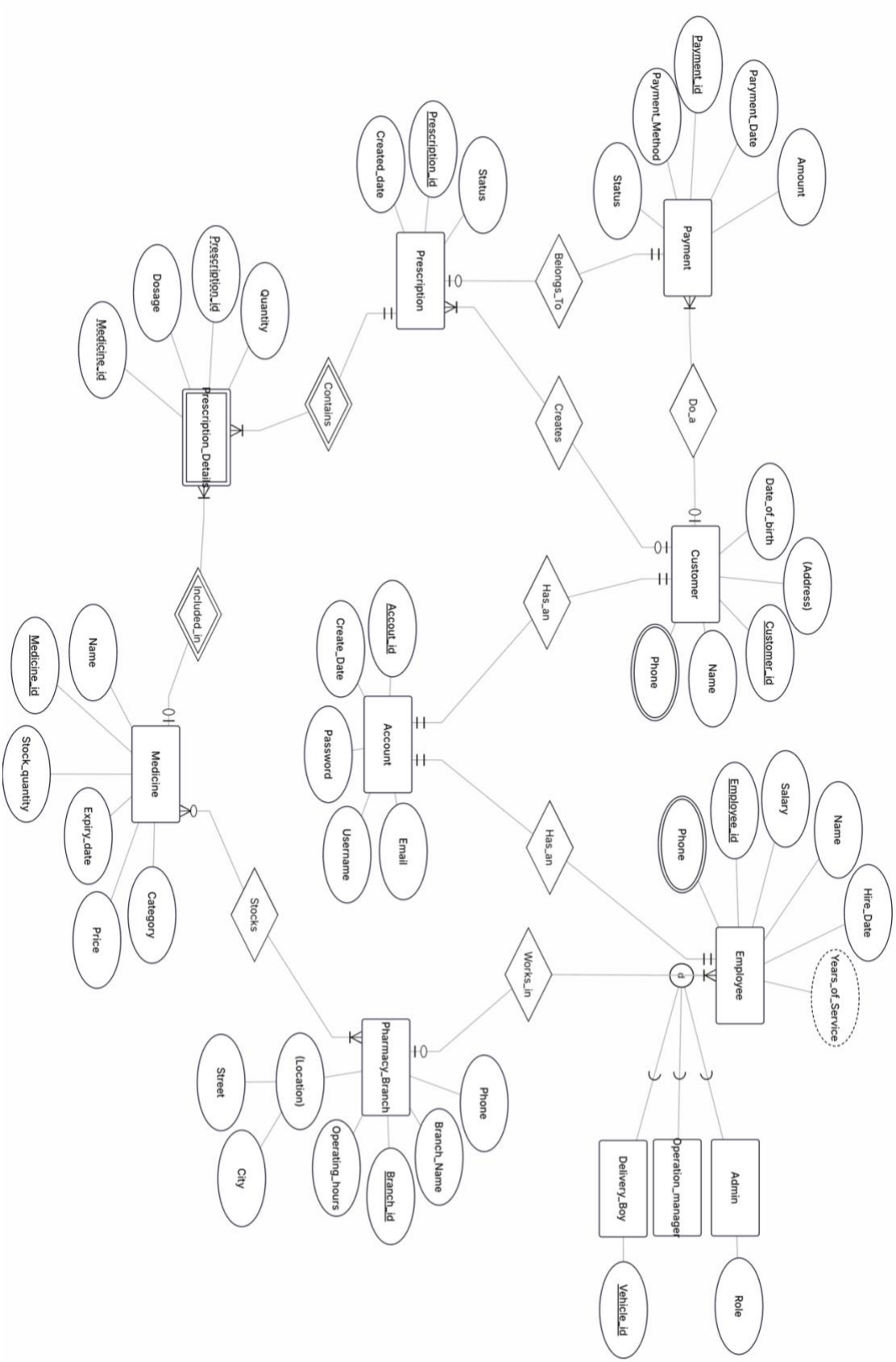
