Technical Requirements for Car Rental Website

1. Frontend Requirements:

- **User-Friendly Interface:**
    - Clean, easy-to-use UI that allows users to browse available cars, view car details, and select rental durations.
- **Responsive Design:**
    - Optimized for both desktop and mobile users.
- **Essential Pages:**
    - **Home Page**: Displays featured cars, search bar, promotions, and rental offers.
    - **Car Listing Page**: Shows a list of available cars, search filters for car types, brands, and availability dates.
    - **Car Details Page**: Shows individual car details (e.g., car features, price per day, availability).
    - **Cart Page**: Allows users to add selected cars and rental durations.
    - **Checkout Page**: Handles the booking process, including rental details, user information, payment, and confirmation.
    - **Order Confirmation Page**: Shows booking summary, confirmation number, and rental details.

---

2. Backend (Sanity CMS as Backend):

- **Sanity CMS** will act as the backend to manage:
    - **Car Data**: Store information about available cars, brands, rental prices, descriptions, availability, etc.
    - **Customer Data**: Manage user profiles, rental history, contact information.
    - **Order Data**: Manage bookings, payment records, and rental statuses.

Sanity CMS Schema Example:

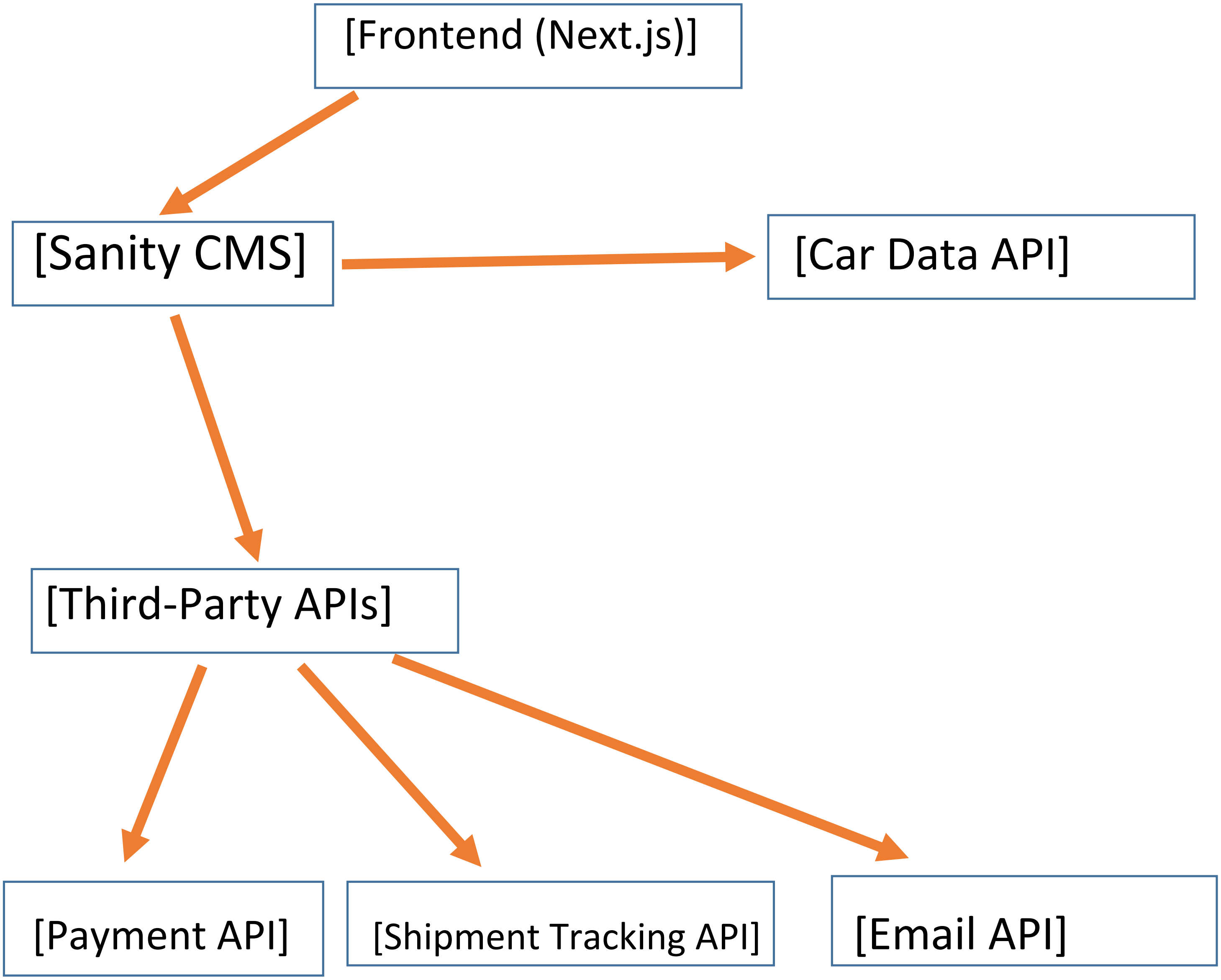
```
// Product (Car)

 export default { name: 'car', type: 'document', fields: [

 { name: 'name', type: 'string', title: 'Car Name' },

 { name: 'brand', type: 'string', title: 'Car Brand' },

{ name: 'price', type: 'number', title: 'Price per day' },

 { name: 'availableFrom', type: 'date', title: 'Available From' },

{ name: 'availableTo', type: 'date', title: 'Available To' },

 { name: 'description', type: 'text', title: 'Car Description' },
{ name: 'image', type: 'image', title: 'Car Image' },

 { name: 'stock', type: 'number', title: 'Stock Level' }

// Number of cars available ] }; // Rental Order

 export default { name: 'order', type: 'document', fields: [

 { name: 'user', type: 'reference', to: [{ type: 'user' }] },

{ name: 'car', type: 'reference', to: [{ type: 'car' }] },

 { name: 'rentalDuration', type: 'number', title: 'Rental Duration (Days)' },

 { name: 'depositAmount', type: 'number', title: 'Deposit Amount' },

{ name: 'rentalStartDate', type: 'date', title: 'Rental Start Date' },

 { name: 'rentalEndDate', type: 'date', title: 'Rental End Date' },

 { name: 'status', type: 'string', title: 'Order Status' }, ] };
```

