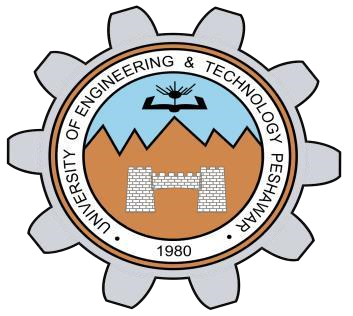
**Key Milestone 4**



**DBMS Final Project**

**Submitted By:** Muhammad Saad Amjad Khan

**Project Group #19:**

Muhammad Ehzaz Khan (22pwcse2108)

Muhammad Saad Amjad Khan (22pwcse2133)

Muhammad Kamil Khan (22pwcse2174)

**Section:** B

**Submitted to:** Engr.Sumayyea Salahuddin

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work”

Department of Computer Systems Engineering

University of Engineering and Technology Peshawar

**Project: Restaurant Management System**

# Implementation Details of Restaurant Management System

## **Introduction**

The Restaurant Management System (RMS) is a web-based application developed using Laravel (PHP framework) to streamline and digitize the operations of a restaurant. It handles orders, invoices, menus, customer details, and table bookings efficiently while ensuring a user-friendly interface for both admins and customers.

## **Technologies Used**

* Backend: PHP (Laravel 12)
* Frontend: HTML, CSS, Blade Templates
* Database: MySQL
* Tools: VS Code, XAMPP, Composer
* Libraries: Bootstrap (for UI), Laravel Eloquent ORM

## **Modules Implemented**

## **User Authentication**

* Users can register and log in using Laravel’s built-in authentication.
* Admin has privileges to view and manage all modules.

## **Food/Menu Management**

* Admin can add, update, and delete food items.
* Each food item includes:
  + Name
  + Description
  + Price
  + Category
  + Image (uploaded and stored in public folder)

**Order Management:**

* Users can place orders from the available food menu.
* Orders are stored in the database and linked to both user and food entities.
  + Admin can view all orders placed with details such as:
  + Food ordered
  + Quantity

## Customer information

## Total cost

## Order status

## **Invoice Generation**

* Admin can generate invoices for completed orders.
* Each invoice includes:
  + Invoice ID
  + Linked order details
  + Date & time
  + Total bill amount
  + Payment status

## **Table Reservation**

* Users can reserve tables by providing:
  + Name
  + Phone number
  + Date and time
  + Number of people
  + Admin can manage all reservations and mark them as confirmed or cancelled.

## **Review & Feedback**

* Users can submit reviews and feedback after ordering.
* Admin can view and moderate these reviews.

## **Database Design**

## **Main Tables**

* users: Stores user information (id, name, email, password, phone, address).
* foods: Stores menu items (id, name, price, description, image).
* orders: Links users to food items with quantity and status.
* invoices: Generated for completed orders.
* reservations: Stores table booking information.
* reviews: Stores customer reviews for food or service.

## **Admin Dashboard**

* Displays key metrics:
* Total Orders
* Completed Orders
* Revenue Generated
* Reservations Count
* Provides quick links to manage each module.
* Uses Laravel Blade templates with Bootstrap components.

## **User Interface**

* Built using HTML/CSS with Laravel Blade.
* Clean navigation bar.
* Food menu displayed as responsive cards.
* Reservation and order forms with validation.
* Admin pages with tables and actionable buttons.

## **Error Handling and Validation**

* Server-side validation using Laravel’s validate() method.
* Try-catch blocks implemented for database operations.
* Custom error pages for 404 and 500.

## **Security Measures**

* Passwords hashed using Laravel's built-in bcrypt().
* Middleware used to protect admin routes.
* CSRF protection enabled by default.
* Mass assignment protection using $fillable.

## **Testing and Deployment**

* Manual testing performed on:
* Order placement
* Invoice generation
* Reservation booking
* Deployed locally using XAMPP.
* Database migrations handled via Laravel Artisan.

**Business Rules for Restaurant Management System:**

1. users

**Business Role:**  
Stores the details of customers who use the restaurant system. Each user can browse the menu, add items to their cart, place orders, leave reviews, and book tables.

**Key Functional Rules:**

* One user can have multiple orders.
* One user can post multiple reviews.
* One user can have multiple invoices and carts.
* Each user is uniquely identified by their ID.

1. employees

**Business Role:**  
Manages records of restaurant staff such as waiters, chefs, and managers. Employees are assigned to handle orders and work shifts.

**Key Functional Rules:**

* Each employee has a designated shift.
* Employees may be linked to handling one or more orders.

