Requirements Document for Morent: A Rental E-commerce Marketplace

Business Overview

Morent is a platform where users can rent automobiles such as cars, bikes, and vans. The marketplace aims to simplify the vehicle rental process while ensuring a seamless user experience for both renters and vehicle owners.

Key Objectives

1. **User-Friendly Platform**: Enable users to browse, book, and pay for rentals effortlessly.
2. **Secure Transactions**: Ensure trust and safety for both renters and vehicle owners.
3. **Scalable Architecture**: Support a growing user base and potential feature expansions.

Target Audience

- **Primary Users**: Individuals and businesses needing short-term vehicle rentals.
- **Vehicle Owners**: People or companies willing to rent out their vehicles.
- **Location**: Focus on metropolitan areas in Pakistan, with plans for regional expansion.

Core Features

For Renters:

- **Vehicle Search**: Filter vehicles by type, location, price, and availability.
- **Booking System**: Reserve vehicles with real-time availability.
- **Payment Integration**: Secure payment options, including credit cards and local payment gateways.
- **Reviews and Ratings**: View feedback on vehicles and owners.

For Vehicle Owners:

- **Listing Management**: Upload and manage vehicle details, pricing, and availability.
- **Earnings Dashboard**: Track rental income.
- **Owner Verification**: Ensure trust via KYC processes.

Admin Panel:

- **User Management**: Monitor and manage renters and owners.
- **Transaction Monitoring**: Oversee payment flows and resolve disputes.
- **Analytics**: Insights into platform performance and user behavior.

Technical Requirements

1. **Frontend**: Responsive web application using Next.js with Tailwind CSS.
2. **Backend**:
    - Framework: Node.js with Express or FastAPI.
    - Database: PostgreSQL for relational data storage.
3. **APIs**:
    - User Authentication: Register, login, and manage profiles.
    - Vehicle Management: CRUD operations for listings.
    - Booking and Payments: Secure endpoints for transactions.
4. **Third-Party Integrations**:
    - Payment Gateway: Stripe or local solutions like JazzCash.
    - Geolocation Services: Google Maps API for location-based search.
5. **Scalability**:
    - Use AWS or Coolify for hosting.
    - Implement caching with Redis for high-traffic endpoints.

Compliance and Security

- **Data Protection**: Follow GDPR-equivalent standards for user data privacy.
- **Secure Payments**: PCI-DSS compliance for financial transactions.
- **Verification**: Ensure KYC compliance for vehicle owners.