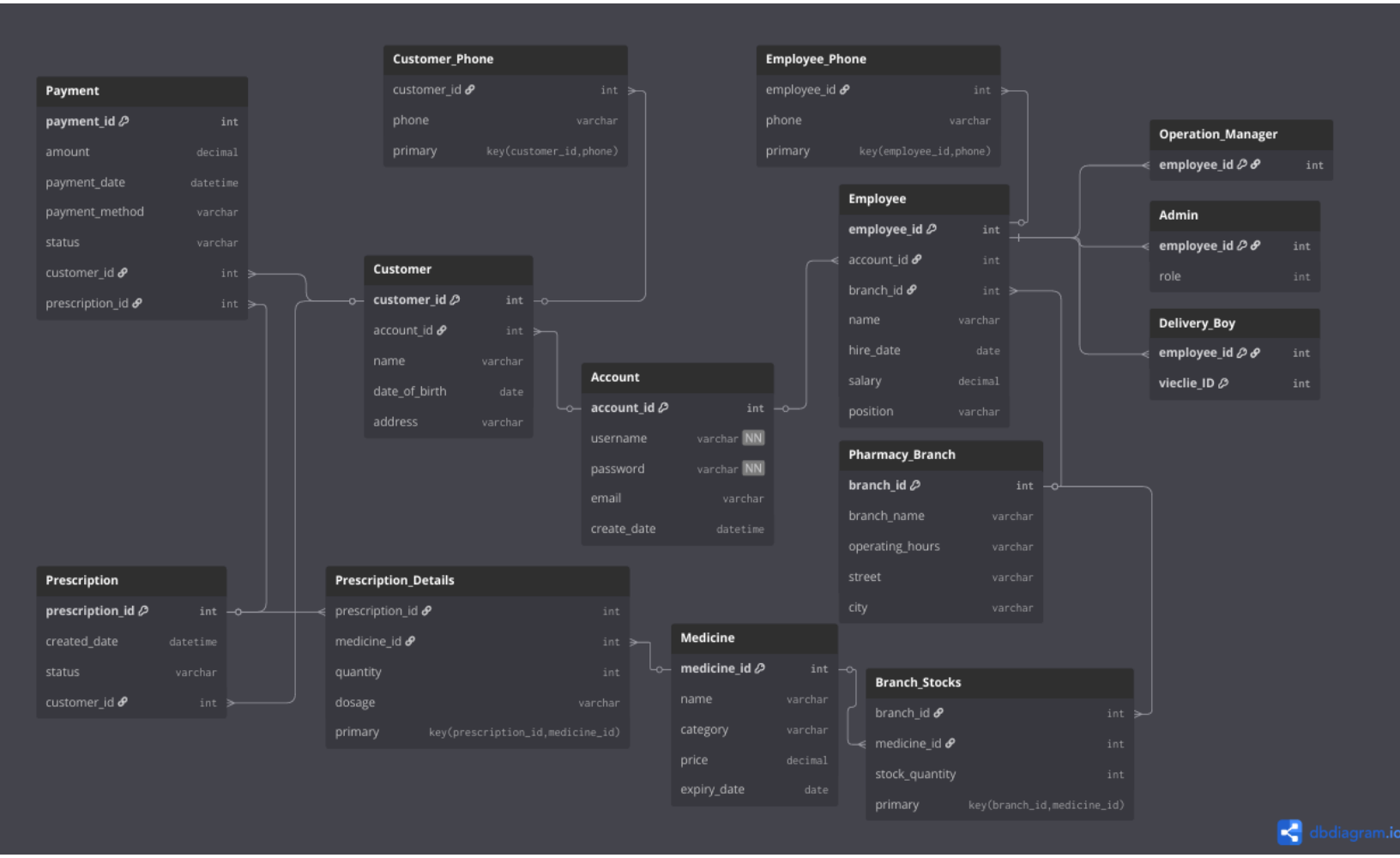
Entity relationship diagram



