### Comprehensive Report for the Food Hub Database

### ****1. Project Overview****

The **Food Hub** database is designed to support a robust and scalable food delivery web application. It encompasses all the functionalities required to manage users, food items, orders, delivery processes, payments, shopping carts, reviews, and addresses. This database structure is modular, allowing seamless data flow across the application and providing a solid foundation for current and future functionalities.

### ****2. Database Tables****

The database consists of 11 tables. Each table is detailed below with its purpose, attributes, relationships, and functionality:

#### ****2.1. Users Table****

**Purpose:** Stores data for customers and administrators.  
**Attributes:**

* id: Unique identifier (Primary Key).
* username: Login username.
* password: Encrypted password.
* email: User email for communication.
* role: Role (customer or admin).
* created\_at: Registration timestamp.

**Functionality:**

* User registration and login.
* Assign roles for access control.
* Relates to orders, reviews, shopping\_cart, and user\_addresses.

#### ****2.2. Food Categories Table****

**Purpose:** Categorizes food items into logical groups.  
**Attributes:**

* id: Unique identifier (Primary Key).
* name: Category name (e.g., Appetizers).

**Functionality:**

* Organizes food items for easy browsing.
* Relates to food\_items.

#### ****2.3. Food Items Table****

**Purpose:** Stores details of all food items available for purchase.  
**Attributes:**

* id: Unique identifier (Primary Key).
* name: Name of the food item.
* description: Detailed description.
* price: Price of the food item.
* image\_url: URL to the item's image.
* category\_id: Foreign key linking to food\_categories.

**Functionality:**

* Displays menu items to users.
* Allows association with order\_items and reviews.

#### ****2.4. Orders Table****

**Purpose:** Tracks all orders placed by users.  
**Attributes:**

* id: Unique identifier (Primary Key).
* user\_id: Foreign key linking to users.
* total\_amount: Total cost of the order.
* status: Current status (e.g., pending, completed).
* created\_at: Order timestamp.

**Functionality:**

* Central entity for tracking purchases.
* Connects to order\_items, payments, and deliveries.

#### ****2.5. Order Items Table****

**Purpose:** Records details of items in each order.  
**Attributes:**

* id: Unique identifier (Primary Key).
* order\_id: Foreign key linking to orders.
* food\_item\_id: Foreign key linking to food\_items.
* quantity: Number of units ordered.
* price: Price per unit.

**Functionality:**

* Tracks itemized details for each order.

#### ****2.6. Deliveries Table****

**Purpose:** Manages delivery operations for each order.  
**Attributes:**

* id: Unique identifier (Primary Key).
* order\_id: Foreign key linking to orders.
* delivery\_address: Address for delivery.
* delivery\_time: Scheduled or completed time.
* delivery\_status: Status (e.g., pending, delivered).
* worker\_id: Foreign key linking to workers.

**Functionality:**

* Monitors delivery progress.
* Assigns workers for delivery tasks.

#### ****2.7. Workers Table****

**Purpose:** Tracks details of delivery personnel.  
**Attributes:**

* id: Unique identifier (Primary Key).
* name: Worker’s name.
* vehicle\_type: Vehicle used (e.g., bike, car).
* status: Current status (e.g., available, on delivery).
* last\_known\_latitude: Worker’s GPS latitude.
* last\_known\_longitude: Worker’s GPS longitude.

**Functionality:**

* Assigns deliveries to workers.
* Monitors worker availability.

#### ****2.8. Payments Table****

**Purpose:** Tracks payment transactions for orders.  
**Attributes:**

* id: Unique identifier (Primary Key).
* order\_id: Foreign key linking to orders.
* payment\_status: Status (e.g., pending, paid).
* payment\_method: Payment mode (e.g., card, cash).

**Functionality:**

* Processes and verifies payments.
* Links directly to orders.

#### ****2.9. Shopping Cart Table****

**Purpose:** Stores items added to a user’s cart.  
**Attributes:**

* id: Unique identifier (Primary Key).
* user\_id: Foreign key linking to users.
* food\_item\_id: Foreign key linking to food\_items.
* quantity: Number of units added.

**Functionality:**

* Temporary storage for unconfirmed purchases.
* Allows seamless transition to orders.

#### ****2.10. Reviews Table****

**Purpose:** Captures feedback on food items.  
**Attributes:**

* id: Unique identifier (Primary Key).
* food\_item\_id: Foreign key linking to food\_items.
* user\_id: Foreign key linking to users.
* rating: Numeric rating (1-5).
* comment: User's written feedback.
* created\_at: Review timestamp.

**Functionality:**

* Enables user feedback collection.
* Displays reviews for food items.

#### ****2.11. User Addresses Table****

**Purpose:** Manages delivery addresses for users.  
**Attributes:**

* id: Unique identifier (Primary Key).
* user\_id: Foreign key linking to users.
* address: Full delivery address.
* city: User’s city.
* postal\_code: Address postal code.
* country: Address country.
* is\_default: Marks default delivery address.

**Functionality:**

* Stores multiple addresses per user.
* Enables easy selection during checkout.

### ****3. Database Relationships****

The following relationships are implemented:

1. **Users ↔ Orders:** A user can place multiple orders.
2. **Users ↔ Shopping Cart:** A user can manage items in their shopping cart.
3. **Users ↔ Reviews:** A user can leave multiple reviews.
4. **Users ↔ Addresses:** A user can save multiple addresses.
5. **Food Categories ↔ Food Items:** Each food item belongs to one category.
6. **Food Items ↔ Order Items:** Food items are added to orders via order items.
7. **Orders ↔ Order Items:** An order can have multiple order items.
8. **Orders ↔ Payments:** An order can have one payment.
9. **Orders ↔ Deliveries:** An order can have one delivery.
10. **Workers ↔ Deliveries:** A worker can deliver multiple orders.

