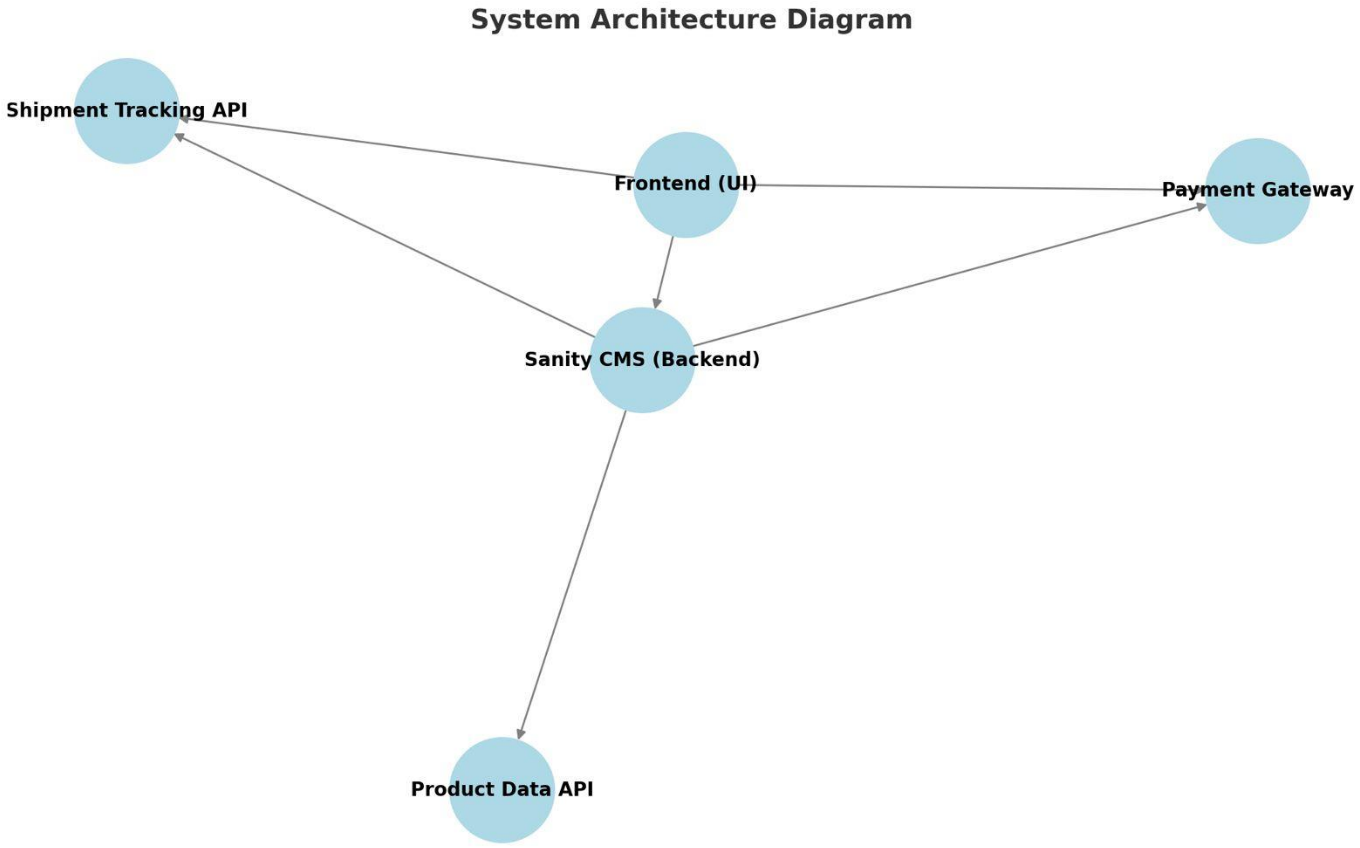
1. **System Architecture Diagram**
   I'll create a simple diagram showing the interactions between components.
2. **Technical Documentation**
   I'll write the required document, including workflows, API requirements, and an overview of Sanity schemas.
3. **Sanity Schema Draft**
   I'll prepare example schemas for products, orders, and customers.
4. **Submission Files**
   I'll organize everything into appropriately named files for submission.

Let’s start! I’ll create the system architecture diagram first.



Here is the **System Architecture Diagram** showing the interaction between the components:

- **Frontend (UI):** The user-facing interface where customers browse products, place orders, and track shipments.
- **Sanity CMS (Backend):** Manages product data, customer details, and orders.
- **Product Data API:** Fetches product details for display on the frontend.
- **Shipment Tracking API:** Provides real-time order delivery updates.
- **Payment Gateway:** Processes secure online payments.

Next, I'll create the technical documentation.

Marketplace Technical Foundation - Handmade Crafts Marketplace

1. System Architecture Overview

The architecture consists of the following key components:

- **Frontend (UI):** Built with a framework like Next.js to provide a responsive user experience for browsing handmade crafts, placing orders, and tracking shipments.
- **Sanity CMS (Backend):** Acts as the backend to manage product data, customer details, and order records.
- **Third-Party APIs:**
    - **Shipment Tracking API:** Fetches real-time delivery updates for orders.
    - **Payment Gateway:** Processes secure payment transactions.

Data Flow:

1. Users interact with the **Frontend** to browse products, add items to their cart, and complete orders.
2. The **Frontend** communicates with **Sanity CMS** via APIs for product listings, customer details, and orders.
3. **Third-Party APIs** handle shipment tracking and payment processing.

Key Workflows

1. **Browsing Products:**
    - User visits the homepage.
    - Frontend sends a request to the Product Data API (via Sanity CMS).
    - Products are displayed dynamically.
2. **Placing an Order:**
    - User adds items to the cart and proceeds to checkout.
    - Order details are sent to Sanity CMS via an API.
    - Payment is processed via the Payment Gateway.
    - Confirmation is sent to the user.
3. **Shipment Tracking:**
    - Shipment status is fetched via the Shipment Tracking API.
    - Real-time updates are displayed on the frontend.

4. API Requirements

Endpoint Name	Method	Description	Payload/Response
`/products`	`GET`	Fetch all available products from Sanity CMS.	**Response:** `{ "id": 1, "name": "Vase", "price": 20, "stock": 10 }`
`/orders`	`POST`	Create a new order in Sanity CMS.	**Payload:** `{ "customerId": 123, "productId": 456, "quantity": 2, "total": 40 }`
`/shipment`	`GET`	Track order status via third-party API.	**Response:** `{ "orderId": 789, "status": "In Transit", "ETA": "3 days" }`

Sanity Schema Example

Product Schema:

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

Order Schema:

```
export default {
  name: 'order',
  type: 'document',
  fields: [
    { name: 'customerId', type: 'string', title: 'Customer ID' },
    { name: 'productId', type: 'string', title: 'Product ID' },
    { name: 'quantity', type: 'number', title: 'Quantity' },
    { name: 'total', type: 'number', title: 'Total Price' },
    { name: 'status', type: 'string', title: 'Order Status' }
  ]
};
```