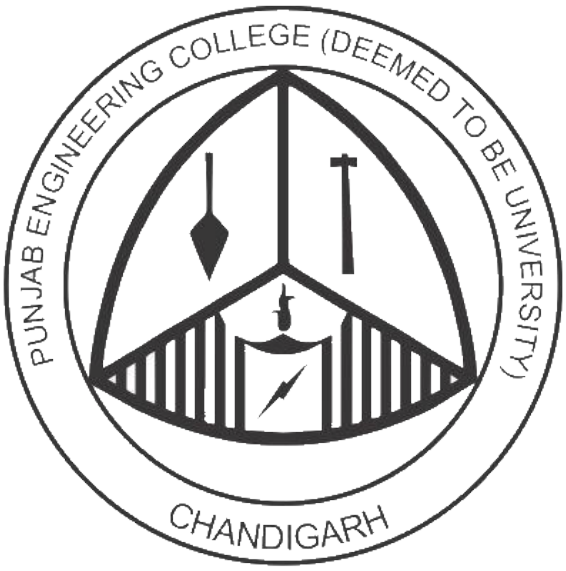


DBMS Assignment

Name: Parth Tandon
SID: 23106065

Name: Ishaan Pratap
SID: 23106056

Under the guidance of
Dr. Alka Jindal

Date of Start: February 10, 2025

Overview

This project presents the design and implementation of a relational database management system tailored for an online food delivery platform inspired by services like Zomato and Swiggy. The backend has been constructed using **MySQL**, emphasizing modularity, normalization, and scalability. The system is capable of handling various real-world operations such as user onboarding, restaurant listings, digital menus, order processing, delivery management, and customer reviews.

The core features include:

- Secure user authentication and account management
- Restaurant onboarding and profile setup
- Comprehensive menu item listings with price and availability control
- Real-time order placement and order history tracking
- Dynamic assignment of delivery partners to orders
- Rating and review system for restaurants
- Seamless database connectivity for integration with web or mobile applications

Designed for production-readiness, this database supports high availability and can be extended with APIs, caching layers, and third-party services.

Entity-Relationship Model

Entities:

- Users
- Restaurants
- Menu Items
- Orders
- Order Items
- Reviews
- Delivery Partners
- Deliveries

Relationships:

- One-to-many between Users and Orders
- One-to-many between Restaurants and Menu Items
- Many-to-many between Orders and Menu Items (via Order Items)
- One-to-many between Restaurants and Reviews
- One-to-many between Delivery Partners and Deliveries

MySQL Table Definitions

Users

```
CREATE TABLE users (
    user_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(255) UNIQUE NOT NULL,
    phone VARCHAR(15) UNIQUE NOT NULL,
    password_hash TEXT NOT NULL,
    address TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Restaurants

```
CREATE TABLE restaurants (
    restaurant_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    address TEXT NOT NULL,
    phone VARCHAR(15) UNIQUE NOT NULL,
    cuisine_type VARCHAR(100),
    rating FLOAT DEFAULT 0,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Menu Items

```
CREATE TABLE menu_items (
    item_id INT AUTO_INCREMENT PRIMARY KEY,
    restaurant_id INT,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    price DECIMAL(10,2) NOT NULL,
    is_available BOOLEAN DEFAULT TRUE,
    FOREIGN KEY (restaurant_id) REFERENCES restaurants(
       restaurant_id) ON DELETE CASCADE
);
```

Orders

```
CREATE TABLE orders (
    order_id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    restaurant_id INT,
```

```
    status ENUM('Pending', 'Preparing', 'Out for Delivery', '
       Delivered', 'Cancelled'),
    total_amount DECIMAL(10,2) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE
        CASCADE,
    FOREIGN KEY (restaurant_id) REFERENCES restaurants(
       restaurant_id) ON DELETE CASCADE
);
```

Order Items

```
CREATE TABLE order_items (
    order_item_id INT AUTO_INCREMENT PRIMARY KEY,
    order_id INT,
    item_id INT,
    quantity INT NOT NULL CHECK (quantity > 0),
    price DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (order_id) REFERENCES orders(order_id) ON
       DELETE CASCADE,
    FOREIGN KEY (item_id) REFERENCES menu_items(item_id) ON
       DELETE CASCADE
);
```

Reviews

```
CREATE TABLE reviews (
    review_id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    restaurant_id INT,
    rating INT CHECK (rating BETWEEN 1 AND 5),
    comment TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE
        CASCADE,
    FOREIGN KEY (restaurant_id) REFERENCES restaurants(
       restaurant_id) ON DELETE CASCADE
);
```