Relational schema



Assumptions

🦠 1. General Overview

The *Medicine Ordering System* is designed to manage the process of ordering medicines online from pharmacy branches.
It handles user registration, prescription management, payment, delivery, and stock control across multiple branches.

⚙️ 2. General Assumptions

1. Each **customer** must have an **account** to place orders or create prescriptions.
2. A **customer** can make multiple **payments**, but each payment is linked to only one **prescription**.
3. Each **prescription** can include multiple medicines through the **Prescription_Details** relationship.
4. **Medicines** are managed per branch (each branch keeps its own stock levels).
5. Each **employee** works in one pharmacy branch only.
6. **Employees** can be one of the following types (inheritance):
    - **Admin:** manages system operations and users.
    - **Operation_Manager:** manages pharmacy branches and medicines.
    - **Delivery_Boy:** responsible for delivering medicines and has a vehicle assigned.
7. **Payment** can be completed by multiple methods (cash, card, wallet, etc.).
8. Each **branch** has a specific location, operating hours, and phone number.
9. **Medicine** has a unique ID, name, category, expiry date, price, and stock quantity.
10. The system supports **multiple pharmacy branches**, each managed separately but sharing a central database.

🧍 3. Entity-Specific Assumptions

👩 Customer

- Must provide a valid phone number, address, and date of birth.
- Can create multiple prescriptions.
- Each customer has exactly one account (1:1 relationship).

💊 Medicine

- Each medicine must belong to at least one pharmacy branch through the stock relationship.
- Expiry date must be later than the current system date.
- Stock quantity is updated automatically after every sale.

🏪 Pharmacy_Branch

- Each branch operates independently but shares the same product catalog.
- Location includes both *street* and *city*.
- A branch must have at least one employee assigned.

👔 Employee

- Each employee has an Employee_id, Name, Hire_date, Years_of_service, Salary, and Phone.
- The employee's **role** determines access privileges.
- Only admins can add or remove employees from the system.

📜 Prescription

- Each prescription is created by a **customer** and linked to a **payment** record.
- Each prescription has a Status (Pending / Approved / Delivered).
- Prescription_Details links medicines to prescriptions, including quantity and dosage.

💳 Payment

- Each payment is for a single prescription only.
- Payment includes Amount, Date, Method, and Status.
- Payment status can be Pending, Completed, or Failed.

🚚 Delivery_Boy

- Each delivery boy is an employee with an assigned vehicle ID.
- Responsible for delivering orders to customer addresses.
- Must report delivery completion to update the prescription status.

🔒 4. System Constraints & Non-Functional Assumptions

- Each user's **password** is stored securely (hashed).
- System must support **concurrent users** across multiple branches.
- Data should be **consistent** across all branches (centralized DB).
- System must provide **data validation** (no duplicate medicines or customers).
- The UI should be user-friendly and mobile-compatible.
- Backups of the database are taken automatically every 24 hours.

🧮 5. Business Rules

1. No medicine can be sold after its expiry date.
2. A prescription cannot be marked "Delivered" until payment is confirmed.
3. Only admins can delete users or medicines.
4. Stock updates occur automatically upon confirmed payment.
5. Each branch must have at least one operation mana

Relationship	Entities Involved	Cardinality	Participation	Explanation
Creates	Customer – Prescription	1-to-Many (1:N)	Customer: Partial, Prescription: Total	A customer can create many prescriptions; each prescription is created by one customer.
Belongs_To	Prescription – Payment	1-to-1 (1:1)	Prescription: Partial, Payment: Total	Each payment is linked to one prescription; a prescription may exist before payment.
Do_a	Customer – Payment	1-to-Many (1:N)	Customer: Partial, Payment: Total	A customer can make several payments; each payment must belong to one customer.
Contains	Prescription – Prescription_Details	1-to-Many (1:N)	Total – Total	Each prescription must contain at least one medicine detail, and each detail belongs to exactly one prescription.
Included_in	Prescription_Details – Medicine	Many-to-1 (N:1)	Prescription_Details: Total, Medicine: Partial	Each prescription detail refers to one medicine; a medicine can exist even if not prescribed yet.
Has_an	Customer – Account	1-to-1 (1:1)	Total – Total	Every customer has one account, and each account belongs to one customer.
Has_an	Employee – Account	1-to-1 (1:1)	Total – Total	Each employee has one login account, and each account belongs to one employee.
Works_in	Employee – Pharmacy_Branch	Many-to-1 (N:1)	Employee: Total, Branch: Partial	Every employee must work in one branch; a branch can exist even if no employees are assigned yet.
Stocks	Pharmacy_Branch – Medicine	Many-to-Many (M:N)	Branch: Total, Medicine: Partial	Each branch stocks many medicines; a medicine can exist in multiple branches.
Assigned_to	Delivery_Boy – Vehicle	1-to-1 (1:1)	Delivery_Boy: Partial, Vehicle: Total	A vehicle is always assigned to one delivery boy, but not all delivery boys have vehicles.
ISA	Employee – {Admin, Operation_Manager, Delivery_Boy}	1-to-1 (Disjoint, Total)	Employee: Total, Subtypes: Total	Every employee must belong to one subtype only.