1. food

**Business Role:**  
Maintains the master list of food items offered by the restaurant. This includes information like name, description, price, and image for display.

**Key Functional Rules:**

* Food items appear in the menu.
* Food items are referenced in orders, cart, and reviews.

1. orders

**Business Role:**  
Stores all placed orders made by users. Each order links a user to a food item and the employee responsible for fulfilling it. Contains the order’s current status and amount.

**Key Functional Rules:**

* Each order belongs to one user.
* Each order is assigned to one employee.
* Each order is for one food item (in flattened design).
* Order statuses include: Pending, Completed, Cancelled.

1. tables

**Business Role:**  
Stores information about the physical dining tables in the restaurant including their seating capacity and current availability status.

**Key Functional Rules:**

* Each table can be reserved (linked via books).
* Table status can be: Available, Reserved, Occupied.

1. books

**Business Role:**  
Manages customer reservations for dining tables. Tracks which user reserved which table at what date and time, for how many guests.

**Key Functional Rules:**

* A user can make multiple bookings.
* A table can have multiple bookings (on different dates/times).
* Booking includes time and number of guests.

1. reviews

**Business Role:**  
Captures feedback from users about food items. Each review includes a rating and review text, tied to a specific food item.

**Key Functional Rules:**

* A user can review multiple food items.
* A food item can have many reviews.
* Ratings help in calculating average food quality scores.

1. carts

**Business Role:**  
Temporarily stores food items that a user is planning to order. Acts like a shopping basket before converting to an order.

**Key Functional Rules:**

* A cart is user-specific.
* Items in the cart are not yet part of any official order.
* Cart is cleared when the order is placed.

1. invoice

**Business Role:**  
Keeps a financial record of each completed order. Used for billing, accounting, and payment tracking.

**Key Functional Rules:**

* Each invoice belongs to a user and an order.
* Invoices store final amount, status (e.g., Paid, Unpaid), and timestamp.
* One invoice per order.

**Main Entities & Their Description:**

1. **users:**

|  |  |  |
| --- | --- | --- |
| **Attributes** | **Datatypes** | **Name in Database** |
| **id** (PK) | Integer | id |
| name | String | name |
| phone | String | phone |
| email | String | email |
| address | String | address |
| password | string | password |

1. **employees:**

|  |  |  |
| --- | --- | --- |
| **Attributes** | **Datatypes** | **Name in Database** |
| **id** (PK) | Integer | id |
| name | String | name |
| role | String | role |
| phone | String | phone |
| salary | Integer | salary |
| shift\_timing | DateTime | shift |
| date\_of\_joining | Date | date\_joining |

1. **foods:**

|  |  |  |
| --- | --- | --- |
| **Attributes** | **Datatypes** | **Name in Database** |
| **id** (PK) | Integer | id |
| title | String | title |
| detail | String | detail |
| price | Integer | price |
| image | String | image |

1. **orders:**

|  |  |  |
| --- | --- | --- |
| **Attributes** | **Datatypes** | **Name in Database** |
| **id** (PK) | Integer | id |
| **user\_id** (FK) | Integer | user\_id |
| **food\_id** (FK) | Integer | food\_id |
| **Employee\_id** (FK) | integer | Employee\_id |
| amount | Integer | amount |
| status | String | status |

1. **tables:**

|  |  |  |
| --- | --- | --- |
| **Attributes** | **Datatypes** | **Name in Database** |
| **id**(PK) | Integer | id |
| capacity | Integer | capacity |
| details | String | detail |
| status | String | status |

1. **books:**

|  |  |  |
| --- | --- | --- |
| **Attributes** | **Datatypes** | **Name in Database** |
| **id** (PK) | Integer | id |
| **table\_id** (FK) | Integer | table\_id |
| phone | String | phone |
| guests | Integer | guests |
| date | DateTime | date |
| time | DateTime | time |

1. **reviews:**

|  |  |  |
| --- | --- | --- |
| **Attributes** | **Datatypes** | **Name in Database** |
| **id** (PK) | Integer | id |
| **food\_id** (FK) | Integer | food\_id |
| review | String | review |
| rating | Integer | rating |
| date | DateTime | date |

