="sr-only">Next</span>
    </a>

  </div>
 </div>
</section><!-- End Hero -->

<main id="main">
```

```
  <!-- ======= About Us Section ======= -->
  <section id="about" class="about">
    <div class="container">

      <div class="section-title">
        <h2>About Us</h2>
      </div>

      <div class="row">
        <div class="col-lg-6">
          <h3>Welcome to <strong>Pizza World</strong></h3>
          <h3><strong>The Worldwide Leader in Pizza
Delivery</strong></h3>
          <p class="font-italic">
        </div>
        <div class="col-lg-6 pt-4 pt-lg-0 content">
          <div class="skills-content">
            <p><b>Rating: </b></p>
            <div class="progress">
              <span class="skill">5 star <i class="val">93%</i></span>
              <div class="progress-bar-wrap">
                <div class="progress-bar" role="progressbar" aria-
valuenow="93" aria-valuemin="0" aria-valuemax="100"></div>
              </div>
            </div>

            <div class="progress">
              <span class="skill">4 star <i class="val">90%</i></span>
              <div class="progress-bar-wrap">
```

```
            <div class="progress-bar" role="progressbar" aria-
valuenow="90" aria-valuemin="0" aria-valuemax="100"></div>
          </div>
        </div>

        <div class="progress">
          <span class="skill">3 star <i class="val">30%</i></span>
          <div class="progress-bar-wrap">
            <div class="progress-bar" role="progressbar" aria-
valuenow="30" aria-valuemin="0" aria-valuemax="100"></div>
          </div>
        </div>

        <div class="progress">
          <span class="skill">2 star <i class="val">5%</i></span>
          <div class="progress-bar-wrap">
            <div class="progress-bar" role="progressbar" aria-
valuenow="5" aria-valuemin="0" aria-valuemax="100"></div>
          </div>
        </div>

        <div class="progress">
          <span class="skill">1 star <i class="val">0%</i></span>
          <div class="progress-bar-wrap">
            <div class="progress-bar" role="progressbar" aria-
valuenow="0" aria-valuemin="0" aria-valuemax="100"></div>
          </div>
        </div>
```

```
      </div>
     </div>
    </div>
   </div>
</section><!-- End About Us Section -->

<!-- ======= Counts Section ======= -->
<section class="counts section-bg">
 <div class="container">

  <div class="row no-gutters">

   <div class="col-lg-3 col-md-6 d-md-flex align-items-md-stretch">
    <div class="count-box">
     <i class="icofont-simple-smile"></i>
     <span data-toggle="counter-up">232</span>
     <p><strong>Happy Customers</strong></p>
    </div>
   </div>

   <div class="col-lg-3 col-md-6 d-md-flex align-items-md-stretch">
    <div class="count-box">
     <i class="icofont-document-folder"></i>
     <span data-toggle="counter-up">121</span>
     <p><strong>Items</strong></p>
    </div>
   </div>

   <div class="col-lg-3 col-md-6 d-md-flex align-items-md-stretch">
```

```
        <div class="count-box">
          <i class="icofont-live-support"></i>
          <span data-toggle="counter-up">1,463</span>
          <p><strong>Hours Of Support</strong></p>
        </div>
      </div>

      <div class="col-lg-3 col-md-6 d-md-flex align-items-md-stretch">
        <div class="count-box">
          <i class="icofont-users-alt-5"></i>
          <span data-toggle="counter-up">15</span>
          <p><strong>Hard Workers</strong></p>
        </div>
      </div>

    </div>

  </div>
</section><!-- End Counts Section -->

<!-- ======= Our Team Section ======= -->
<section id="team" class="team">
  <div class="container">

    <div class="section-title">
      <h2>Our Team</h2>
    </div>

    <div class="row" style="padding-left: 228px;">
```

```
        <div class="col-xl-3 col-lg-4 col-md-6" data-wow-delay="0.1s">
          <div class="member">
            <img src="assets/img/team/team-1.jpg" class="img-fluid" alt="">
            <div class="member-info">
              <div class="member-info-content">
                <h4>Darshan Parmar</h4>
              </div>
              <div class="social">
                <a href="https://twitter.com/darshankparmar"
target="_blank"><i class="icofont-twitter"></i></a>
                <a href="https://github.com/darshankparmar"
target="_blank"><i class="fab fa-github"></i></a>
                <a href="https://www.linkedin.com/in/darshankparmar/"
target="_blank"><i class="icofont-linkedin" target="_blank"></i></a>
              </div>
            </div>
          </div>
        </div>

        <div class="col-xl-3 col-lg-4 col-md-6" data-wow-delay="0.2s">
          <div class="member">
            <img src="assets/img/team/team-3.jpg" class="img-fluid" alt="">
            <div class="member-info">
              <div class="member-info-content">
                <h4>Harsh Patel</h4>
              </div>
              <div class="social">
                <a href=""><i class="icofont-twitter"
```

```
target="_blank"></i></a>
              <a href="https://github.com/7Har" target="_blank"><i
class="fab fa-github"></i></a>
              <a href=""><i class="icofont-linkedin"
target="_blank"></i></a>
            </div>
          </div>
        </div>
      </div>

      <div class="col-xl-3 col-lg-4 col-md-6" data-wow-delay="0.2s">
        <div class="member">
          <img src="assets/img/team/team-5.jpg" class="img-fluid" alt=""
style="height: 198px;width: 198px;">
          <div class="member-info">
            <div class="member-info-content">
              <h4>Bhavesh Parmar</h4>
            </div>
            <div class="social">
              <a href=""><i class="icofont-twitter"
target="_blank"></i></a>
              <a href="https://github.com/Blparmar007" target="_blank"><i
class="fab fa-github"></i></a>
              <a href=""><i class="icofont-linkedin"
target="_blank"></i></a>
            </div>
          </div>
        </div>
      </div>
```

```
      </div>
    </div>
  </section><!-- End Our Team Section -->
</main>

<?php include 'partials/_footer.php';?>

  <!-- Optional JavaScript -->
  <!-- jQuery first, then Popper.js, then Bootstrap JS -->
  <script src="https://code.jquery.com/jquery-3.4.1.min.js"
integrity="sha256-
CSXorXvZcTkaix6Yvo6HppcZGetbYMGWSFlBw8HfCJo="
crossorigin="anonymous"></script>
  <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.j
s" integrity="sha384-
Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvo
xMfooAo" crossorigin="anonymous"></script>
  <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.j
s" integrity="sha384-
wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3O
g8ifwB6" crossorigin="anonymous"></script>
  <script src="https://unpkg.com/bootstrap-show-
password@1.2.1/dist/bootstrap-show-password.min.js"></script>

  <!-- Vendor JS Files -->
  <script src="assets/vendor/jquery/jquery.min.js"></script>
```

```
  <script
src="assets/vendor/bootstrap/js/bootstrap.bundle.min.js"></script>
  <script src="assets/vendor/jquery-sticky/jquery.sticky.js"></script>
  <script
src="assets/vendor/waypoints/jquery.waypoints.min.js"></script>
  <script src="assets/vendor/counterup/counterup.min.js"></script>

  <!-- Template Main JS File -->
  <script src="assets/js/main.js"></script>


 </body>
</html>
```

CSS

```
/*===== GOOGLE FONTS =====*/
@import
url("https://fonts.googleapis.com/css2?family=Nunito:wght@400;600;700
&display=swap");

/*===== VARIABLES CSS =====*/
:root{
 --header-height: 3rem;
 --nav-width: 68px;

 /*===== Colors =====*/
 --first-color: #4723D9;
 --first-color-light: #AFA5D9;
```

```
  --white-color: #F7F6FB;

  /*===== Font and typography =====*/
  --body-font: 'Nunito', sans-serif;
  --normal-font-size: 1rem;

  /*===== z index =====*/
  --z-fixed: 100;
}

/*===== BASE =====*/
*,::before,::after{
  box-sizing: border-box;
}

body{
  position: relative;
  margin: var(--header-height) 0 0 0;
  padding: 0 1rem;
  font-family: var(--body-font);
  font-size: var(--normal-font-size);
  transition: .5s;
}

a{
  text-decoration: none;
}

/*===== HEADER =====*/
```

```
.header{
  width: 100%;
  height: var(--header-height);
  position: fixed;
  top: 0;
  left: 0;
  display: flex;
  align-items: center;
  justify-content: space-between;
  padding: 0 1rem;
  background-color: var(--white-color);
  z-index: var(--z-fixed);
  transition: .5s;
}

.header__toggle{
  color: var(--first-color);
  font-size: 1.5rem;
  cursor: pointer;
}

.header__img{
  width: 35px;
  height: 35px;
  display: flex;
  justify-content: center;
  border-radius: 50%;
  overflow: hidden;
}
```

```
.header__img img{
  width: 40px;
}

/*===== NAV =====*/
.l-navbar{
  position: fixed;
  top: 0;
  left: -30%;
  width: var(--nav-width);
  height: 100vh;
  background-color: var(--first-color);
  padding: .5rem 1rem 0 0;
  transition: .5s;
  z-index: var(--z-fixed);
}

.nav{
  height: 100%;
  display: flex;
  flex-direction: column;
  justify-content: space-between;
  overflow: hidden;
}

.nav__logo, .nav__link{
  display: grid;
  grid-template-columns: max-content max-content;
```

```
  align-items: center;
  column-gap: 1rem;
  padding: .5rem 0 .5rem 1.5rem;
}

.nav__logo{
  margin-bottom: 2rem;
}

.nav__logo-icon{
  font-size: 1.25rem;
  color: var(--white-color);
}

.nav__logo-name{
  color: var(--white-color);
  font-weight: 700;
}

.nav__link{
  position: relative;
  color: var(--first-color-light);
  margin-bottom: 1.5rem;
  transition: .3s;
}

.nav__link:hover{
  color: var(--white-color);
}
```

```
.nav__icon{
  font-size: 1.25rem;
}

/*Show navbar movil*/
.show{
  left: 0;
}

/*Add padding body movil*/
.body-pd{
  padding-left: calc(var(--nav-width) + 1rem);
}

/*Active links*/
.active{
  color: var(--white-color);
}

.active::before{
  content: '';
  position: absolute;
  left: 0;
  width: 2px;
  height: 32px;
  background-color: var(--white-color);
}
```

```
/* ===== MEDIA QUERIES=====*/
@media screen and (min-width: 768px){
 body{
  margin: calc(var(--header-height) + 1rem) 0 0 0;
  padding-left: calc(var(--nav-width) + 2rem);
 }

 .header{
  height: calc(var(--header-height) + 1rem);
  padding: 0 2rem 0 calc(var(--nav-width) + 2rem);
 }

 .header__img{
  width: 40px;
  height: 40px;
 }

 .header__img img{
  width: 45px;
 }

 .l-navbar{
  left: 0;
  padding: 1rem 1rem 0 0;
 }

 /*Show navbar desktop*/
 .showa{
  width: calc(var(--nav-width) + 156px);
```

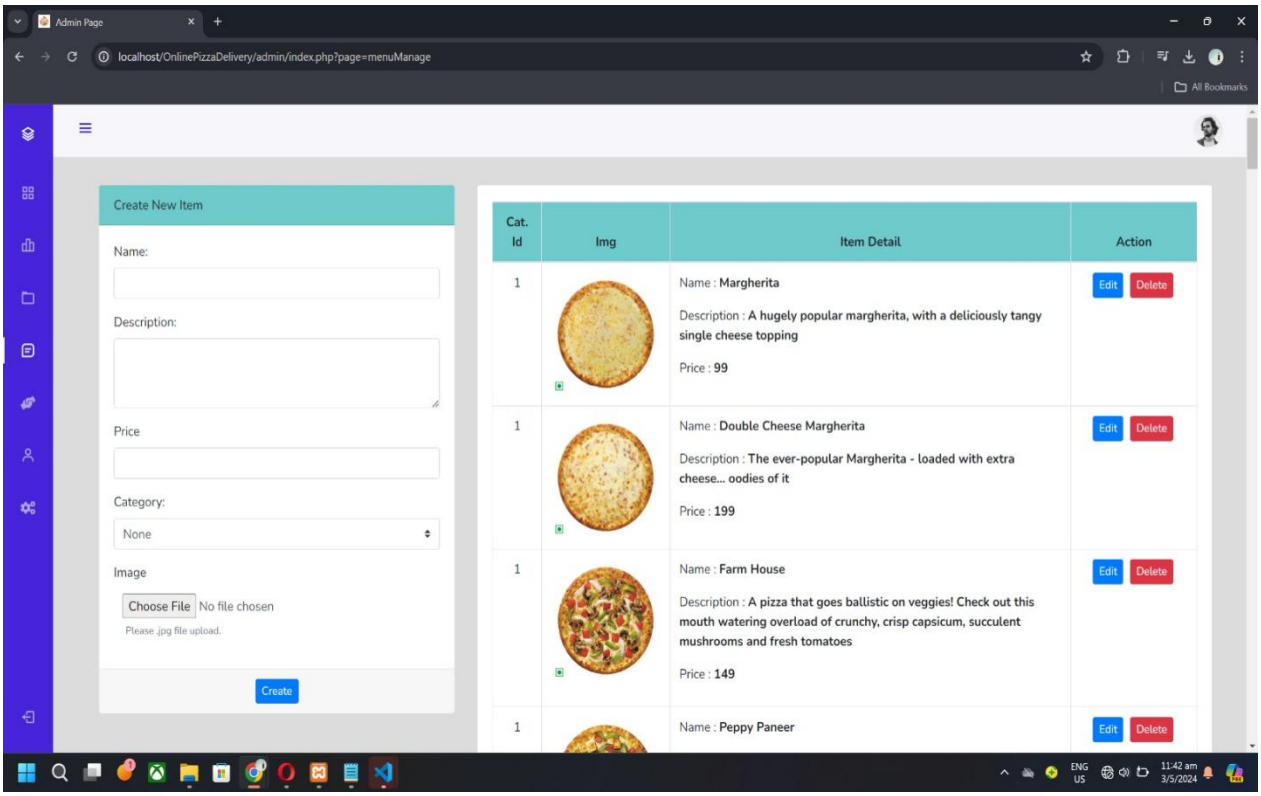
```
    }