### ****4. Database Integrity and Constraints****

* **Primary Keys:** Ensure unique identification of each record.
* **Foreign Keys:** Maintain referential integrity between tables.
* **Enum Values:** Restrict values for fields like status and role.
* **Check Constraints:** Ensure valid values (e.g., ratings between 1 and 5).

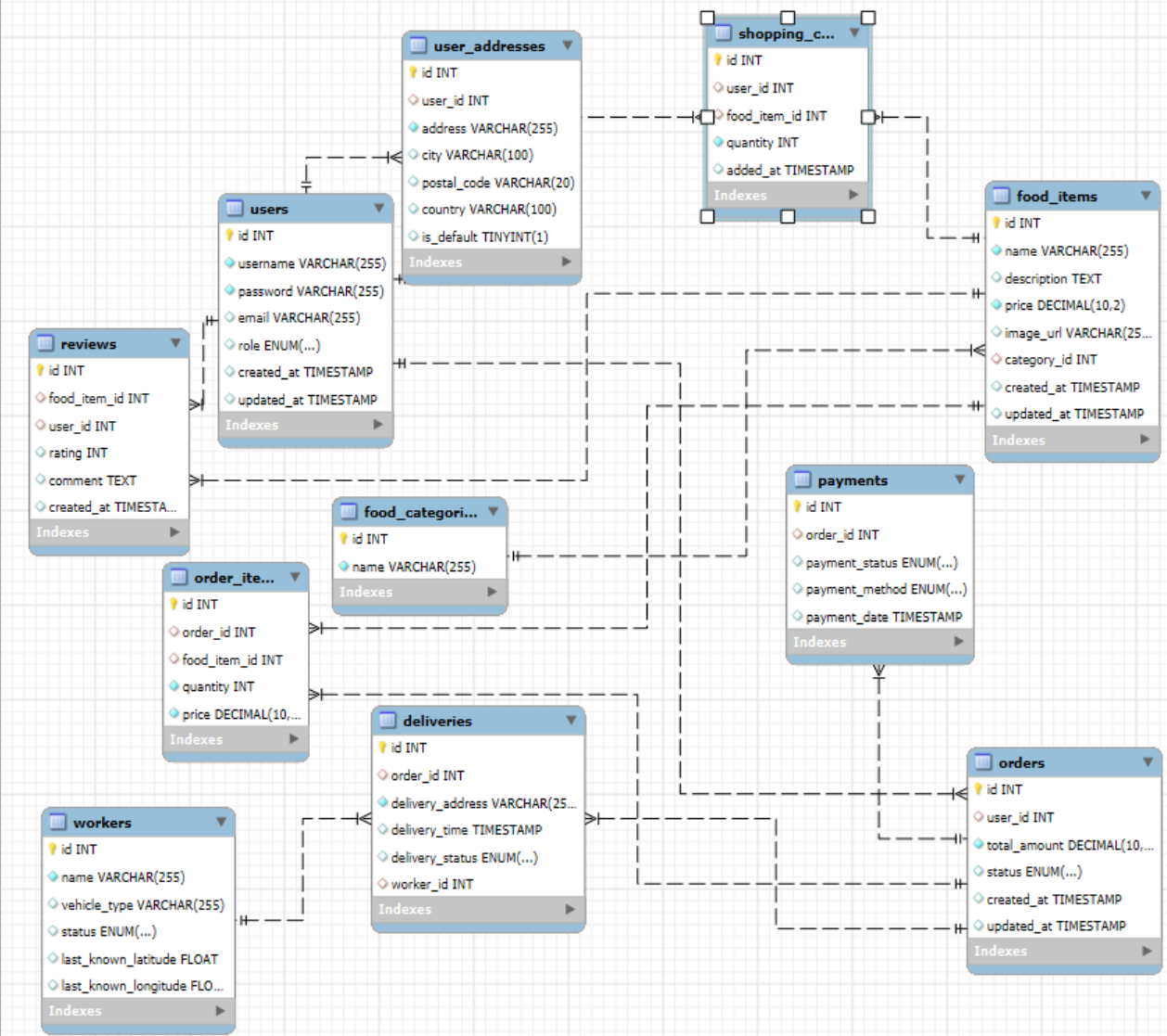
5. Functionality Overview

|  |  |  |
| --- | --- | --- |
| Feature | Tables Involved | Description |
| User Registration & Login | Users | Users register and log in with roles defined for access control. |
| Browsing Food Menu | Food Categories, Food Items | Food items are categorized and displayed to users. |
| Placing Orders | Orders, Order Items | Users place orders, linking food items with quantities. |
| Payment Processing | Payments, Orders | Payments are processed and linked to orders. |
| Delivery Management | Deliveries, Workers | Tracks delivery status and assigns workers for each order. |
| Shopping Cart | |  | | --- | |  |  |  | | --- | | Shopping Cart, Food Items, Users | | Users can add items to their cart for checkout later. |
| Reviewing Food Items | Reviews, Food Items, Users | Users can review and rate food items after purchase. |
| Managing Addresses | User Addresses, Users | Users can save and manage multiple delivery addresses. |

### ****6. Scalability and Future Enhancements****

* **Analytics:** Integrate analytics to track popular food items, peak order times, and worker efficiency.
* **Loyalty Program:** Add a rewards system to track user purchases and award points.
* **Real-Time Tracking:** Enhance the deliveries table to include live GPS tracking for workers.

# **Entity Relationship Diagram**



### ****7. Conclusion****

The **Food Hub** database is a robust, scalable, and modular system, ensuring seamless operations for a food delivery platform. It efficiently supports the end-to-end workflow, from user registration and order management to delivery and feedback collection.