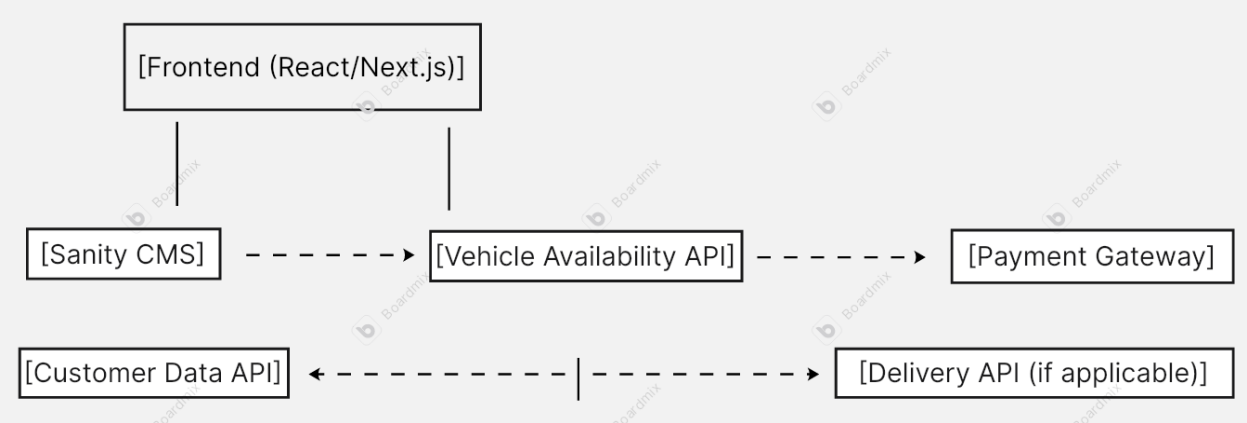
Car Rental Marketplace Technical Foundation

1. System Architecture Overview

The car rental marketplace system comprises several interconnected components: the frontend, the Sanity CMS backend, third-party APIs for car availability and payment processing, and user interaction workflows. Below is the breakdown:

- **Frontend (React/Next.js)**:
  - Responsible for the user interface, including browsing available cars, viewing car details, and managing the booking process.
- **Sanity CMS (Backend)**:
  - Stores car data (make, model, price, availability, etc.), rental policies, customer details, and booking information.
- **Third-Party APIs**:
  - **Payment Gateway API**: Processes payment transactions.
  - **Vehicle Availability API**: Checks availability of cars based on rental dates.
  - **Shipment or Delivery API (if applicable)**: Used for delivery and pick-up services.

A high-level system architecture diagram for the car rental marketplace is as follows:



2. Key Workflows

The following user workflows describe interactions between the components.

1. **User Registration**:
   - User registers with personal information (name, email, driver's license, etc.).
   - Data is stored in Sanity CMS.
   - A confirmation email is sent to the user.

2. **Car Browsing**:
    - User browses available cars by category (e.g., compact, SUV, luxury).
    - Frontend fetches car details from Sanity CMS through API requests.
    - Users can filter by price, car type, and availability.
3. **Booking Process**:
    - User selects a car and specifies rental duration (pickup date, return date).
    - Frontend sends rental details to Sanity CMS to create a booking record.
    - Vehicle Availability API checks if the selected car is available for the chosen dates.
4. **Payment**:
    - User proceeds to checkout and selects payment method.
    - Payment Gateway API securely processes the payment.
    - Payment confirmation is sent to both the user and recorded in Sanity CMS.
5. **Car Delivery or Pickup** (if applicable):
    - After booking, if delivery is selected, the Delivery API is used to track and manage the delivery.
    - User is notified of the expected delivery time.

---

3. Category-Specific Instructions

For a **Car Rental** eCommerce marketplace, we focus on rental duration, vehicle condition, and delivery/pickup management.

- **Rental Duration**: Each car has a rental duration field, which is the time period the car is available for rental.
- **Condition Report**: Each car will have a field indicating its condition upon pickup (e.g., new, good, needs maintenance).
- **Delivery Management**: If delivery is part of the service, we need to manage both pickup and drop-off logistics through a third-party API.

Endpoint	Method	Purpose	Response Example
/cars	GET	Fetch list of available cars	{ "id": 1, "make": "Toyota", "model": "Camry", "price": 50, "availability": true }
/cars/{id}	GET	Fetch details of a specific car	{ "id": 1, "make": "Toyota", "model": "Camry", "price": 50, "availability": true, "condition": "New" }
/bookings	POST	Create a new car rental booking	{ "userId": 123, "carId": 1, "startDate": "2025-01-20", "endDate": "2025-01-22", "paymentStatus": "Pending" }
/payment	POST	Process payment for booking	{ "bookingId": 456, "paymentMethod": "Credit Card", "status": "Success" }
/delivery	GET	Track car delivery or pickup status	{ "orderId": 789, "status": "In Transit", "ETA": "2 hours" }

5. Sanity Schema Example

The Sanity schema will include entities like `Car`, `Booking`, and `User`. Here's an example schema for `Car`:

```
export default {
  name: 'car',
  type: 'document',
  fields: [
    { name: 'make', type: 'string', title: 'Car Make' },
    { name: 'model', type: 'string', title: 'Car Model' },
    { name: 'price', type: 'number', title: 'Price per Day' },
    { name: 'availability', type: 'boolean', title: 'Car Availability' },
    { name: 'condition', type: 'string', title: 'Condition' },
    { name: 'images', type: 'array', of: [{ type: 'image' }], title: 'Car
Images' }
  ]
};
```

---

6. Technical Roadmap

Phase 1: Initial Setup and Prototyping

- Setup Sanity CMS and create the car schema.
- Develop basic frontend structure using React/Next.js.
- Set up the basic API endpoints for car listings and booking.

Phase 2: API Integration

- Integrate the Vehicle Availability API to check car availability.
- Implement the Payment Gateway API for processing transactions.

Phase 3: Testing and Validation

- Test all workflows, including browsing cars, creating bookings, and processing payments.
- Ensure real-time updates on booking status and car availability.

Phase 4: Launch

- Finalize deployment.
- Monitor and resolve any issues with real-time data flows.