    /*Add padding body desktop*/
    .body-pd{
      padding-left: calc(var(--nav-width) + 188px);
    }
  }
```

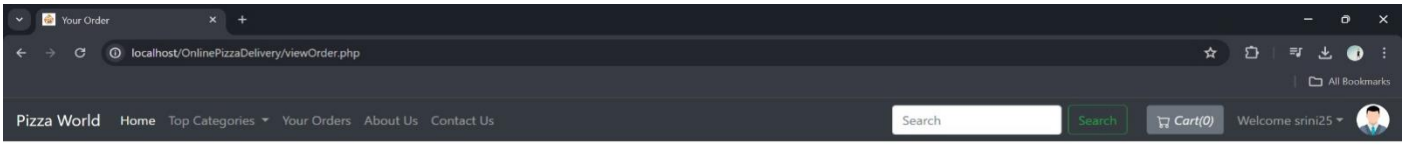
JS

```
/*===== SHOW NAVBAR  =====*/
const showNavbar = (toggleId, navId, bodyId, headerId) =>{
   const toggle = document.getElementById(toggleId),
   nav = document.getElementById(navId),
   bodypd = document.getElementById(bodyId),
   headerpd = document.getElementById(headerId)

   // Validate that all variables exist
   if(toggle && nav && bodypd && headerpd){
      toggle.addEventListener('click', ()=>{
         // show navbar
         nav.classList.toggle('showa')
         // change icon
         toggle.classList.toggle('bx-x')
         // add padding to body
         bodypd.classList.toggle('body-pd')
         // add padding to header
         headerpd.classList.toggle('body-pd')
      })
   }
}