3. Third-Party APIs:

- **Payment Gateway API**: To handle secure transactions (e.g., Stripe, PayPal).
- **Shipment Tracking API**: For updating users on the delivery and collection of vehicles (could be integrated with a car fleet tracking system).
- **Email/SMS API**: For sending order confirmations, rental reminders, and customer support notifications.

System Architecture Overview

Here’s the architecture flow for your car rental website:



1. **User Browsing Cars:**
    - The user visits the website’s car listing page.
    - The frontend sends a request to the **Car Data API** (Sanity CMS) to fetch available cars.
    - Car data (name, price, description, stock) is returned and displayed.
2. **User Adds Car to Cart and Proceeds to Checkout:**
    - User selects a car and rental duration.
    - The system calculates the total rental price, including the deposit.
    - Rental details are sent to **Sanity CMS**, and the order is saved in the database.
3. **User Makes Payment:**
    - User is redirected to a **Payment Gateway API** to process the payment securely.
    - Payment confirmation is sent back to the website and stored in **Sanity CMS**.
4. **User Receives Confirmation:**
    - After successful payment, the **Order Confirmation Page** displays a summary with order details.
    - A confirmation email (or SMS) is sent to the user.
5. **Car Delivery and Pickup:**
    - The **Shipment Tracking API** updates the user on the delivery/pickup status of the vehicle.

Key Workflows

1. **User Registration:**
    - **User Sign-Up**: User provides details (name, email, password), data is stored in **Sanity CMS**.
    - **Email Confirmation**: A confirmation email is sent to the user upon successful registration.
2. **Car Browsing:**
    - User views available cars by filtering on the website (brand, price range, availability).
    - The **Car Data API** fetches product details from **Sanity CMS** and displays them.
3. **Rental Booking:**
    - User selects a car and rental duration.
    - The order is stored in **Sanity CMS** with details such as rental duration, deposit, and rental dates.
4. **Payment Processing:**
    - User proceeds to payment via **Payment Gateway API**.
    - After a successful transaction, the payment status is updated in **Sanity CMS**.
5. **Shipment Tracking:**
    - The system fetches real-time vehicle delivery or pickup status via the **Shipment Tracking API** and updates the user.

API ENDPOINTS:

End points	**Method**	**Purpose**	**Response Example**
/cars	GET	Fetch all available cars	{ "id": 1, "name": "Toyota Corolla", "price": 50, "availableFrom": "2025-02-01", "availableTo": "2025-02-10" }
/order	POST	Create a new rental order	{ "confirmationId": 123, "status": "Success", "rentalDuration": 7 }
/payment	POST	Process payment transaction	{ "transactionId": "abc123", "status": "Success", "amount": 350 }
/shipment/status	GET	Fetch shipment status for rental car	{ "status": "In Transit", "expectedDelivery": "2025-02-08" }