Marketplace Technical Foundation - System Architecture (Day 2)

Overview

This document details the system architecture, key workflows, and API requirements for the Furniture E-Commerce Marketplace. It serves as a foundational guide for the implementation phase.

---

1. System Architecture Design

Diagram

```
[Frontend (Next.js)]
    |
[Sanity CMS] ---------> [Product Data API]
    |
[Third-Party API] -----> [Shipment Tracking API]
    |
[Payment Gateway]
```

Component Roles:

- **Frontend (Next.js):** Serves as the user interface, optimized for both desktop and mobile responsiveness.
- **Sanity CMS:** Manages dynamic content, such as product data, customer profiles, and order records.
- **Third-Party APIs:** Supports logistics (shipment tracking) and real-time communication between the platform and external services.
- **Payment Gateway:** Processes secure financial transactions.

---

2. Key Workflows

User Browsing Workflow

1. User navigates to the marketplace.\n2. Frontend fetches and displays product data via API.\n3. Dynamic filtering allows users to explore specific categories.

Order Placement Workflow

1. User adds items to the cart and proceeds to checkout.\n2. Order details are securely stored in Sanity CMS.\n3. Payment is processed via the payment gateway.\n4. Confirmation is sent to the user.

Shipment Tracking Workflow

1. Order ID is linked with shipment data through APIs.\n2. Users view real-time updates on their order status.

---

3. API Endpoint:

Here is the list of possible API endpoints required for Day 2:

1. **GET /api/products**
    - **Description**: Fetch all available products.
    - **Request Parameters**: None.
    - **Response**:

```
[
    {
        "id": "123",
        "name": "Sofa",
        "price": 999,
        "stock": 15
    }
]
```

2. **POST /api/cart**

- **Description: Add an item to the cart.**
- **Request Body:**

```
{
    "productId": "123",
    "quantity": 1
```

}

Response:

```
{
  "message": "Item added to cart",
  "cart": [
    {
      "productId": "123",
      "quantity": 1
    }
  ]
}
```

3. POST /api/orders

- **Description: Create a new order.**
- **Request Body:**

```
{
  "cart": [
    {
      "productId": "123",
      "quantity": 1
    }
  ],
  "customer": "Jane Doe",
  "totalAmount": 999
}
```

Response:

```
{
  "message": "Order created",
  "orderId": "ORD001"
}
```

Sanity CMS Schema Updates

Product Schema

```
export default {
  name: 'product',
  type: 'document',
  fields: [
    { name: 'name', type: 'string', title: 'Product Name' },
    { name: 'price', type: 'number', title: 'Price' },
    { name: 'description', type: 'text', title: 'Description' },
    { name: 'stock', type: 'number', title: 'Stock Level' },
    { name: 'image', type: 'image', title: 'Product Image' }
  ]
};
```

Order Schema

```
export default {
  name: 'order',
  type: 'document',
  fields: [
    { name: 'orderId', type: 'string', title: 'Order ID' },
    { name: 'customer', type: 'string', title: 'Customer Name' },
    { name: 'items', type: 'array', of: [{ type: 'reference', to: [{ type: 'product' }] }], title: 'Items' },
```

{ name: 'totalAmount', type: 'number', title: 'Total Amount' },

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

]

};

Testing Plan

1. **Use Postman to test endpoints:**
    - **Validate request and response formats.**
    - **Test edge cases like invalid product IDs and empty carts.**
2. **Integrate mocked API data with the frontend.**

Deliverables by End of Day 2

1. **API endpoints for products, cart, and orders fully functional.**
2. **Sanity CMS updated with relevant schemas.**
3. **All endpoints tested and ready for frontend integration.**

---

4. Goals and Achievements

1. **Technical Blueprint:** Defined all foundational components for development.\n2. **API Documentation:** Outlined endpoints for smooth backend/frontend integration.\n3. **Alignment with Business Goals:** Ensured that workflows support customer experience and operational scalability.