**Marketplace Technical Foundation – Q-Commerce for Medication and Healthcare**

**Step 4.1: Goal Explanation**

*The goal of Day 2 is to transition from planning the business aspects of our Q-Commerce project to creating the technical foundation for coding. This includes designing the system architecture, outlining workflows, drafting API endpoints, and building schemas for Sanity CMS. The output will serve as the blueprint for the project’s future development.*

**Step 4.2: Technical Requirements**

**Frontend Requirements:**

* Framework: Next.js
* Key Features:
  + Pages: Homepage, product listing, cart, checkout, etc.
  + Mobile-first design.

**Backend Requirements (Sanity CMS):**

* Manage product inventory, user details, orders, and deliveries.
* Example schema fields:
  + Products: name, price, stock.
  + Users: name, address.

**Third-Party APIs:**

* Payment gateway (e.g., Stripe).
* Delivery tracking (real-time order status).

**Step 4.3: System Architecture**

**Diagram**

* The diagram demonstrates how the system components work together. Here’s an outline for your architecture:

[Frontend (Next.js)]

↕︎

[Sanity CMS] ↔ [Products Database]

↕︎

[Third-Party APIs] ↔ [Payment Gateway + Delivery Tracking]

**Description**

SYSTEM ARCHITECTURE

**User**

**Shop**

**Product**

**Cart**

**Checkout**

**Product**

**&**

**Order Date**

**Tracks**

**Order**

**Delivery**

**Payment**

**[Paypal, Easypaisa]**

**Fetched**

**Data**

**FRONTEND UI**

**SANITY CMS**

**THIRD PARTY APIs**

**SHIPMENT APIs**

**DATABASE**

* **Frontend**: Built using Next.js for product browsing, order placement, and tracking deliveries.
* **Backend (Sanity CMS)**: Manages the database for storing products, users, orders, payments, and delivery records.
* **Products Database**: Handles all product categories (e.g., perishables, pantry staples) and ensures freshness by tracking expiration dates.
* **Third-Party APIs**: Interfaces for external services like payments and live delivery tracking.

**Steps to Add This Diagram:**

1. Use a tool like **Figma**, **Lucidchart**, or even a hand-drawn sketch. Save the diagram file.
2. Insert the image in your document with the title **System Architecture Diagram**.

**Step 4.4: API Planning**

Add these API endpoint examples:

|  |  |  |  |
| --- | --- | --- | --- |
| Endpoint | Method | Description | Example Response |
| /products | GET | Fetch all products by category | { "id": 1, "name": "Apples", "category": "Fresh Produce", "price": 5.99, "expiration\_date": "2025-01-30" } |
| /orders | POST | Place a new order | { "orderId": 123 } |
| /delivery-status | GET | Check delivery progress | { "orderId": 123, "ETA": "30 mins", "status": "Out for Delivery" } |
| /payment | POST | Process a payment | { "status": "Successful" } |

**How to Add This Section:**

* Write the endpoints table in your document, providing both examples and a brief explanation for each.
* Describe the role of APIs in facilitating communication between the frontend, backend, and external services.

**Step 4.5: Key Workflows**

Explain the following workflows clearly:

1. **Browsing Products**
   * The user visits the homepage and navigates by category (e.g., Fresh Produce, Pantry Items).
   * The frontend retrieves product data from the Sanity CMS.
   * Products are displayed dynamically, including stock levels and expiration dates for perishable items.
2. **Placing an Order**
   * The user selects products, adds them to the cart, and proceeds to checkout.
   * On order submission, the backend updates the product stock and records the order in Sanity CMS.
3. **Delivery Tracking**
   * The user checks their order status through the frontend.
   * The frontend fetches delivery data via a Third-Party API.
   * The user sees live updates (e.g., ETA: 30 minutes, Out for Delivery).
4. **Payment Processing**
   * Payment information is submitted through the frontend.
   * A Third-Party Payment Gateway processes the transaction.
   * A success or failure message is returned, and the backend updates the payment record.

**Step 4.6: Sanity CMS Code Writing**

Provide schemas for products and orders as examples.

**Product Schema:**  
Save this file as productSchema.js in your code folder.

export default {

name: 'product',

type: 'document',

fields: [

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

{ name: 'category', type: 'string', title: 'Category' },

{ name: 'description', type: 'string', title: 'Description' },

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

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

{ name: 'expiration\_date', type: 'datetime', title: 'Expiration Date' },

],

};

**Order Schema:**  
Save this file as orderSchema.js in your code folder.

export default {

name: 'order',

type: 'document',

fields: [

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

{ name: 'product\_ids', type: 'array', of: [{ type: 'reference', to: [{ type: 'product' }] }] },

{ name: 'order\_date', type: 'datetime', title: 'Order Date' },

{ name: 'delivery\_date', type: 'datetime', title: 'Delivery Date' },

{ name: 'order\_status', type: 'string', title: 'Order Status', options: { list: ['Pending', 'Shipped', 'Delivered', 'Canceled'] } },

{ name: 'total\_price', type: 'number', title: 'Total Price' },

],

};