showNavbar('header-toggle','nav-bar','body-pd','header')
```

7) Result and Discussion

The implementation of the online food ordering system in the database management system (DBMS) project yielded promising results, showcasing efficient data management and seamless user experience. Through the structured database design, we achieved optimized retrieval and storage of crucial information such as user profiles, menu items, orders, and delivery details. The system demonstrated robustness in handling concurrent user requests and maintaining data integrity, ensuring accurate order processing and timely delivery. Additionally, user feedback and performance metrics highlighted the system's responsiveness and usability, validating its effectiveness in meeting the intended objectives of simplifying food ordering processes and enhancing customer satisfaction. Further enhancements could focus on scalability for accommodating increased user demand and incorporating personalized recommendations to enrich the user experience.

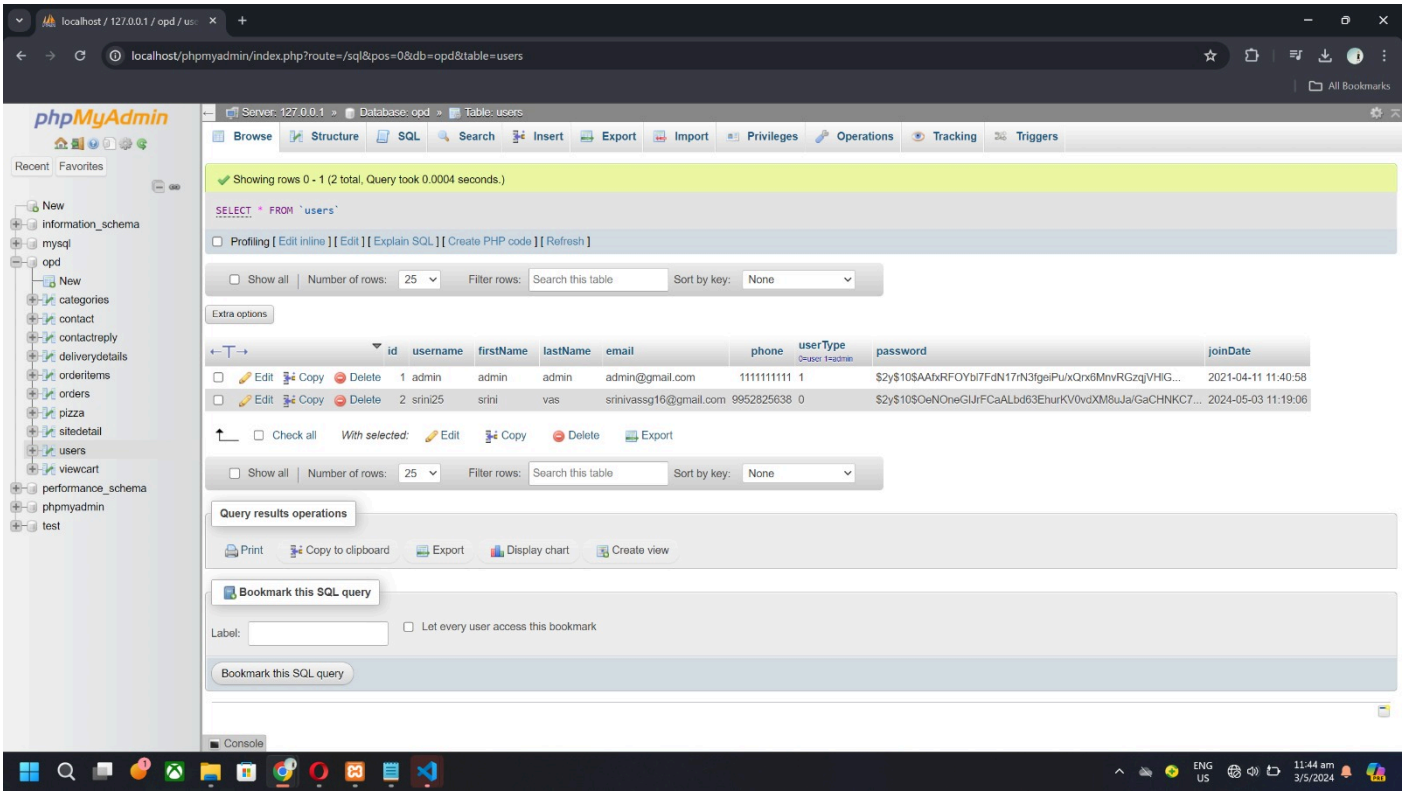


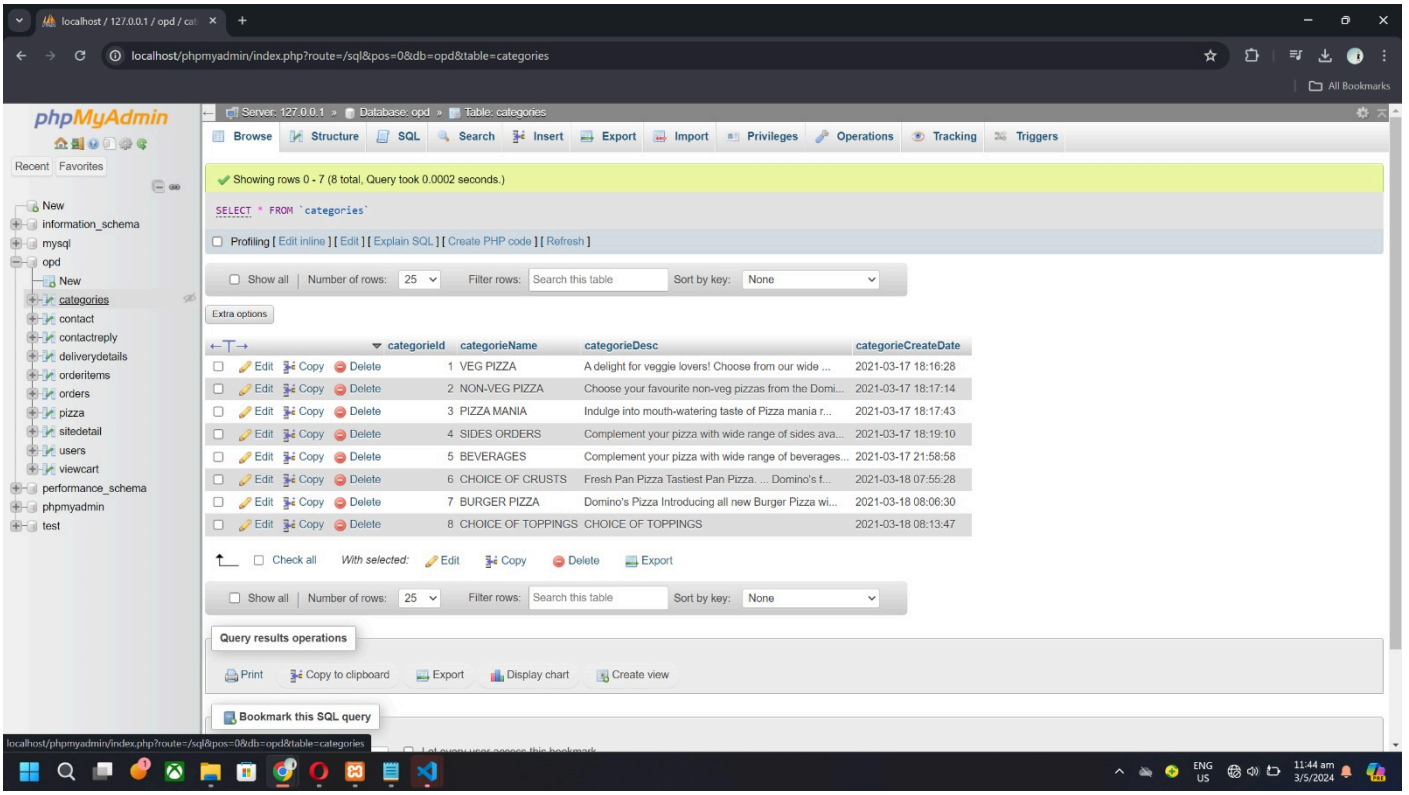


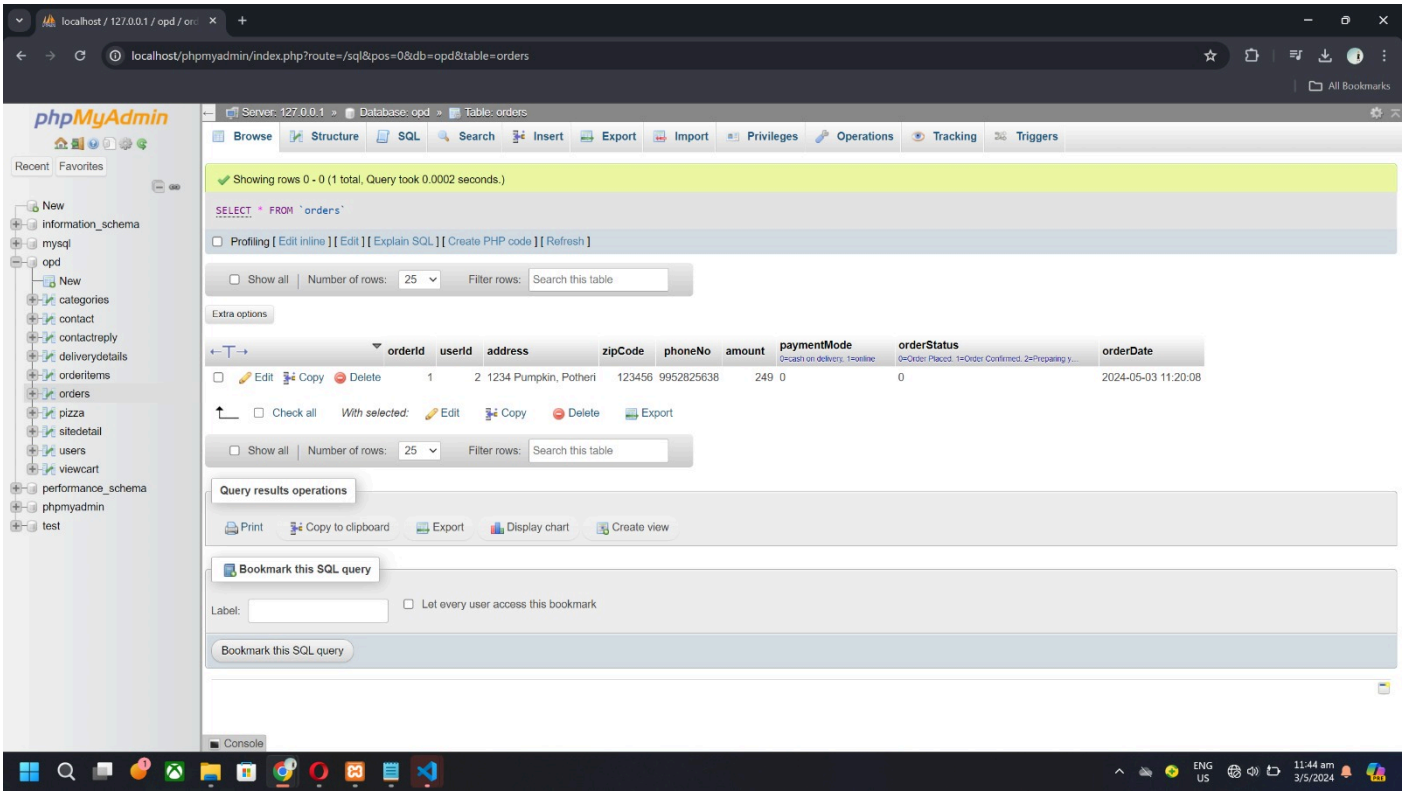
Order Details

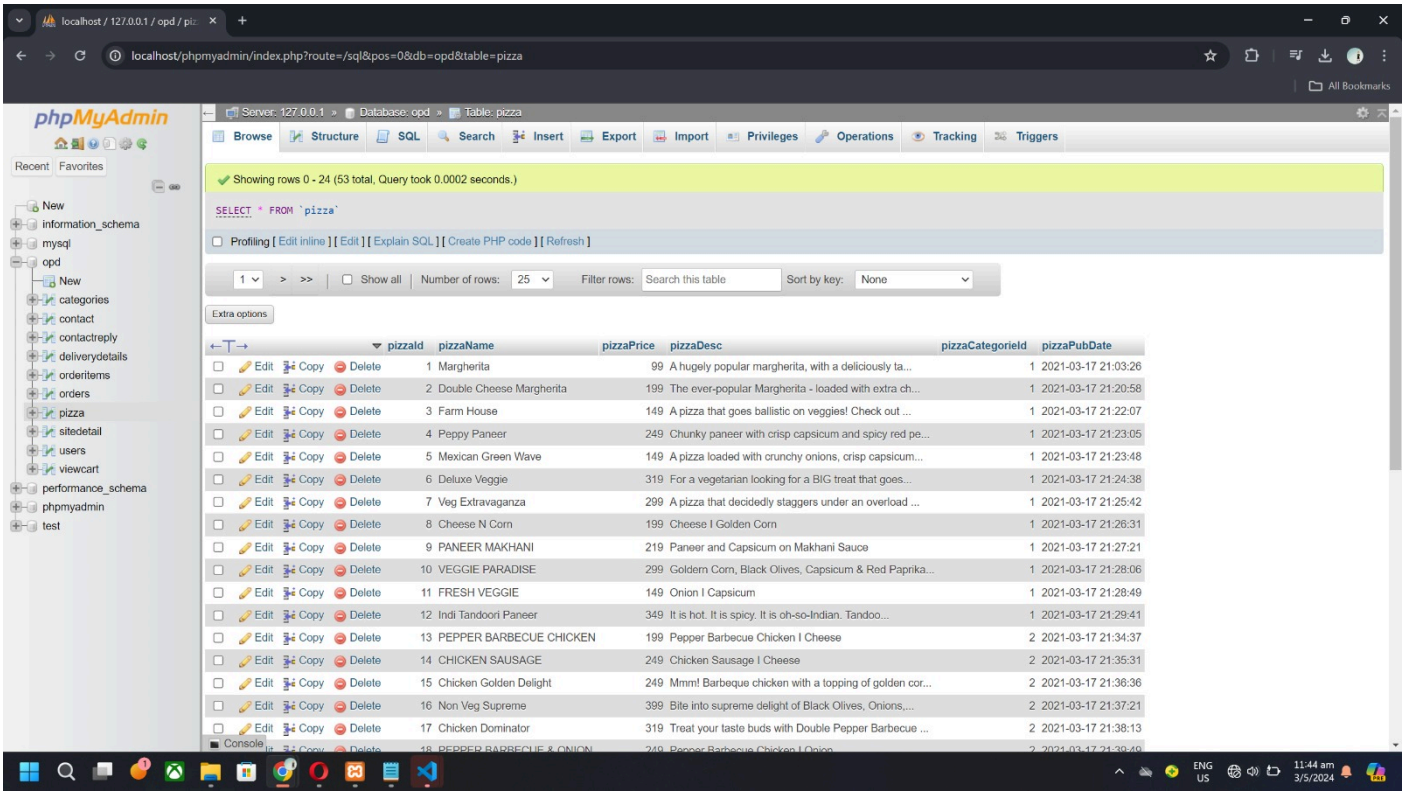
Print | Refresh List

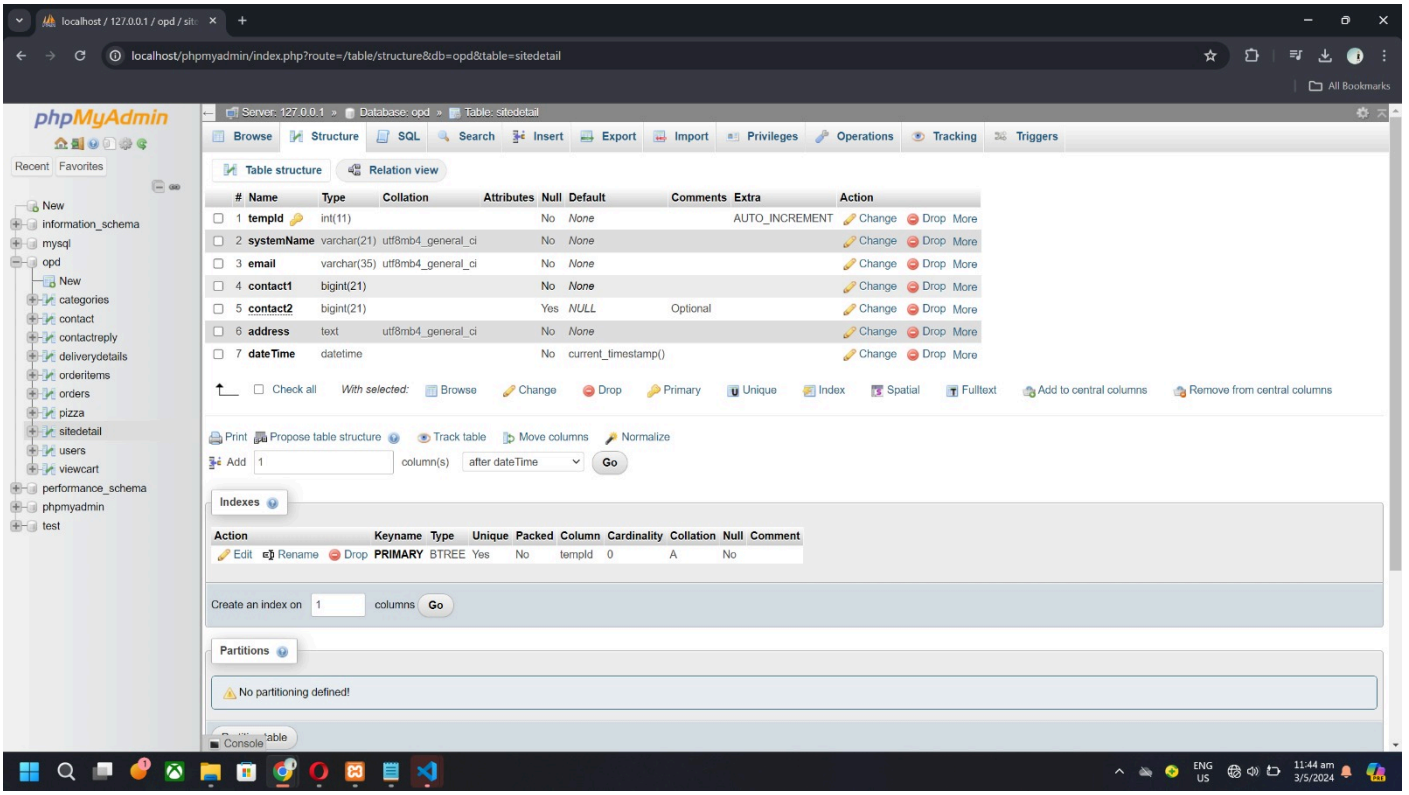
Order Id	Address	Phone No	Amount	Payment Mode	Order Date	Status	Items
1	1234 Pumpkin, Pother...	9952825638	249	Cash on Delivery	2024-05-03 11:20:08		

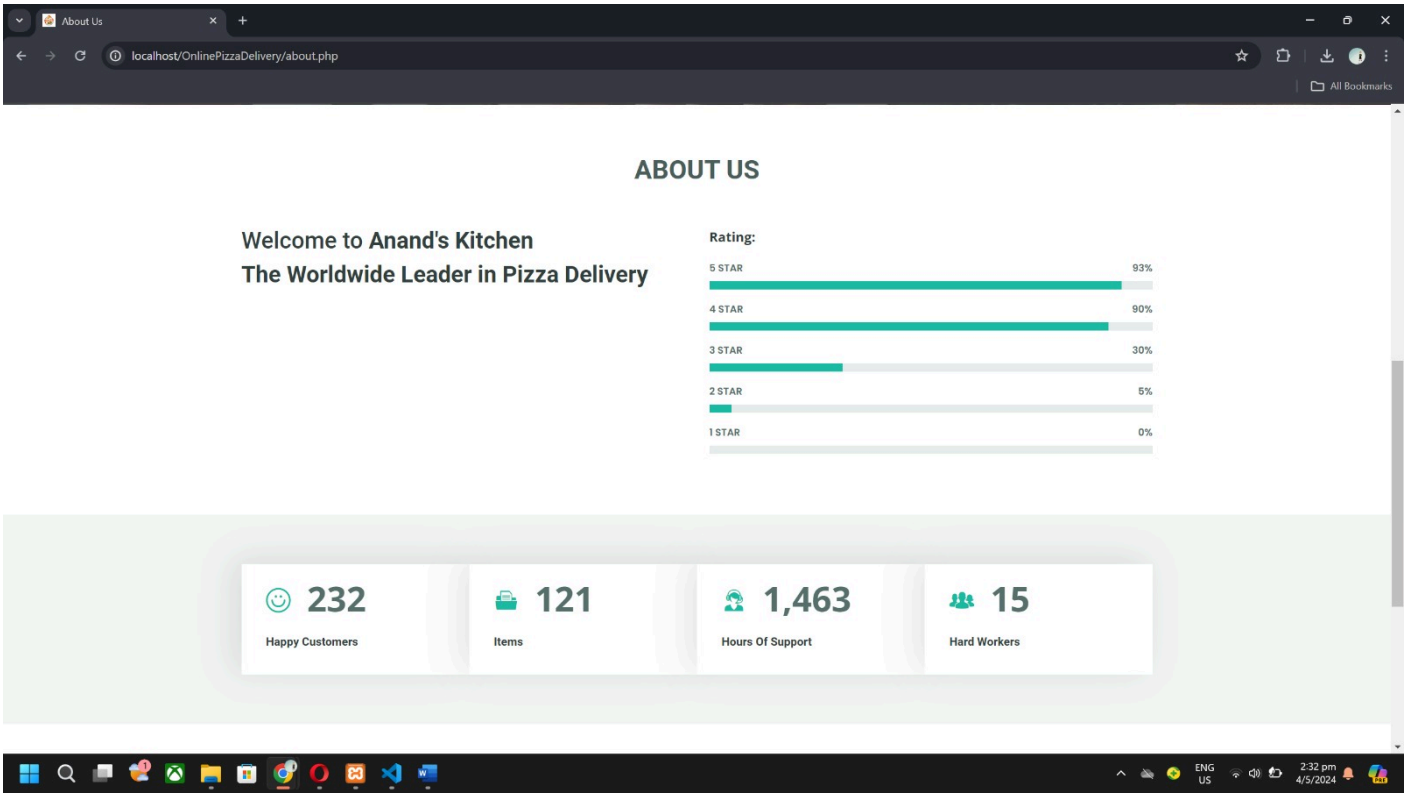


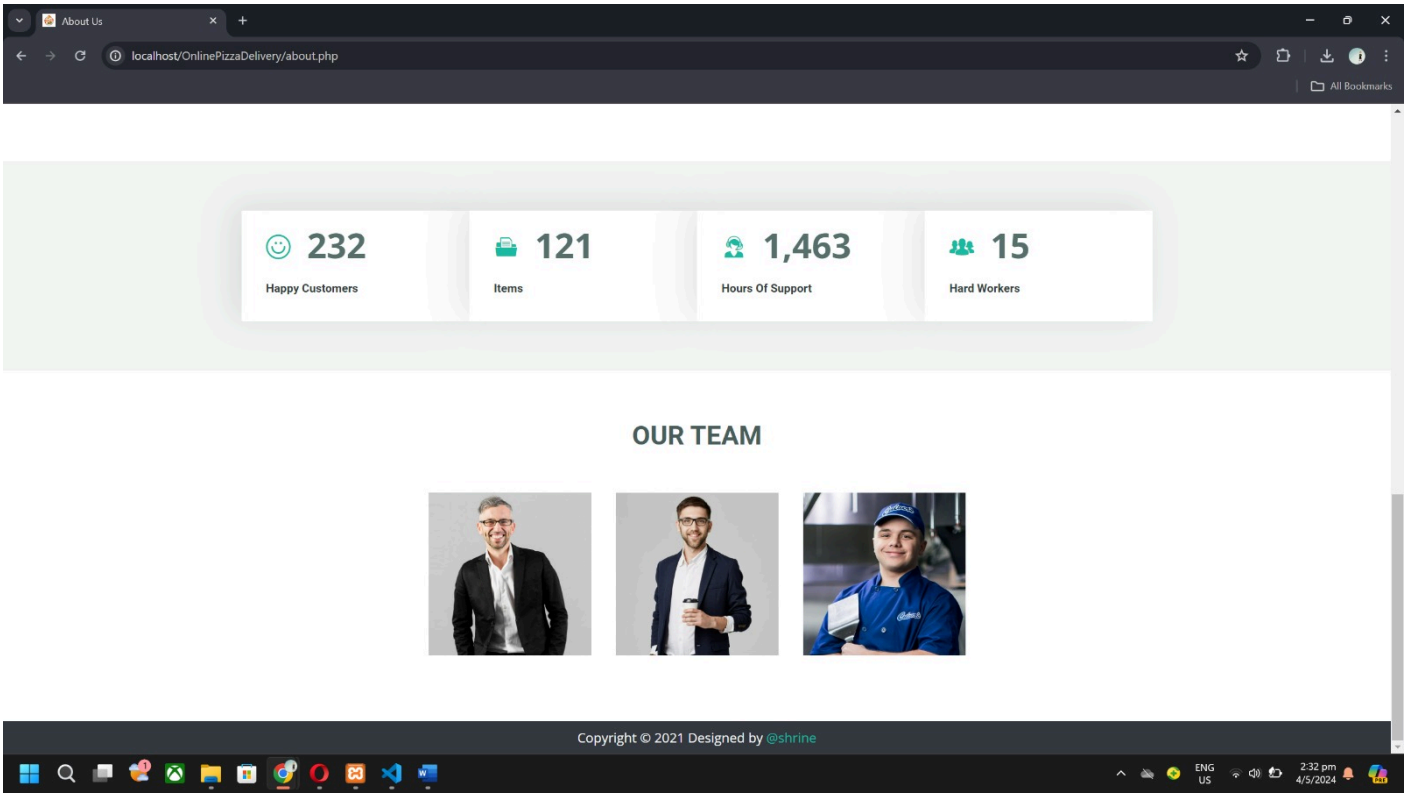

localhost/phpmyadmin/index.php?route=/sql&pos=0&db=opd&table=categories
phpMyAdmin
Server: 127.0.0.1 » Database: opd » Table: categories
Browse Structure SQL Search Insert Export Import Privileges Operations Tracking Triggers
Showing rows 0 - 7 (8 total, Query took 0.0002 seconds.)
SELECT * FROM `categories`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
Extra options
categorieId categorieName categorieDesc categorieCreateDate
1 VEG PIZZA A delight for veggie lovers! Choose from our wide ... 2021-03-17 18:16:28
2 NON-VEG PIZZA Choose your favourite non-veg pizzas from the Domi... 2021-03-17 18:17:14
3 PIZZA MANIA Indulge into mouth-watering taste of Pizza mania r... 2021-03-17 18:17:43
4 SIDES ORDERS Complement your pizza with wide range of sides ava... 2021-03-17 18:19:10
5 BEVERAGES Complement your pizza with wide range of beverages... 2021-03-17 21:58:58
6 CHOICE OF CRUSTS Fresh Pan Pizza Tastiest Pan Pizza. ... Domino's f... 2021-03-18 07:55:28
7 BURGER PIZZA Domino's Pizza Introducing all new Burger Pizza wi... 2021-03-18 08:06:30
8 CHOICE OF TOPPINGS CHOICE OF TOPPINGS 2021-03-18 08:13:47
Check all With selected: Edit Copy Delete Export
