DanyFur Dream

Online Furniture Market Business

Step 1: Define Technical Requirements

Frontend Requirements ➢

User Interface:

- User-friendly and visually appealing interface for browsing furniture products.
- Responsive design for mobile and desktop users.

➢ **Essential Pages**:

- **Home:** Showcase featured furniture and categories.
- **Product Listing:** Display furniture items with filtering and sorting options.
- **Product Details:** Detailed descriptions, images, and purchase options.
- **Cart:** Display selected items with prices.
- **Checkout:** Capture user information and payment details.
- **Order Confirmation:** Show order summary and confirmation message.

Backend Requirements Using Sanity CMS

1. **Sanity CMS for Data Management**:

Use **Sanity CMS** as the primary backend to manage and store the following:

- **Product Data**: Titles, descriptions, images, prices, and stock levels.
- **Customer Details**: Names, addresses, contact information, and order histories.
- **Order Records**: Order IDs, statuses, timestamps, and payment confirmations.

2. **Schema Design in Sanity**:

- Define schemas that reflect the marketplace structure, including products, customers, and orders.
- Align schema fields with Day 1 business goals to enable seamless integration.

Third-Party APIs for Core Functionality

1. **Shipment Tracking Integration**:

 Implement APIs from leading logistics providers to track shipments. Display real-time delivery status updates to customers.

2. **Payment Gateway Integration**:

- Integrate secure and reliable payment gateways (e.g., Stripe, PayPal).
- Enable seamless online payments and support multiple payment methods.

3. **Backend Service APIs**:

 Utilize APIs to enhance backend functionality, such as tax calculation, discount application, and inventory synchronization.

4. **API Reliability**:

 Ensure all APIs provide the necessary data to power frontend features, such as live inventory updates, pricing adjustments, and shipping estimates.

Step 2: Design System Architecture

System Components

1. **Frontend (Next.js)**

- Provides a responsive and interactive user interface.
- Dynamically fetches and displays data such as products and order statuses.

2. **Sanity CMS**

- Serves as the backend for data management.
- Handles product information, user data, and order records.
- Receives and stores product data via an external API

3. **Third-Party APIs**

- **Product Data API**: Supplies product information to be stored in **Sanity CMS**.
- **Shipment Tracking API**: Updates users on the status of their deliveries.
- **Payment Gateway**: Processes secure online transactions.

Key Workflows

1. **User Registration**

- The user signs up via the frontend.
- User data is stored in **Sanity CMS**.
- A confirmation email is sent to the user.

2. **Product Data Fetching and Storage**

- The **Product Data API** supplies updated product data, including details such as names, categories, images, and prices.
- This data is fetched and stored in **Sanity CMS**, ensuring a centralized and manageable product database.
- The **Sanity API** provides product data to the frontend dynamically.

3. **Product Browsing**

- The user views product categories on the frontend.
- The **Sanity API** fetches product data from the **CMS**.
  → Products are displayed dynamically on the site.

4. **Order Placement**

- The user adds items to the cart and proceeds to checkout.
- Order details (e.g., product IDs, quantities, and user information) are sent to **Sanity CMS** for record-keeping.

5. **Shipment Tracking**

- Order status updates are fetched via a **Third-Party Shipment API**.
- Tracking details are displayed in real-time on the frontend.

6. **Payment Processing**

- Payment details are securely sent to a **Payment Gateway**.
- Confirmation of the transaction is sent back to the user and recorded in **Sanity CMS**.

Data Flow Description

1. **Product Data Management**:

- Product data is fetched from an external **Product Data API**.
- This data is stored in **Sanity CMS**, making it available for dynamic display on the frontend.

2. **Frontend Interaction**:

 Users browse, search, and interact with the marketplace via the **Next.js frontend**.