Schema Design

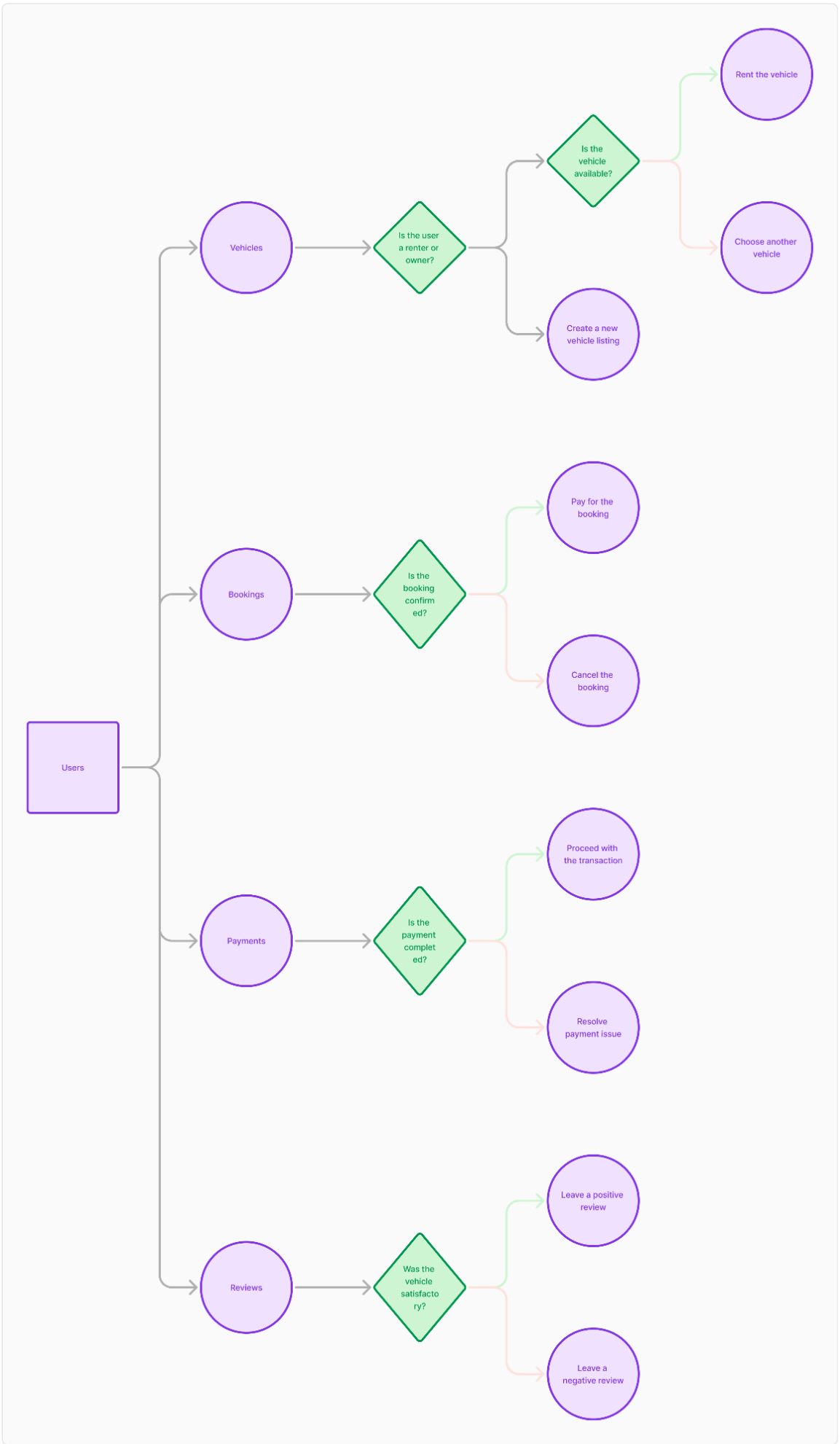
Entities and Fields

1. **Users**
    - **id** (UUID, Primary Key): Unique identifier for the user.
    - **name** (String): Full name of the user.
    - **email** (String, Unique): Email address for authentication.
    - **password** (String): Encrypted password.
    - **phone** (String, Unique): Contact number.
    - **role** (Enum): Defines if the user is a renter or vehicle owner (values: "renter", "owner").
    - **created_at** (Timestamp): Account creation date.
    - **updated_at** (Timestamp): Last profile update date.
2. **Vehicles**
    - **id** (UUID, Primary Key): Unique identifier for the vehicle.
    - **owner_id** (UUID, Foreign Key): References the user who owns the vehicle.

- **type** (Enum): Type of vehicle (e.g., "car", "bike", "van").
- **brand** (String): Vehicle brand.
- **model** (String): Vehicle model.
- **year** (Integer): Year of manufacture.
- **price_per_day** (Decimal): Rental price per day.
- **availability** (Boolean): Whether the vehicle is available for rent.
- **location** (String): Address or city where the vehicle is located.
- **created_at** (Timestamp): Listing creation date.
- **updated_at** (Timestamp): Last update to the listing.

3. **Bookings**
   - **id** (UUID, Primary Key): Unique identifier for the booking.
   - **vehicle_id** (UUID, Foreign Key): References the rented vehicle.
   - **renter_id** (UUID, Foreign Key): References the user who made the booking.
   - **start_date** (Date): Start date of the rental period.
   - **end_date** (Date): End date of the rental period.
   - **total_price** (Decimal): Total cost of the rental.
   - **status** (Enum): Current status of the booking (values: "pending", "confirmed", "cancelled").
   - **created_at** (Timestamp): Booking creation date.
4. **Payments**
   - **id** (UUID, Primary Key): Unique identifier for the payment.
   - **booking_id** (UUID, Foreign Key): References the booking being paid for.
   - **amount** (Decimal): Payment amount.
   - **payment_method** (Enum): Payment method used (e.g., "credit_card", "JazzCash").
   - **status** (Enum): Payment status (values: "pending", "completed", "failed").
   - **created_at** (Timestamp): Payment creation date.
5. **Reviews**
   - **id** (UUID, Primary Key): Unique identifier for the review.
   - **vehicle_id** (UUID, Foreign Key): References the reviewed vehicle.
   - **user_id** (UUID, Foreign Key): References the user who left the review.
   - **rating** (Integer): Rating given (1 to 5).
   - **comment** (Text): Review comments.
   - **created_at** (Timestamp): Review creation date.
   -

Flow chart


Users
Vehicles
Is the user a renter or owner?
Is the vehicle available?
Rent the vehicle
Choose another vehicle
Create a new vehicle listing
Bookings
Is the booking confirmed?
Pay for the booking
Cancel the booking
Payments
Is the payment completed?
Proceed with the transaction
Resolve payment issue
Reviews
Was the vehicle satisfactory?
Leave a positive review
Leave a negative review