**DBMS Project**

**“Key Milestone 1”**

A logo of a university of engineering and technology

Description automatically generated

**Spring 2025**

**CSE 404L – Database Management Systems Lab**

**SUBMITTED BY:** Hoorish Ahmad

Afreen Nasrullah

Sidrah Ishtiaq

**REGISTRAION NUMBER:** 22PWCSE2198

22PWCSE2195

22PWCSE2159

**BATCH:** 24

**CLASS SECTION:** B

**SUBMITTED TO:** Engr. Sumayyea Salahuddin

*Department of Computer Systems EngineeringUniversity of Engineering and Technology, Peshawar*

**1. Entity Description:**

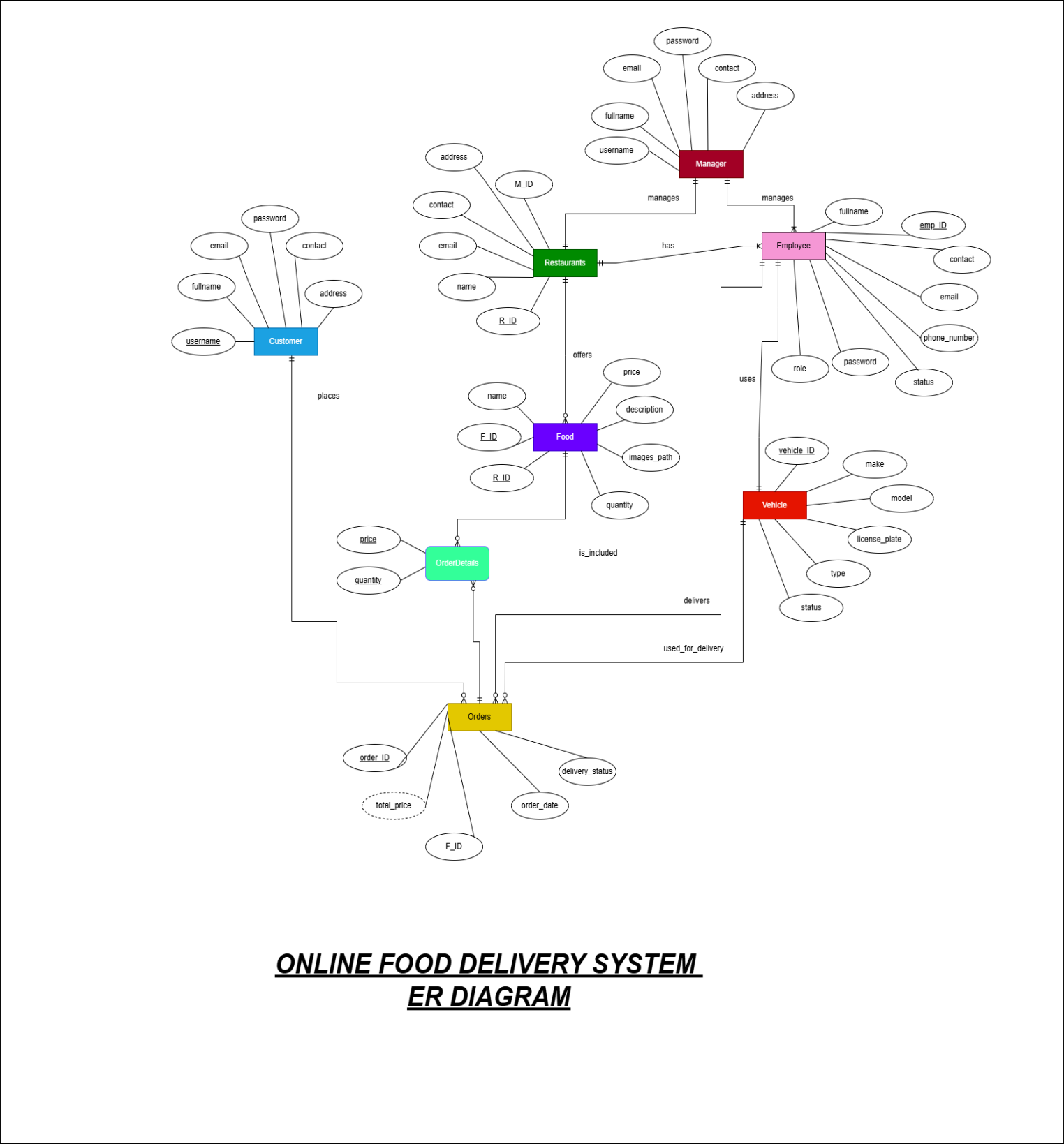
|  |  |
| --- | --- |
| Entity | Description |
| User | Customers using the platform to browse, order food, and manage their profile. |
| Food | Menu items listed by the system. Includes name, price, type, picture, etc. |
| Order | Represents a food order placed by a user (delivery or pickup). |
| Food\_Order | Pivot table that stores quantity and price for each food item in an order. |
| Order\_Items | Alternative pivot that may be used to represent additional item-specific order logic. |
| Billing *(optional)* | Can be inferred from the orders.total, representing the total amount paid. |

**2. Detailed Business Rules:**

1. **Users** must register with necessary info like name, email, password, and address.
   * A user can place multiple orders.
   * Each order belongs to exactly one user.
2. **Foods** are predefined in the system and include fields like food ID, name, price, description, type (e.g., pizza, burger), and image URL.
3. **Orders** contain:
   * type: Either “delivery” or “pickup”
   * deliveryAddress if type is "delivery"
   * total amount of all food items
   * status: Pending, Preparing, Completed, or Cancelled
   * receipt\_image: proof of payment uploaded by the user
4. Each **order** can contain multiple food items, and each food item can be part of many orders — this is managed through the food\_order pivot table which stores:
   * quantity of each food item in that order
   * price of the item when ordered
5. Order\_Items may optionally represent itemized billing or be used for expansion (like offers, modifiers, etc.)
6. **Billing** is implicit — each order holds its total amount, and status reflects fulfillment and payment stage.
7. A receipt image (stored in receipt\_image) can be uploaded by the user for manual verification by the admin.

**Relationships Summary:**

* One **user** ➝ many **orders**
* One **order** ➝ many **food items** (via food\_order)
* One **food** ➝ part of many **orders**
* One **order** ➝ one **receipt image** (optional)
* One **order** ➝ optional use of order\_items as future flexibility

**ER Diagram:**

**Justification for not using EERD:**

In this project, a standard Entity Relationship Diagram (ERD) was used instead of an Enhanced ERD (EERD) due to the straightforward nature of the food delivery system. EERDs are typically applied when there is a need to model complex relationships such as inheritance, specialization, generalization, or multivalued attributes. However, the entities in this system—such as users, food items, orders, and order details—are simple and well-structured, without any hierarchical or subclass relationships. Each entity contains clearly defined attributes, and the relationships among them are direct and normalized, making an ERD sufficient for capturing the system’s requirements. Using an EERD would have introduced unnecessary complexity without adding significant value to the design.

**References:**

* <https://gemini.google.com> used for understanding the relations.
* <https://chatgpt.com> used for defining entities and attributes.
* <https://www.mermaidchart.com> used for understanding cardinalities in er diagram.