Query results operations
Print Copy to clipboard Export Display chart Create view
Bookmark this SQL query

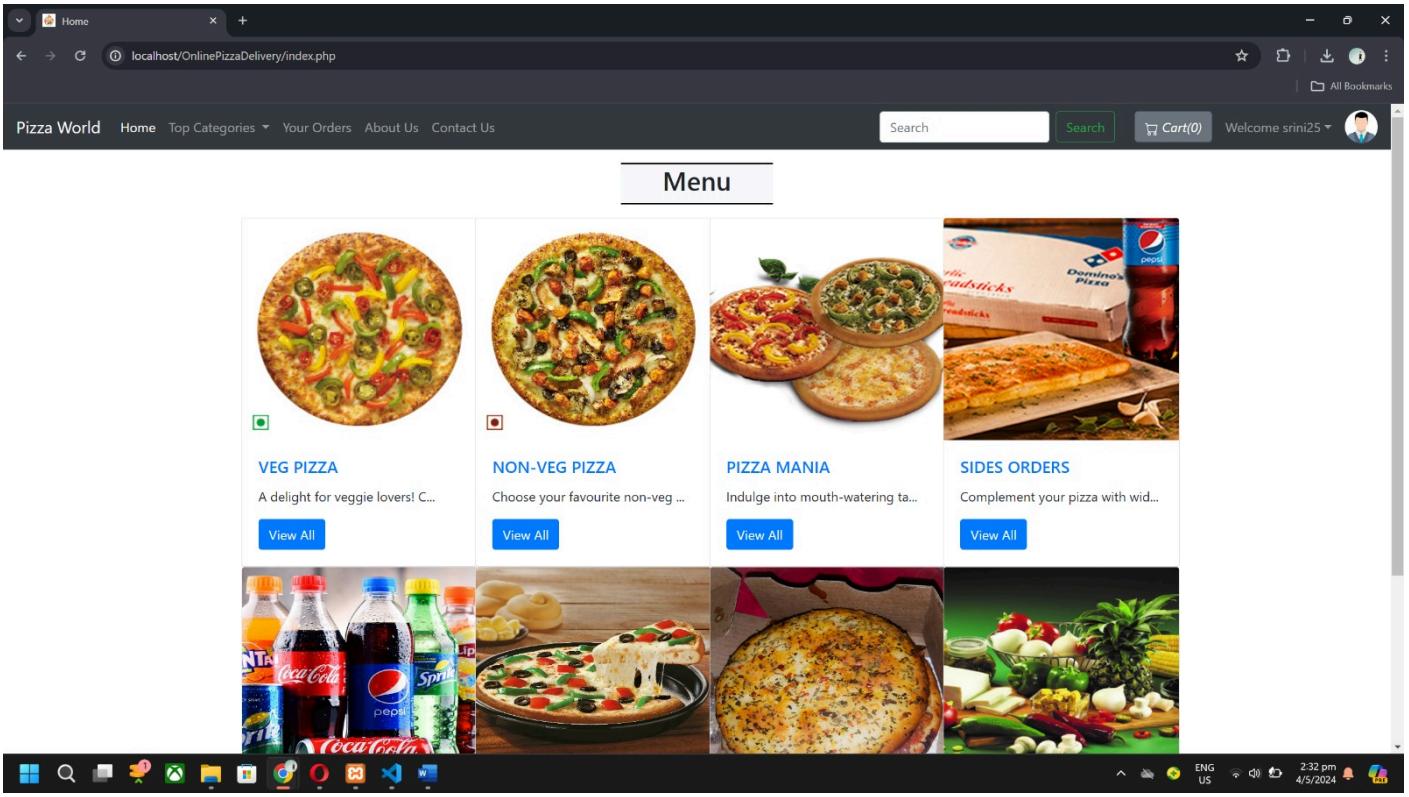

localhost/phpmyadmin/index.php?route=/sql&pos=0&db=opd&table=orders
phpMyAdmin
Server: 127.0.0.1 » Database: opd » Table: orders
Browse Structure SQL Search Insert Export Import Privileges Operations Tracking Triggers
Showing rows 0 - 0 (1 total, Query took 0.0002 seconds.)
SELECT * FROM `orders`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all | Number of rows: 25 Filter rows: Search this table
Extra options
orderId userId address zipCode phoneNo amount paymentMode orderStatus orderDate
1 2 1234 Pumpkin, Potheri 123456 9952825638 249 0 0 2024-05-03 11:20:08
Check all With selected: Edit Copy Delete Export
Query results operations
Print Copy to clipboard Export Display chart Create view
Bookmark this SQL query
Label: Let every user access this bookmark
Bookmark this SQL query
Console

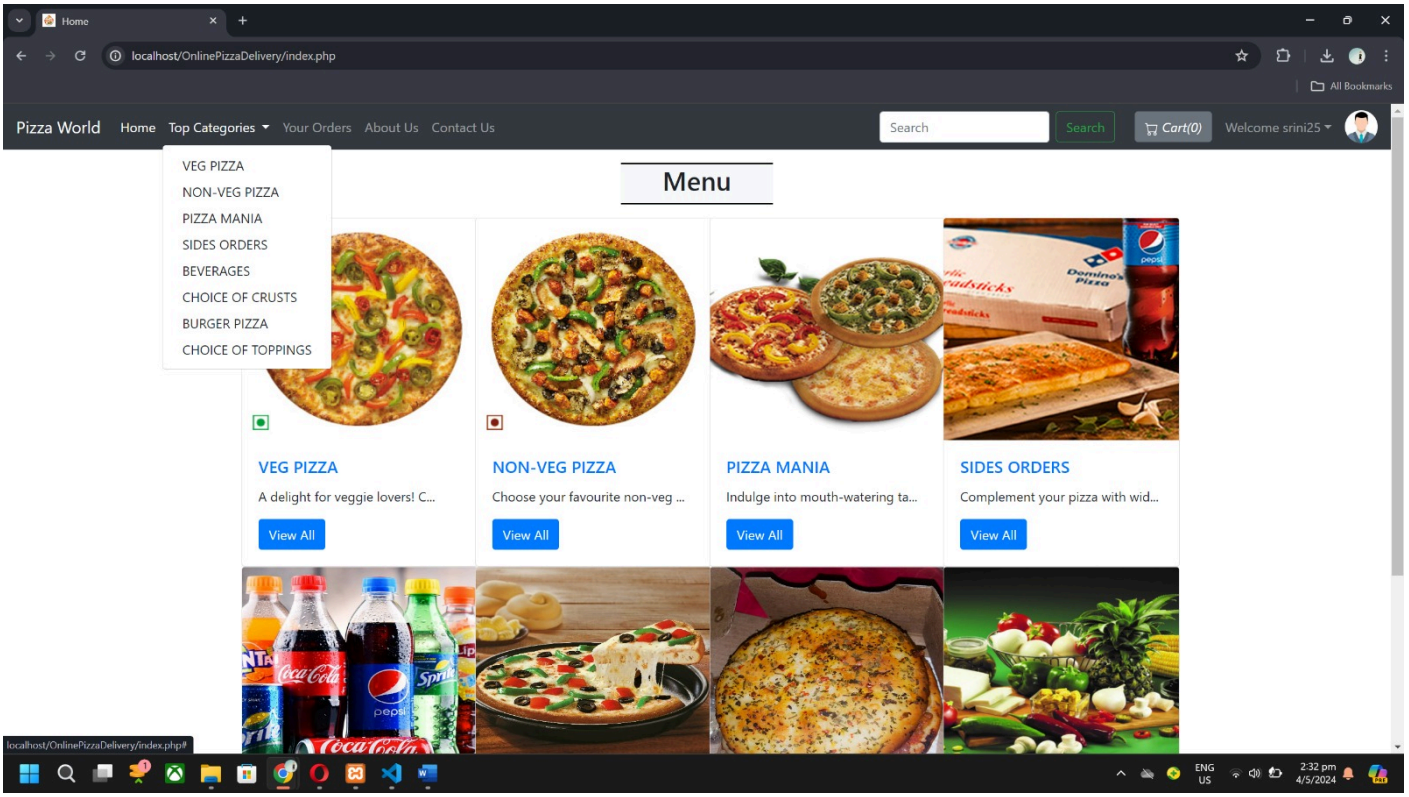

localhost / 127.0.0.1 / opd / pi
localhost/phpmyadmin/index.php?route=/sql&pos=0&db=opd&table=pizza
phpMyAdmin
Recent Favorites
New
information_schema
mysql
opd
New
categories
contact
contactreply
deliverydetails
orderitems
orders
pizza
sitedetail
users
viewcart
performance_schema
phpmyadmin
test
Server: 127.0.0.1 » Database: opd » Table: pizza
Browse Structure SQL Search Insert Export Import Privileges Operations Tracking Triggers
Showing rows 0 - 24 (53 total, Query took 0.0002 seconds.)
SELECT * FROM `pizza`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all Number of rows: 25 Filter rows: Search this table Sort by key: None
Extra options
pizzaId pizzaName pizzaPrice pizzaDesc pizzaCategoryId pizzaPubDate
1 Margherita 99 A hugely popular margherita, with a deliciously ta... 1 2021-03-17 21:03:26
2 Double Cheese Margherita 199 The ever-popular Margherita - loaded with extra ch... 1 2021-03-17 21:20:58
3 Farm House 149 A pizza that goes ballistic on veggies! Check out ... 1 2021-03-17 21:22:07
4 Peppy Paneer 249 Chunky paneer with crisp capsicum and spicy red pe... 1 2021-03-17 21:23:05
5 Mexican Green Wave 149 A pizza loaded with crunchy onions, crisp capsicum... 1 2021-03-17 21:23:48
6 Deluxe Veggie 319 For a vegetarian looking for a BIG treat that goes... 1 2021-03-17 21:24:38
7 Veg Extravaganza 299 A pizza that decidedly staggers under an overload ... 1 2021-03-17 21:25:42
8 Cheese N Corn 199 Cheese I Golden Corn 1 2021-03-17 21:26:31
9 PANEER MAKHANI 219 Paneer and Capsicum on Makhani Sauce 1 2021-03-17 21:27:21
10 VEGGIE PARADISE 299 Golden Corn, Black Olives, Capsicum & Red Paprika... 1 2021-03-17 21:28:06
11 FRESH VEGGIE 149 Onion I Capsicum 1 2021-03-17 21:28:49