Delivery Partners

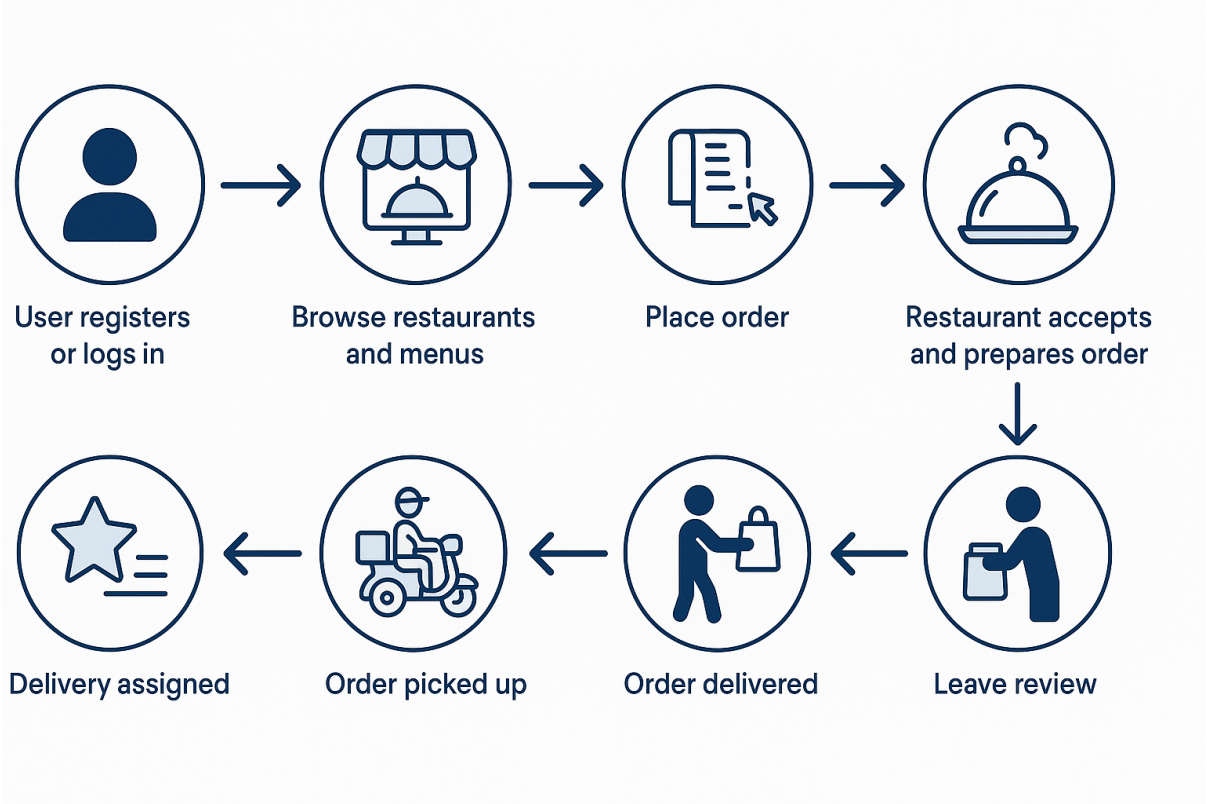
```
CREATE TABLE delivery_partners (
    partner_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    phone VARCHAR(15) UNIQUE NOT NULL,
```

```
    vehicle_details TEXT,
    availability BOOLEAN DEFAULT TRUE
);
```

Deliveries

```
CREATE TABLE deliveries (
    delivery_id INT AUTO_INCREMENT PRIMARY KEY,
    order_id INT,
    partner_id INT,
    status ENUM('Assigned', 'Picked Up', 'Delivered', '
       Cancelled'),
    delivered_at TIMESTAMP,
    FOREIGN KEY (order_id) REFERENCES orders(order_id) ON
       DELETE CASCADE,
    FOREIGN KEY (partner_id) REFERENCES delivery_partners(
       partner_id) ON DELETE SET NULL
);
```

Workflow Diagram



Future Scope and Next Steps

As we continue to evolve this project, there are several enhancements and future directions that can be pursued:

- **Front-End Integration:** Build a responsive web or mobile interface using technologies like React, Flutter, or Angular for interacting with the database in real time.
- **API Development:** Use Node.js, Express, or Django to build RESTful APIs that connect the frontend to the backend.
- **Security Enhancements:** Incorporate OAuth 2.0 authentication, input validation, SQL injection protection, and data encryption techniques.
- **Data Analytics:** Integrate dashboards for order trends, delivery performance, and customer behavior analysis.
- **AI Recommendations:** Implement machine learning models to suggest dishes or restaurants based on user preferences and past orders.
- **High-Performance Scalability:** Introduce indexing, query optimization, and database replication for handling a large number of concurrent users.
- **Continuous Testing & CI/CD Pipelines:** Automate testing and deployment using tools like GitHub Actions, Jenkins, or Docker.

These next steps will help transform this project from a foundational database model into a fully functional, intelligent, and deployable food delivery ecosystem.