1. **carts:**

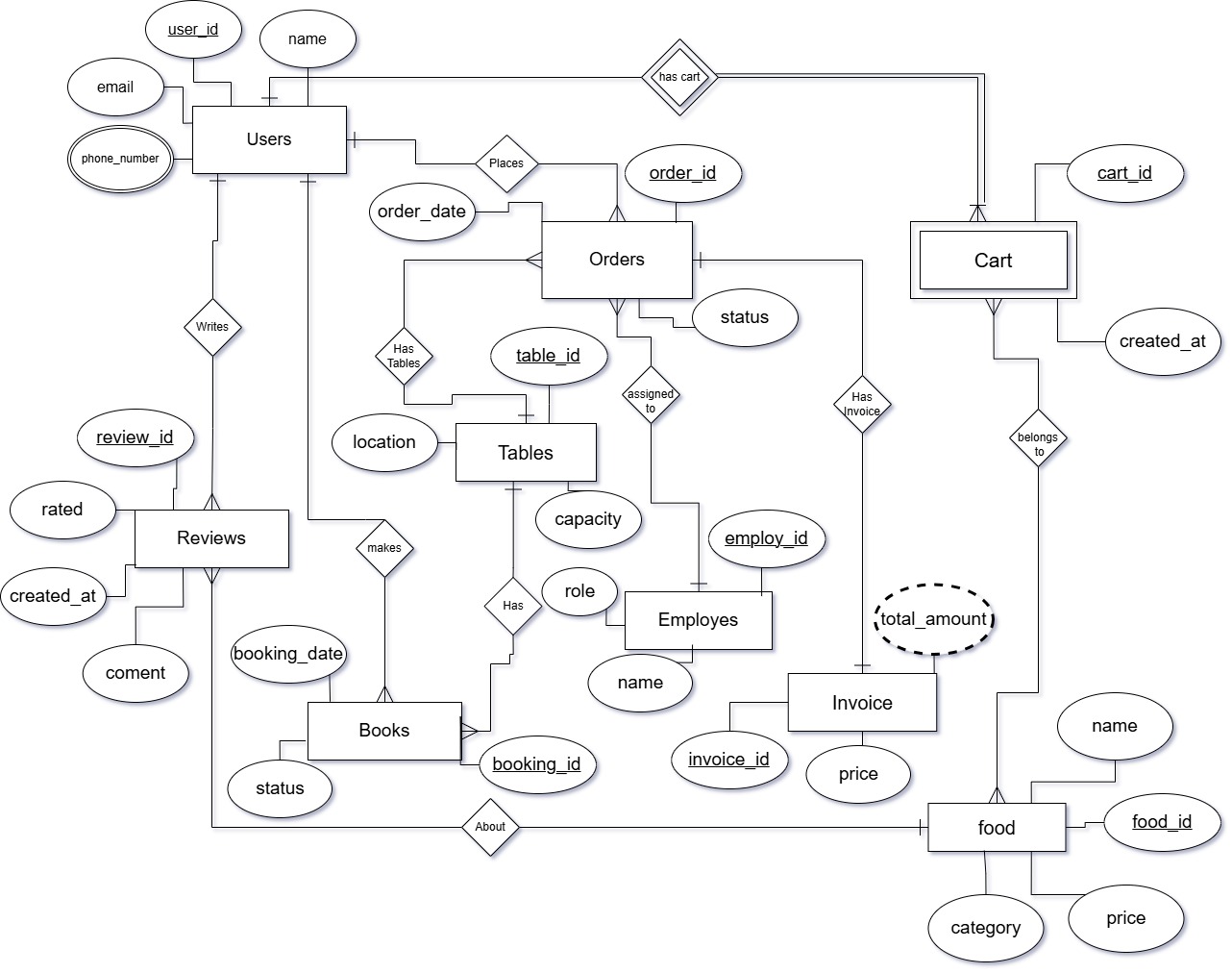
|  |  |  |
| --- | --- | --- |
| **Attributes** | **Datatypes** | **Name in Database** |
| **id** (PK) | Integer | id |
| **user\_id** (FK) | Integer | user\_id |
| **food\_id** (FK) | Integer | food\_id |
| quantity | Integer | quantity |