12 Indi Tandoori Paneer 349 It is hot. It is spicy. It is oh-so-Indian. Tandoo... 1 2021-03-17 21:29:41
13 PEPPER BARBECUE CHICKEN 199 Pepper Barbecue Chicken I Cheese 2 2021-03-17 21:34:37
14 CHICKEN SAUSAGE 249 Chicken Sausage I Cheese 2 2021-03-17 21:35:31
15 Chicken Golden Delight 249 Mmm! Barbeque chicken with a topping of golden cor... 2 2021-03-17 21:36:36
16 Non Veg Supreme 399 Bite into supreme delight of Black Olives, Onions,... 2 2021-03-17 21:37:21
17 Chicken Dominator 319 Treat your taste buds with Double Pepper Barbecue ... 2 2021-03-17 21:38:13
Console
ENG US 11:44 am 3/5/2024

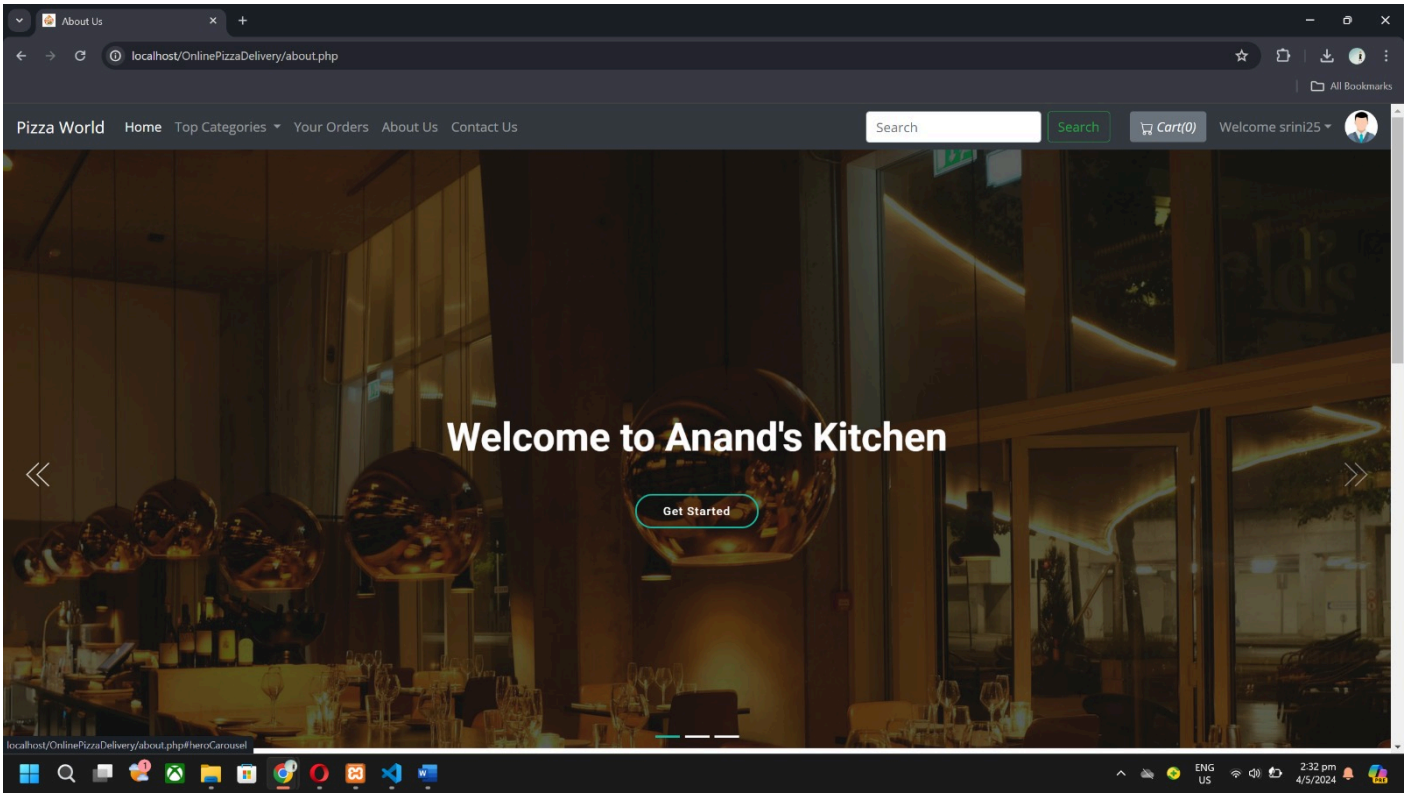

localhost / 127.0.0.1 / opd / sit
localhost/phpmyadmin/index.php?route=/table/structure&db=opd&table=sitedetail
phpMyAdmin
Recent Favorites
New
information_schema
mysql
opd
New
categories
contact
contactreply
deliverydetails
orderitems
orders
pizza
sitedetail
users
viewcart
performance_schema
phpmyadmin
test
Server: 127.0.0.1 » Database: opd » Table: sitedetail
Browse Structure SQL Search Insert Export Import Privileges Operations Tracking Triggers
Table structure Relation view
Name Type Collation Attributes Null Default Comments Extra Action
1 tempId int(11) No None AUTO_INCREMENT Change Drop More
2 systemName varchar(21) utf8mb4_general_ci No None Change Drop More
3 email varchar(35) utf8mb4_general_ci No None Change Drop More
4 contact1 bigint(21) No None Change Drop More
5 contact2 bigint(21) Yes NULL Optional Change Drop More
6 address text utf8mb4_general_ci No None Change Drop More
7 dateTime datetime No current_timestamp() Change Drop More
Check all With selected: Browse Change Drop Primary Unique Index Spatial Fulltext Add to central columns Remove from central columns
Print Propose table structure Track table Move columns Normalize
Add 1 column(s) after dateTime Go
Indexes
Action Keyname Type Unique Packed Column Cardinality Collation Null Comment
Edit Rename Drop PRIMARY BTREE Yes No tempId 0 A No
Create an index on 1 columns Go
Partitions
No partitioning defined!
Console
ENG US 11:44 am 3/5/2024

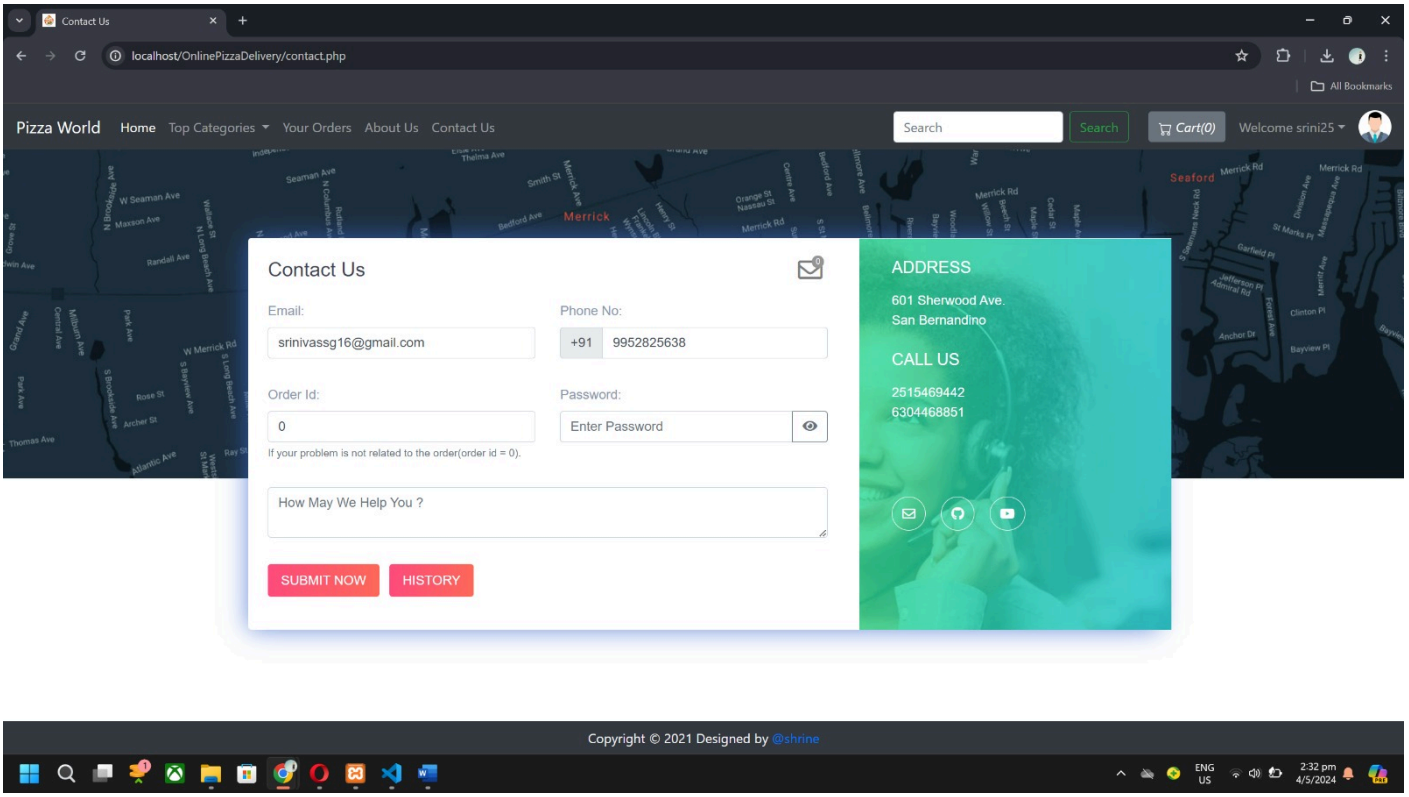

About Us
localhost/OnlinePizzaDelivery/about.php
All Bookmarks
ABOUT US
Welcome to Anand's Kitchen
The Worldwide Leader in Pizza Delivery
Rating:
5 STAR 93%
4 STAR 90%
3 STAR 30%
2 STAR 5%
1 STAR 0%
232
Happy Customers
121
Items
1,463
Hours Of Support
15
Hard Workers


About Us
localhost/OnlinePizzaDelivery/about.php
All Bookmarks
232
Happy Customers
121
Items
1,463
Hours Of Support
15
Hard Workers
OUR TEAM
Copyright © 2021 Designed by @shrine

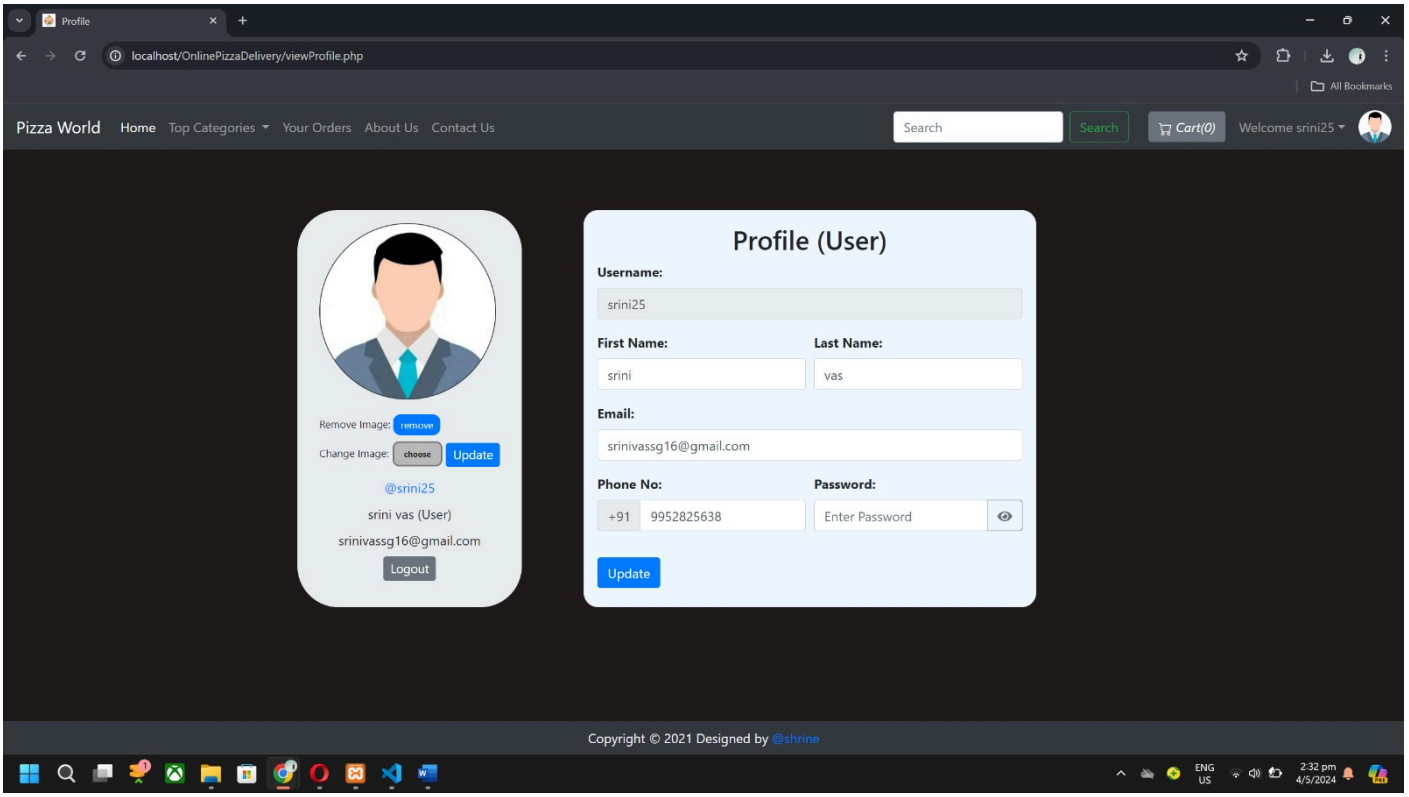

Home
localhost/OnlinePizzaDelivery/index.php
Pizza World
Home
