Name: MUHAMMAD ABUBAKAR
Roll No: 420397
Slot: Friday 7 to 10

Hackathon Day 2:

Planning the Technical Foundation for a Car Rental E-commerce Platform

Business Objectives Recap (Day 1)

On the first day, I focused on establishing the foundational elements of my car rental platform. Here's a summary of key achievements:

1. **Defining the Mission**
    - Addressed challenges like high car ownership costs and complicated rental processes.
    - Presented the platform as an affordable, flexible, and user-friendly solution for hassle-free car rentals.
2. **Identifying the Target Audience**
    - Primary users include:
        - Busy professionals needing quick rentals.
        - Tourists seeking reliable travel options.
        - Families and individuals requiring vehicles for special occasions.
        - Businesses needing short-term rentals for events or employees.
3. **Highlighting Unique Features**
    - Transparent pricing without hidden charges.
    - Seamless online booking with minimal paperwork.
    - A diverse fleet, including eco-friendly options.
    - 24/7 customer support for a worry-free experience.
4. **Establishing a Data Framework**
    - Designed a clear and concise data model for managing Cars, Customers, and Bookings efficiently.

Day 2: Transitioning to Technical Planning

Step 1: Defining Technical Requirements

Technical requirements are critical for developing a scalable, secure, and user-friendly platform. The primary focus areas include frontend design, backend logic, and database management.

Frontend Requirements

- A visually appealing and intuitive interface for car exploration and booking.
- Fully responsive design optimized for both mobile and desktop devices.
- Core pages include:
    - Homepage
    - Car Listings
    - Car Details
    - Reservation Form
    - Payment Gateway
    - Booking Confirmation

Backend with Sanity CMS

Purpose: Sanity CMS will manage key data such as car details, customer information, and booking records.

Schemas:

1. **Car Details Schema**

    Functionality: Stores car-related information.

    Example Schema:

```
export default {
  name: 'car',
  type: 'document',
  fields: [
    { name: 'name', type: 'string', title: 'Car Name' },
    { name: 'type', type: 'string', title: 'Car Type' },
    { name: 'price', type: 'number', title: 'Price per Day' },
    { name: 'availability', type: 'boolean', title: 'Availability'
},
    { name: 'image', type: 'image', title: 'Car Image' },
  ],
};
```

    Example Data:

```
{
  "name": "Toyota Corolla",
  "type": "Sedan",
  "price": 50,
  "availability": true,
  "image": "https://example.com/car1.jpg"
}
```

2. **Customer Information Schema**

   Functionality: Stores user details for renting cars.

   Example Schema:

```
export default {
name: 'customer',
type: 'document',
fields: [
{ name: 'name', type: 'string', title: 'Full Name' },
{ name: 'email', type: 'string', title: 'Email' },
{ name: 'phone', type: 'string', title: 'Phone Number' },
{ name: 'address', type: 'string', title: 'Address' },
{ name: 'license', type: 'string', title: 'Driving License' },
],
};
```

   Example Data:

```
{
"name": "John Doe",
"email": "john@example.com",
"phone": "+123456789",
"address": "123 Main St, City",
"license": "DL123456789"
}
```

3. **Booking Records Schema**

   Functionality: Tracks rental transactions.

   Example Schema:

```
export default {
  name: 'booking',
  type: 'document',
  fields: [
    { name: 'car', type: 'reference', to: [{ type: 'car' }], title:
'Car' },
    { name: 'customer', type: 'reference', to: [{ type: 'customer'
}], title: 'Customer' },
    { name: 'startDate', type: 'date', title: 'Start Date' },
    { name: 'endDate', type: 'date', title: 'End Date' },
    { name: 'status', type: 'string', title: 'Booking Status' },
  ],
};
```

   Example Data:

```
{
  "car": { "name": "Toyota Corolla" },
```

```
  "customer": { "name": "John Doe" },
  "startDate": "2025-01-20",
  "endDate": "2025-01-25",
  "status": "Confirmed"
}
```

Why Use Sanity CMS?

- **Customization:** Build schemas tailored to platform needs.
- **Real-Time Updates:** Any changes in car or booking details reflect instantly.
- **Scalability:** Easily handles increased data volume.

Third-Party APIs

To enhance functionality, integrate third-party APIs for cars, bookings, payments, and delivery tracking.

API Workflow Examples:

1. **User Registration**
   - New users create accounts, and their details are securely stored in Sanity CMS.
   - A confirmation message is sent for verification.
2. **Car Browsing**
   - Fetch available car data from the CMS.
   - Display cars with filters for easy navigation.
3. **Car Booking**
   - Booking details are submitted to Sanity CMS.
   - Users receive confirmation notifications.
4. **Payment Processing**
   - Payment details are sent to a secure gateway (e.g., Stripe).
   - Users are notified upon successful payment.
5. **Delivery Tracking**
   - Shipment status is fetched from a tracking API (e.g., ShipEngine).
   - Users can monitor the car’s delivery status in real-time.

Step 2: System Architecture

Key Processes:

1. **User Registration:** Account creation with data stored in Sanity CMS.
2. **Car Exploration:** Real-time car data fetching and display with advanced filters.
3. **Booking Management:** User data and car details are securely linked and recorded.

4. **Payment Handling:** Secure transactions with instant confirmations.
5. **Delivery Tracking:** Seamless shipment monitoring.

Step 3: API Endpoints

API Name	Method	Purpose Example	Request/Response
/cars	GET	Fetch all available cars.	**Request:** GET /cars **Response:** [{ "id": 1, "name": "Toyota Corolla", ... }]
/bookings	POST	Create a new booking.	**Request:** POST /bookings **Response:** { "bookingId": 123, "status": "Confirmed" }
/payments	POST	Process payment for a booking.	**Request:** POST /payments **Response:** { "paymentId": 789, "status": "Paid" }
/shipment-tracking	GET	Track the status of a booking.	**Request:** GET /shipment-tracking?bookingId=123 **Response:** { "status": "In Transit", "ETA": "30 mins" }