Question: Design a Crowdsourced Delivery Service Platform Database

Description:

You are tasked with designing a database for a crowdsourced delivery service platform. This platform connects individuals needing items delivered (Senders) with people willing to deliver those items (Deliverers). The system should manage users, delivery requests, payments, reviews, notifications, and vehicles.

Requirements:

1. **Users:**
    - The platform will have two types of users: Senders and Deliverers.
    - Each user should have a unique identifier, name, email, phone number, address, and user type (Sender/Deliverer).
    - Users should be able to receive ratings and reviews.
2. **Delivery Requests:**
    - A delivery request includes details such as pickup and dropoff addresses, package details, status (Pending, In Progress, Completed, Canceled), request time, and delivery time.
    - Each delivery request is associated with a sender and a deliverer.
3. **Payments:**
    - Payments are made for completed delivery requests.
    - Payment details include a unique identifier, amount, payment method, and payment date.
    - Each payment is associated with a specific delivery request.
4. **Reviews:**
    - Users can leave reviews and ratings for other users.
    - A review includes a unique identifier, rating, comment, review date, and references to the reviewer and the reviewed user.
    - Each review is associated with a specific delivery request.
5. **Notifications:**
    - Users receive notifications about the status of their delivery requests.
    - Notification details include a unique identifier, user ID, message, timestamp, and read status.
6. **Vehicles:**
    - Deliverers can register their vehicles on the platform.
    - Vehicle details include a unique identifier, vehicle type, license plate, and reference to the deliverer.

Tasks:

1. **Identify Entities and Attributes:**
    - List all the entities and their respective attributes as described above.

2. **Define Relationships:**
    - Identify and define the relationships between the entities. Specify cardinality (one-to-one, one-to-many, many-to-many).
3. **Create an ER Diagram:**
    - Draw an Entity-Relationship Diagram (ERD) that visually represents the entities, attributes, and relationships.

Additional Considerations:

- Ensure that each entity has a primary key.
- Use foreign keys to establish relationships between entities.
- Consider constraints to maintain data integrity (e.g., a **delivery request cannot be completed without a payment**).

Real-World Scenario Questions:

1. **Delivery Request Management:**
    - How would you retrieve all pending delivery requests for a specific user?
    - How can you update the status of a delivery request from "Pending" to "In Progress"?
2. **Payment Processing:**
    - How would you record a payment for a completed delivery request?
    - How can you retrieve all payments made by a specific user over the past month?
3. **User Reviews and Ratings:**
    - How would you calculate the average rating for a specific user?
    - How can you retrieve all reviews written by a user about deliverers?
4. **Notification Handling:**
    - How can you send a notification to a user when their delivery request status changes?
    - How would you retrieve all unread notifications for a specific user?
5. **Vehicle Management:**
    - How can you list all vehicles registered by a specific deliverer?
    - How would you update the details of a deliverer's vehicle (e.g., change the license plate number)?
6. **User Interaction:**
    - How would you match a delivery request with an available deliverer based on proximity and vehicle type?
    - How can you track the real-time location of a delivery in progress?