Top Categories
Your Orders
About Us
Contact Us
Search
Search
Cart(0)
Welcome srini25
Menu
VEG PIZZA
A delight for veggie lovers! C...
View All
NON-VEG PIZZA
Choose your favourite non-veg ...
View All
PIZZA MANIA
Indulge into mouth-watering ta...
View All
SIDES ORDERS
Complement your pizza with wid...
View All

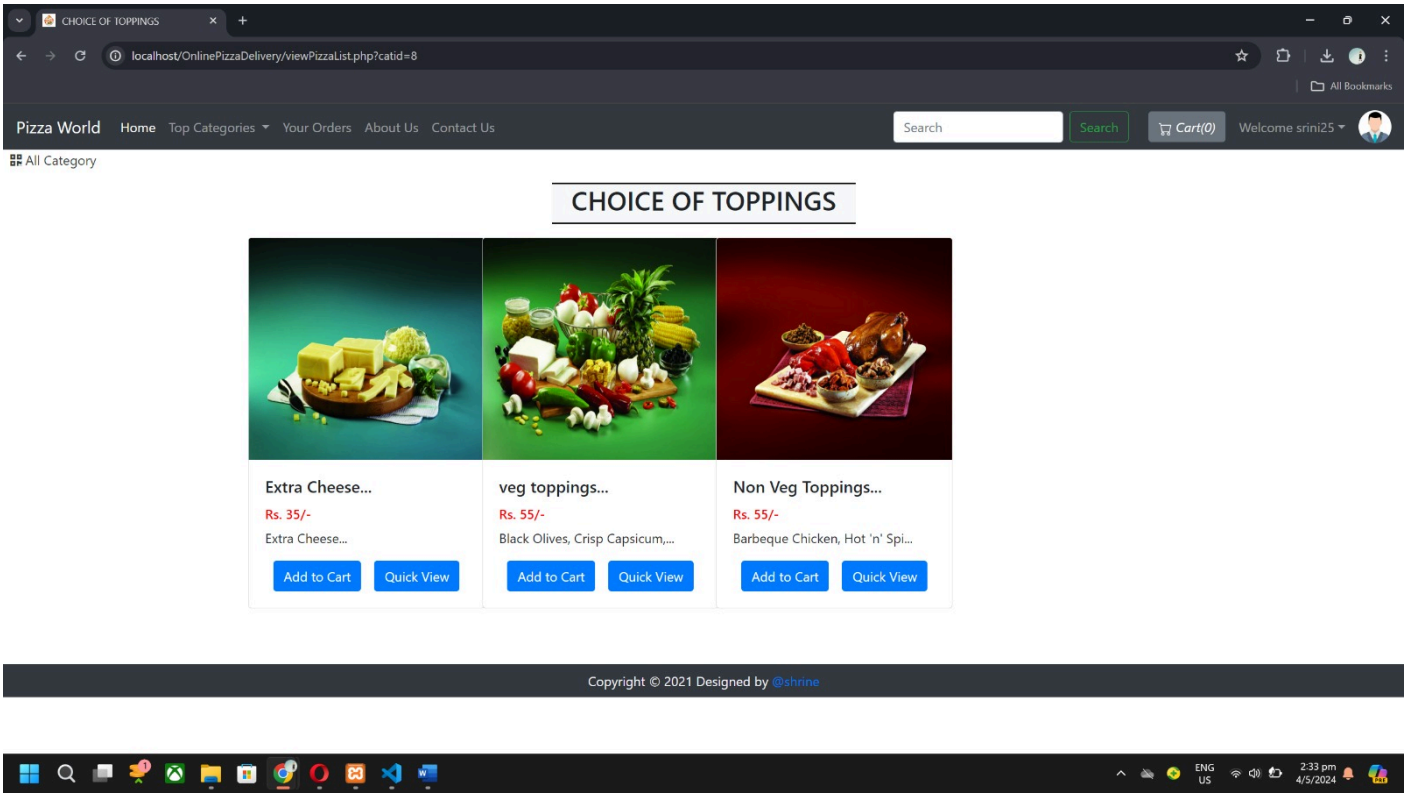

Home
localhost/OnlinePizzaDelivery/index.php
Pizza World
Home
Top Categories
Your Orders
About Us
Contact Us
Search
Search
Cart(0)
Welcome srini25
VEG PIZZA
NON-VEG PIZZA
PIZZA MANIA
SIDES ORDERS
BEVERAGES
CHOICE OF CRUSTS
BURGER PIZZA
CHOICE OF TOPPINGS
Menu
VEG PIZZA
A delight for veggie lovers! C...
View All
NON-VEG PIZZA
Choose your favourite non-veg ...
View All
PIZZA MANIA
Indulge into mouth-watering ta...
View All
SIDES ORDERS
Complement your pizza with wid...
View All

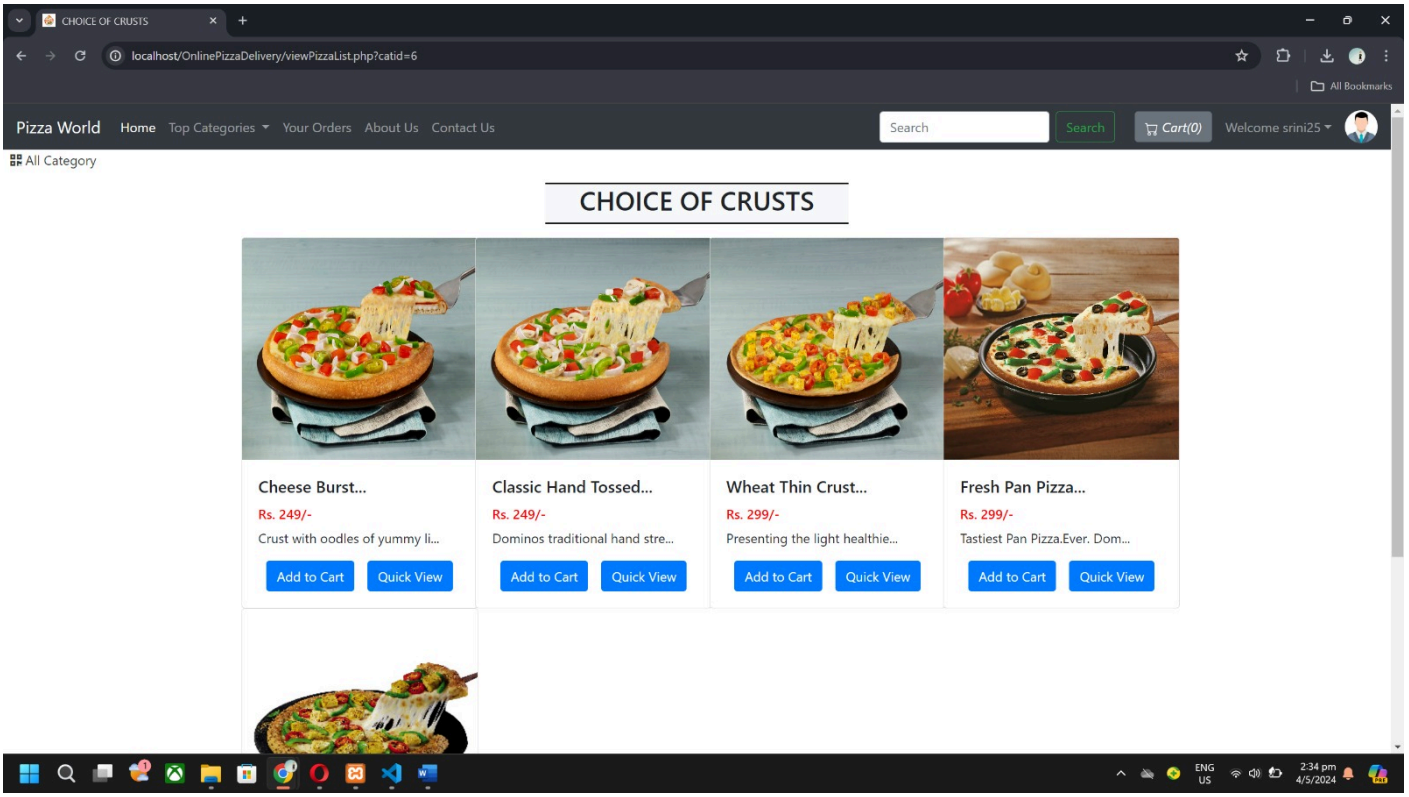

About Us
localhost/OnlinePizzaDelivery/about.php
All Bookmarks
Pizza World
Home
Top Categories
Your Orders
About Us
Contact Us
Search
Search
Cart(0)
Welcome srini25
Welcome to Anand's Kitchen
Get Started
localhost/OnlinePizzaDelivery/about.php#heroCarousel
ENG
US
2:32 pm
4/5/2024

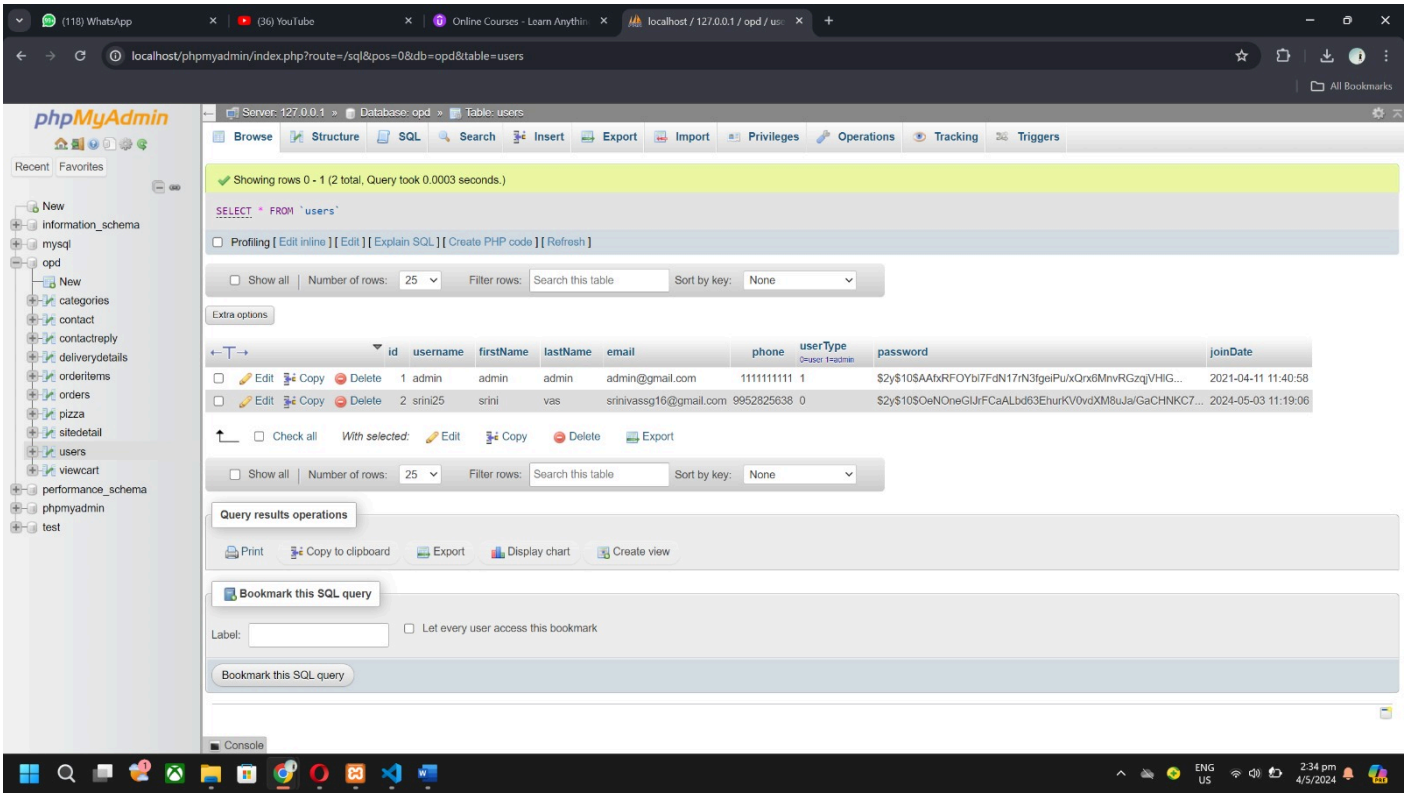

Contact Us
localhost/OnlinePizzaDelivery/contact.php
All Bookmarks
Pizza World
Home
Top Categories
Your Orders
About Us
Contact Us
Search
Search
Cart(0)
Welcome srini25
Contact Us
Email:
srinivassg16@gmail.com
Phone No:
+91
9952825638
Order Id:
0
If your problem is not related to the order(order id = 0).
Password:
Enter Password
How May We Help You ?
SUBMIT NOW
HISTORY
ADDRESS
601 Sherwood Ave.
San Bernandino
CALL US
2515469442
6304468851

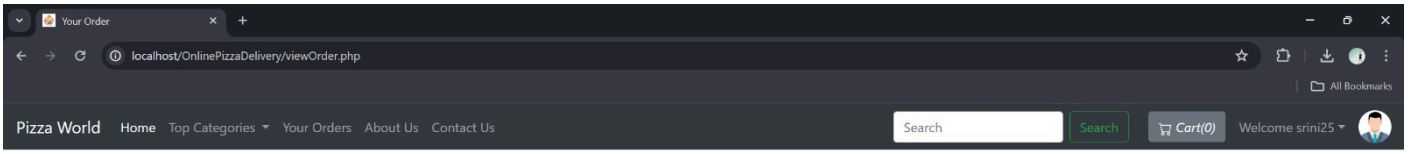
Copyright © 2021 Designed by @shrine
ENG
US
2:32 pm
4/5/2024


Profile
localhost/OnlinePizzaDelivery/viewProfile.php
Pizza World
Home
Top Categories
Your Orders
About Us
Contact Us
Search
Search
Cart(0)
Welcome srini25
Remove Image:
remove
Change Image:
choose
Update
@srini25
srini vas (User)
srinivassg16@gmail.com
Logout
Profile (User)
Username:
srini25
First Name:
srini
Last Name:
vas
Email:
srinivassg16@gmail.com
Phone No:
+91
9952825638
Password:
Enter Password
Update
Copyright © 2021 Designed by @shrine


CHOICE OF TOPPINGS
localhost/OnlinePizzaDelivery/viewPizzaList.php?catid=8
Pizza World
Home
Top Categories
Your Orders
About Us
Contact Us
Search
Search
Cart(0)
Welcome srini25
All Category
CHOICE OF TOPPINGS
Extra Cheese...
Rs. 35/-
Extra Cheese...
Add to Cart
Quick View
veg toppings...
Rs. 55/-
Black Olives, Crisp Capsicum,...
Add to Cart
Quick View
Non Veg Toppings...
Rs. 55/-
Barbeque Chicken, Hot 'n' Spi...
Add to Cart
Quick View
Copyright © 2021 Designed by @shrine