3. **Order Handling**:

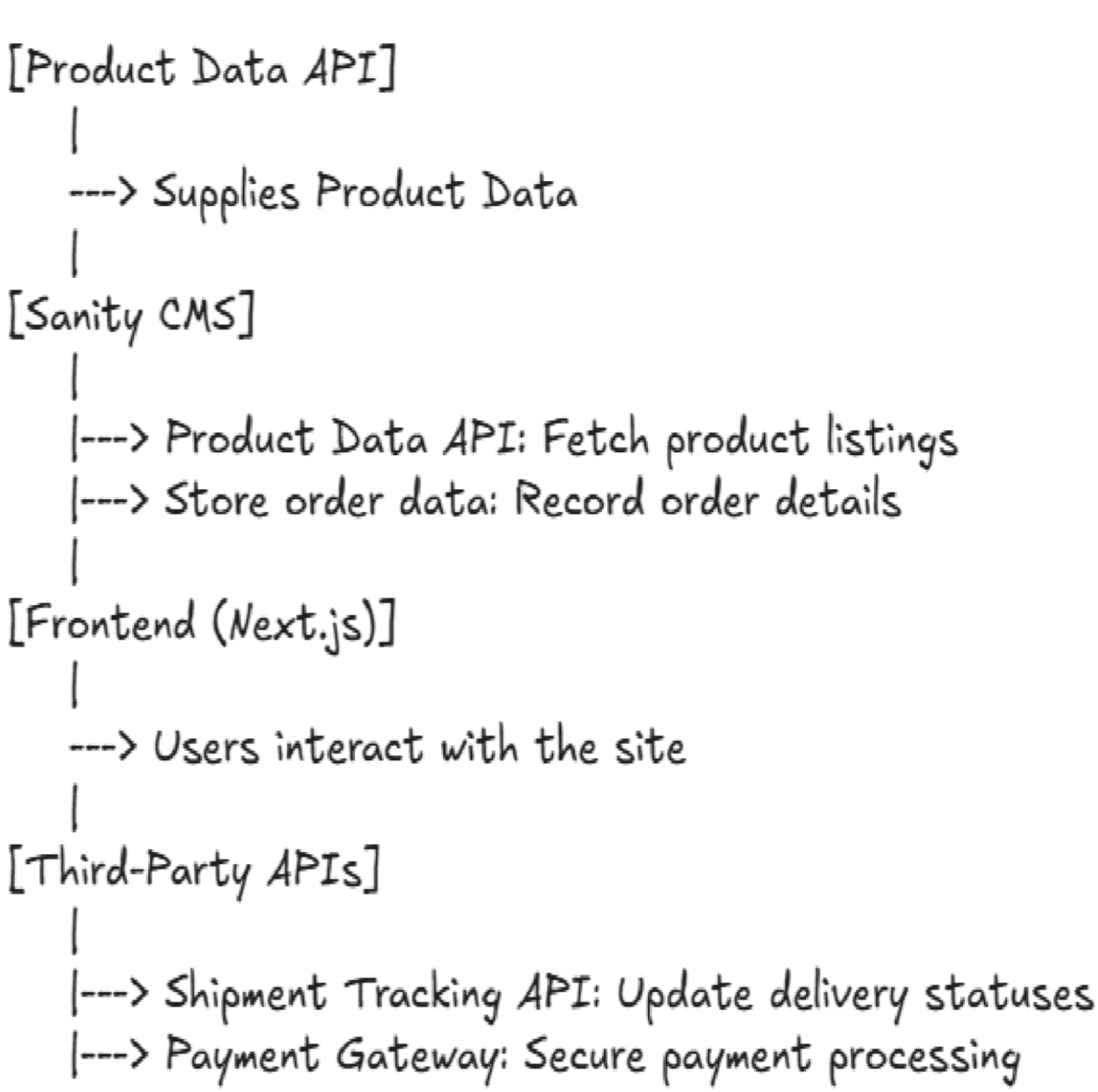
 Order details are stored in **Sanity CMS**. Shipment details and payment statuses are integrated seamlessly.

4. **Real-Time Updates**:

   - **Third-Party APIs** provide shipment tracking, while the **Payment Gateway** handles secure transactions.

Flowchart Diagram

The following diagram visualizes the system's architecture and data flow between its components:



Key Benefits of the Architecture

1. **Centralized Product Management**: Product data from external sources is organized and managed within **Sanity CMS**.
2. **Scalability**: Modular design ensures ease of adding new features or APIs.
3. **Efficiency**: Clear workflows reduce redundancy and ensure smooth operation.
4. **User Experience**: Responsive frontend and real-time updates enhance customer satisfaction.
5. **Security**: Payment gateways provide secure transaction handling.