1. **invoice:**

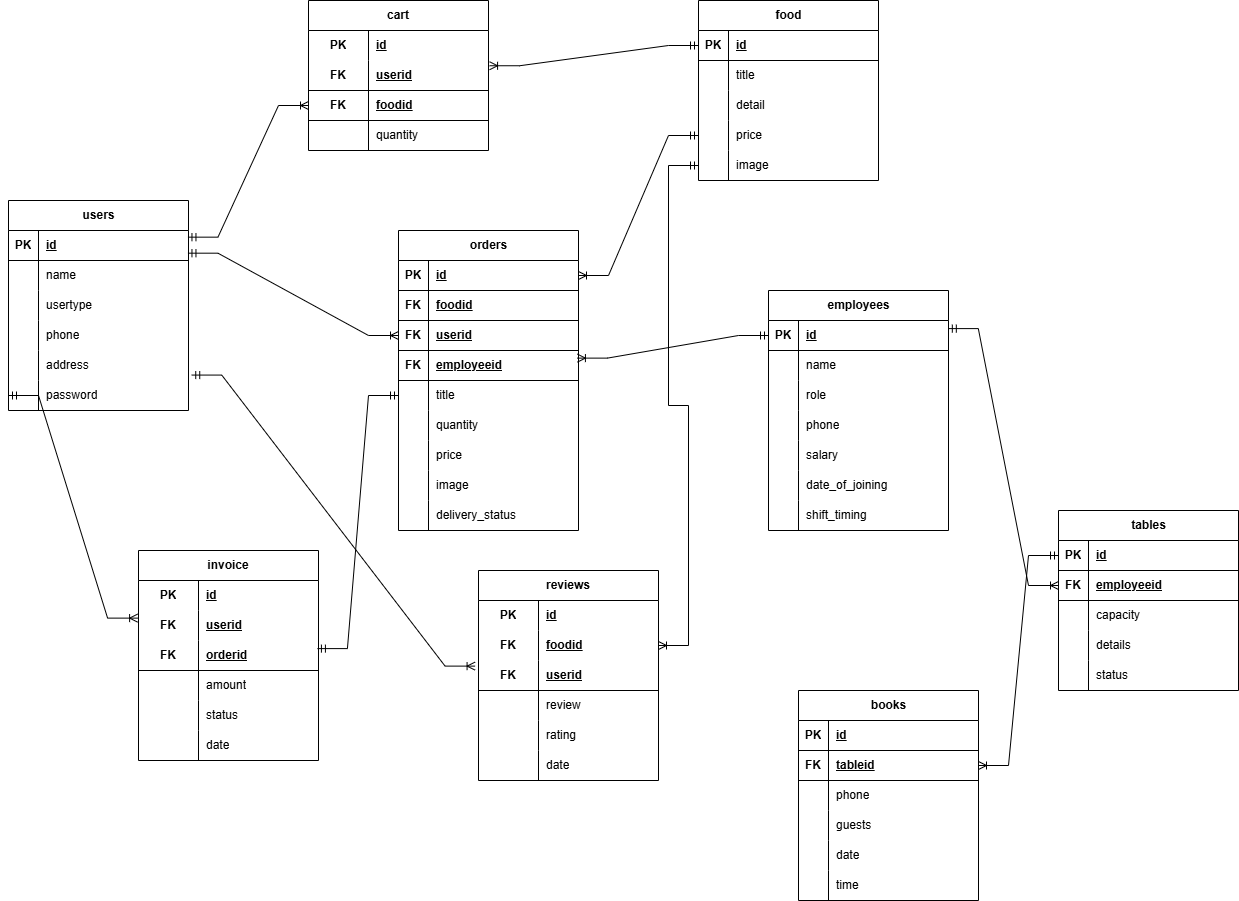
|  |  |  |
| --- | --- | --- |
| **Attributes** | **Datatypes** | **Name in Database** |
| **id** (PK) | Integer | id |
| **order\_id** (FK) | Integer | order\_id |
| **user\_id** (FK) | Integer | user\_id |
| amount | Integer | amount |
| status | String | status |
| date | DateTime | date |

**Finalized Conceptual Schema:**

**ERD (Entity Relationship Diagram):**

****

**EERD (Enhanced Entity Relationship Diagram):**



**Final Normalized Forms:**

users(id PK, name, phone, email, address, password)

employees(id PK, name, role, phone, salary, date\_of\_joining, shift\_timing)

books(id PK, table\_id FK, phone, guests, date)

carts(id PK, user\_id FK, food\_id FK, quantity)

tables(id PK, details, capacity, status)

orders(id PK, food\_id FK, user\_id FK, employee\_id FK, total\_amount, order\_status)

reviews(id PK, food\_id FK, user\_id FK, review, rating, date)

invoice(id PK, order\_id FK, user\_id FK, amount, status, date)

## **Conclusion**

The Restaurant Management System simplifies restaurant operations and enhances the customer experience. With a modular architecture and modern technologies, it can be extended further by integrating features like online payments, notification emails, or multi-branch support.

**References:**

* ChatGPT
* Youtube
* Draw.io
* Cursor AI
* Claude