CHOICE OF CRUSTS
localhost/OnlinePizzaDelivery/viewPizzaList.php?catid=6
Pizza World
Home
Top Categories
Your Orders
About Us
Contact Us
Search
Search
Cart(0)
Welcome srini25
All Category
CHOICE OF CRUSTS
Cheese Burst...
Rs. 249/-
Crust with oodles of yummy li...
Add to Cart
Quick View
Classic Hand Tossed...
Rs. 249/-
Dominos traditional hand stre...
Add to Cart
Quick View
Wheat Thin Crust...
Rs. 299/-
Presenting the light healthie...
Add to Cart
Quick View
Fresh Pan Pizza...
Rs. 299/-
Tastiest Pan Pizza.Ever. Dom...
Add to Cart
Quick View

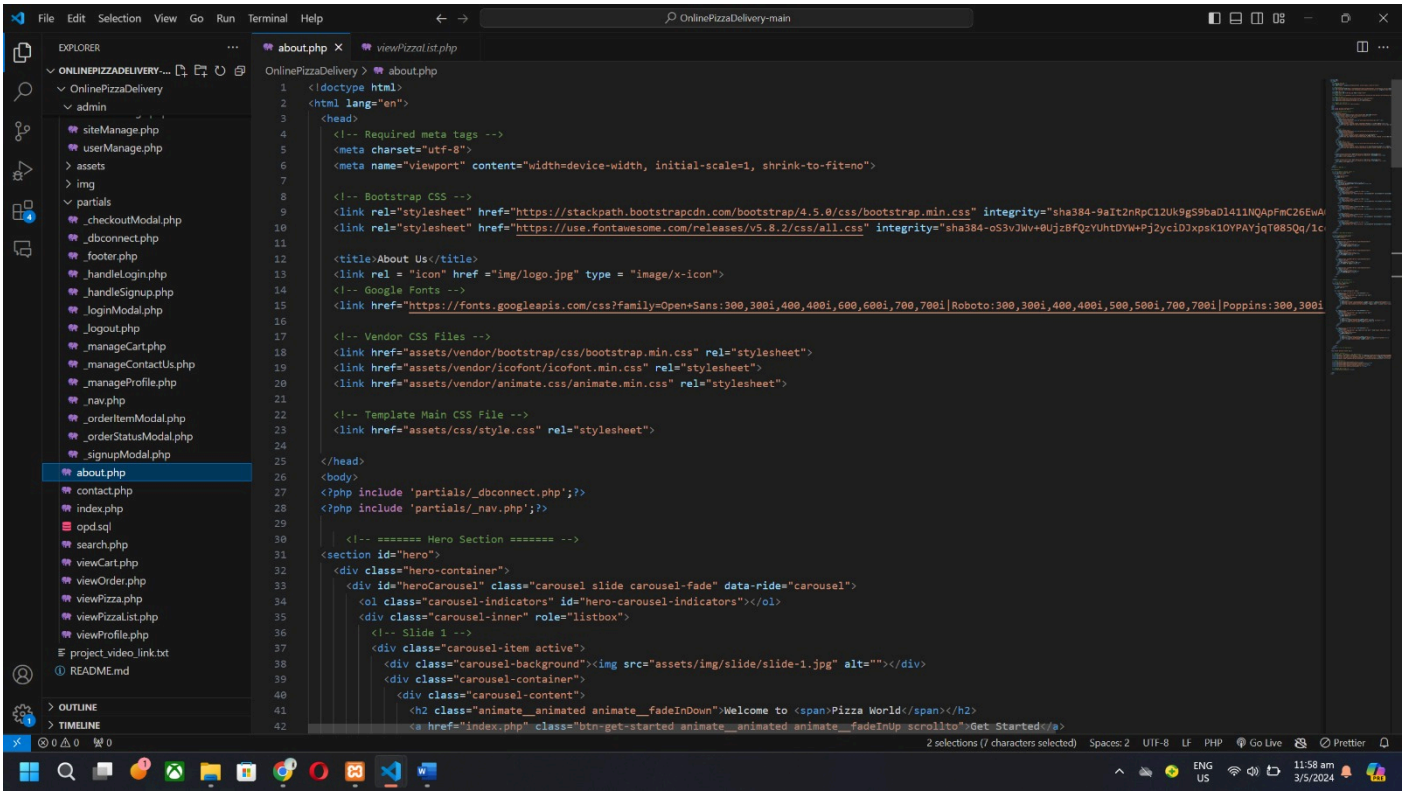

localhost/phpmyadmin/index.php?route=/sql&pos=0&db=opd&table=users
phpMyAdmin
Server: 127.0.0.1 » Database: opd » Table: users
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Tracking
Triggers
Showing rows 0 - 1 (2 total, Query took 0.0003 seconds.)
SELECT * FROM `users`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None
Extra options
id username firstName lastName email phone userType password joinDate
1 admin admin admin admin@gmail.com 1111111111 1 $2y$10$AAfxRFOYbl7FdN17rN3fgeiPu/xQrx6MnvRGzqjVHlG... 2021-04-11 11:40:58
2 srini25 srini vas srinivassg16@gmail.com 9952825638 0 $2y$10$OeNOneGlJrFCaALbd63EhurKV0vdXM8uJa/GaCHNKC7... 2024-05-03 11:19:06
Check all With selected: Edit Copy Delete Export
Query results operations
Print
Copy to clipboard
Export
Display chart
Create view
Bookmark this SQL query
Label:
Let every user access this bookmark
Bookmark this SQL query
Console

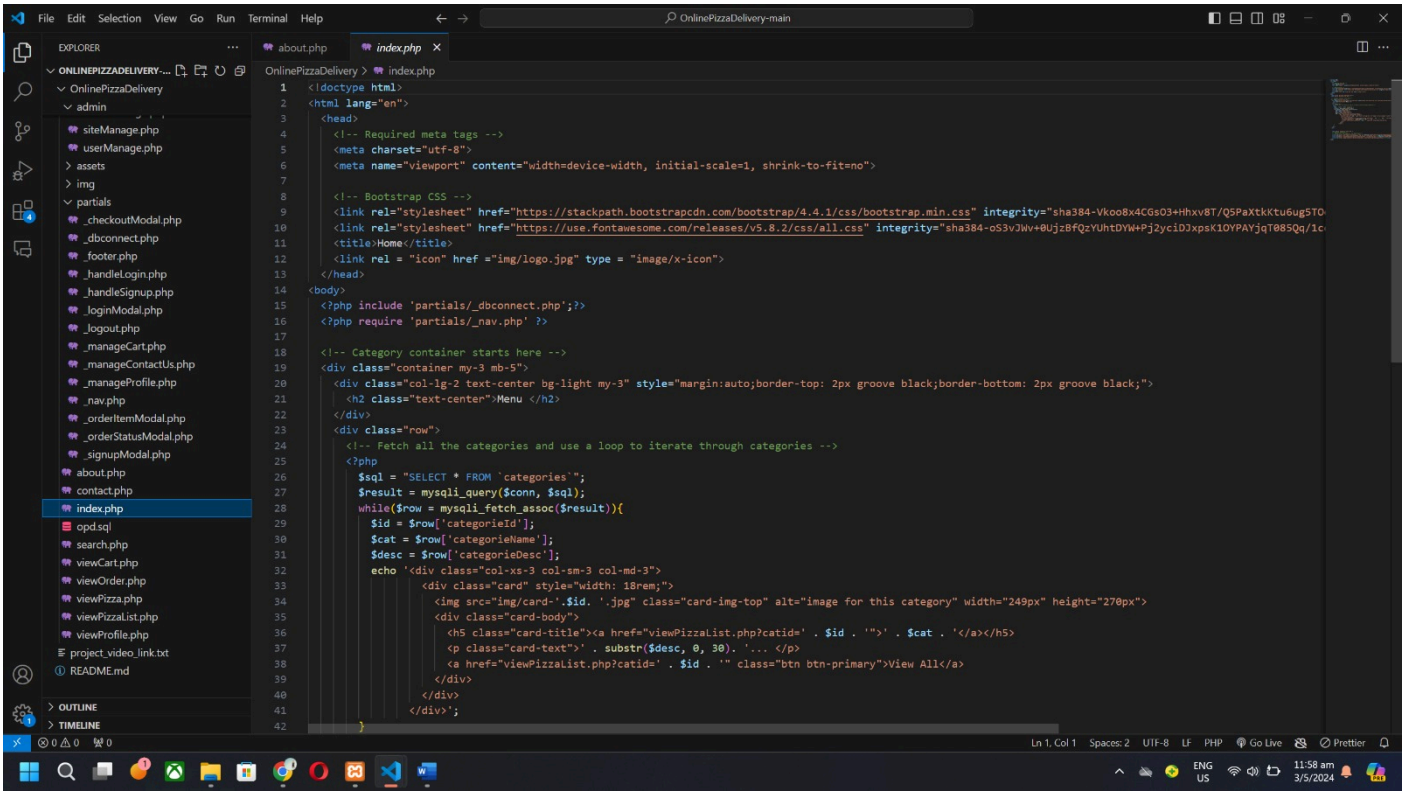


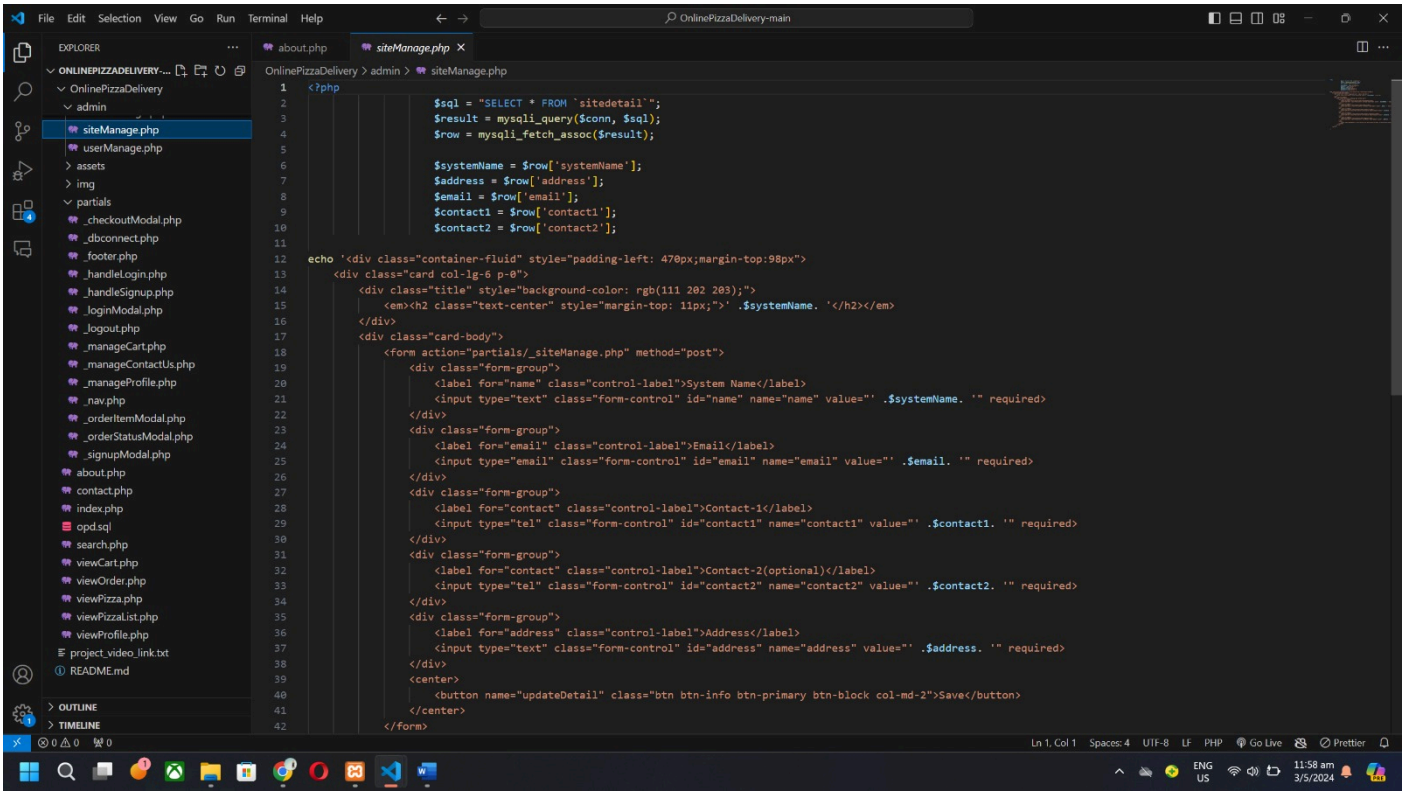
Order Details

Print | Refresh List

Order Id	Address	Phone No	Amount	Payment Mode	Order Date	Status	Items
1	1234 Pumpkin, Pother...	9952825638	249	Cash on Delivery	2024-05-03 11:20:08		
2	1234 bel lab, srmist...	9952825638	199	Cash on Delivery	2024-05-04 13:56:10		




File Edit Selection View Go Run Terminal Help
OnlinePizzaDelivery-main
EXPLORER
about.php
index.php
OnlinePizzaDelivery > index.php
ONLINEPIZZADELIVERY-...
OnlinePizzaDelivery
admin
siteManage.php
userManage.php
assets
img
partials
_checkoutModal.php
_dbconnect.php
_footer.php
_handleLogin.php
_handleSignup.php
_loginModal.php
_logout.php
_manageCart.php
_manageContactUs.php
_manageProfile.php
_nav.php
_orderItemModal.php
_orderStatusModal.php
_signupModal.php
about.php
contact.php
index.php
opd.sql
search.php
viewCart.php
viewOrder.php
viewPizza.php
viewPizzaList.php
viewProfile.php
project_video_link.txt
README.md
OUTLINE
TIMELINE
Ln 1, Col 1 Spaces: 2 UTF-8 LF PHP Go Live Prettier
ENG US
11:58 am 3/5/2024


File Edit Selection View Go Run Terminal Help
OnlinePizzaDelivery-main
EXPLORER
about.php
siteManage.php