Step 3: Plan API Requirements

API Design Principles

1. Use RESTful APIs for efficient communication.
2. Ensure security with HTTPS and API authentication.
3. Provide comprehensive API documentation for easy integration.

Defined API Endpoints

1. **Fetch All Products**

   Endpoint Name: /products

   Method: GET

   Description: Retrieve all furniture products from **Sanity CMS**.

   Response:

```
[
  { "id": 101, "name": "Dining Table", "price": 15000, "stock": 10, "image": "URL", "category": "Furniture" },
  { "id": 102, "name": "Sofa Set", "price": 30000, "stock": 5, "image": "URL", "category": "Furniture" }
]
```

2. **Create Order**

 Endpoint Name: /orders

 Method: POST

 Description: Save order details in **Sanity CMS** for tracking and fulfillment. **Payload**:

```
{
  "customer": { "id": 301, "name": "Ali", "email": "ali@example.com" },
  "products": [{ "id": 101, "quantity": 1 }],
  "deliveryAddress": "Karachi, Pakistan",
  "paymentStatus": "Paid"
}
```

3. **Track Shipment**

 Endpoint Name: /shipment

 Method: GET

 Description: Fetch real time-delivery status for furniture items.

 Response:

```
{
  "shipmentId": "SHIP789",
  "orderId": "ORD456",
  "status": "Out for Delivery",
  "expectedDeliveryDate": "2025-01-20"
}
```

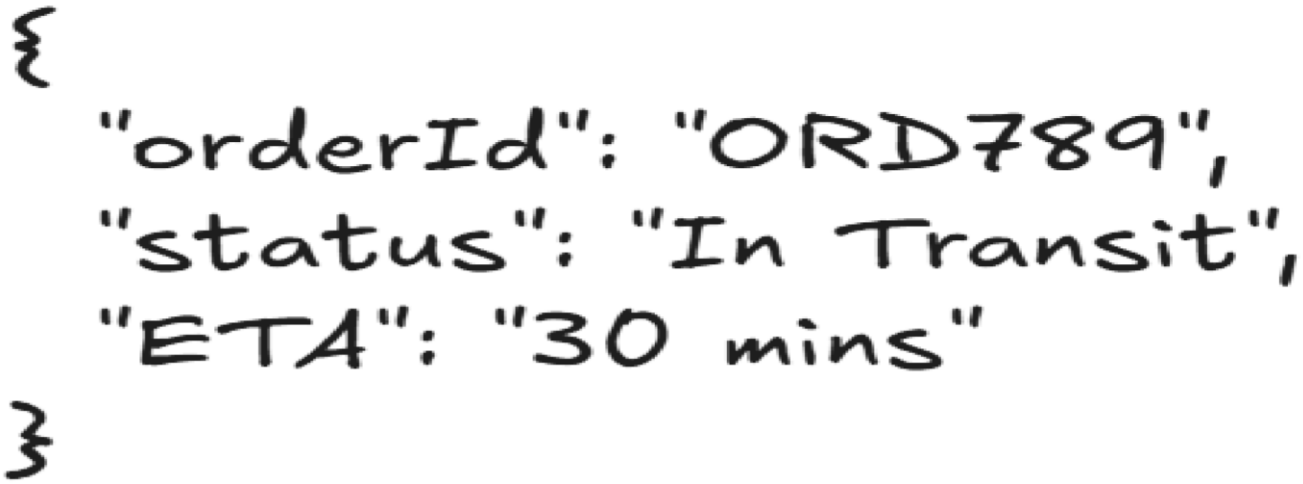
4. **Quick Delivery Status**

 Endpoint Name: /quick-delivery-status

 Method: GET

 Description: Provide real-time updates for express furniture delivery.

 Response:



API Workflow Summary

1. **Product Data**:

    Furniture items are managed in **Sanity CMS** and fetched via /products for frontend display.

2. **Order Placement**:

    Customers place orders through /orders, saving data in **Sanity CMS**.

3. **Shipment Tracking**:

    Real-time shipment tracking is handled via /shipment or /quick-delivery-status for Q-Commerce.

4. **Payment**:

    Secure payments processed via a **Payment Gateway**, updating order status in **Sanity CMS**.

5. **Quick Delivery (Q-Commerce)**:

- Specialized endpoint /quick-delivery-status ensures users get rapid updates on furniture delivery status.