OnlinePizzaDelivery > admin > siteManage.php
ONLINEPIZZADELIVERY-...
OnlinePizzaDelivery
admin
siteManage.php
userManage.php
assets
img
partials
_checkoutModal.php
_dbconnect.php
_footer.php
_handleLogin.php
_handleSignup.php
_loginModal.php
_logout.php
_manageCart.php
_manageContactUs.php
_manageProfile.php
_nav.php
_orderItemModal.php
_orderStatusModal.php
_signupModal.php
about.php
contact.php
index.php
opd.sql
search.php
viewCart.php
viewOrder.php
viewPizza.php
viewPizzaList.php
viewProfile.php
project_video_link.txt
README.md
OUTLINE
TIMELINE
Ln 1, Col 1 Spaces: 4 UTF-8 LF PHP Go Live Prettier
ENG US
11:58 am 3/5/2024

8) Online Course Certification

CERTIFICATE
OF EXCELLENCE

THIS CERTIFICATE IS AWARDED TO

SCALER Topics

Siddharth Gaur

In recognition of the completion of the tutorial: **DBMS Course - Master the Fundamentals and Advanced Concepts**

Following are the the learning items, which are covered in this tutorial

74 Video Tutorials 16 Modules 16 Challenges 19 April 2024

Anshuman Singh
Co-founder SCALER

CERTIFICATE OF EXCELLENCE BY SCALER

CERTIFICATE OF EXCELLENCE

THIS CERTIFICATE IS AWARDED TO

SCALER Topics

ANISH PRANAV (RA2211003011624)

In recognition of the completion of the tutorial: **DBMS Course - Master the Fundamentals and Advanced Concepts**

Following are the the learning items, which are covered in this tutorial

74 Video Tutorials 16 Modules 16 Challenges 19 April 2024

Anshuman Singh

Co-founder **SCALER**

CERTIFICATE OF EXCELLENCE BY SCALER

CERTIFICATE OF EXCELLENCE

THIS CERTIFICATE IS AWARDED TO

SCALER Topics

SRINIVAS G (RA2211003011687)

In recognition of the completion of the tutorial: **DBMS Course - Master the Fundamentals and Advanced Concepts**

Following are the the learning items, which are covered in this tutorial

74 Video Tutorials 16 Modules 16 Challenges 19 April 2024

Anshuman Singh

Co-founder SCALER

9) Conclusion

In conclusion, the development of the online food ordering system database management system (DBMS) project has demonstrated its efficacy in streamlining the food ordering process, enhancing user experience, and optimizing restaurant operations. Through the implementation of a well-structured database schema and efficient query handling, the system successfully facilitates seamless interactions between customers and restaurants, ensuring timely order processing and delivery. Additionally, the incorporation of features such as user authentication, order tracking, and feedback mechanisms contributes to customer satisfaction and loyalty. Overall, the project underscores the significance of robust DBMS solutions in modernizing and improving the efficiency of food service industries.