Marketplace Builder Hackathon 2025 (Day-2)

General E-Commerce Marketplace Plan

Created by Ismat Fatima

Objective

Frontend Requirements:

- User-friendly interface for browsing food items.
- Responsive design for both mobile and desktop users.
- Essential pages: Home, Menu Listing, Item Details, Cart, Checkout, and Order Confirmation.

Backend Requirements using Sanity CMS:

- Manage food items, customer details, and order records.
- Design schemas in Sanity to align with the business goals of quick delivery.

Third-Party APIs:

- Integrate APIs for delivery tracking, payment gateways, and other necessary backend services

---

System Architecture Diagram

Graph TD:

Design System Architecture

Let's create a diagram showing how different components interact. Here's a high-level overview:

[Frontend (Next.js)]

|

[Sanity CMS] ----------> [Food Data API]

|

[Third-Party API] -----> [Delivery Tracking API]

|

[Payment Gateway]

In this architecture:

- The frontend (Next.js) interacts with the Sanity CMS for managing food data.

- The delivery tracking and payment processing are handled through third-party APIs.

Features & Workflow

Key Workflows:

1. **User Registration and Login**:
   - **Step 1**: User registers or logs in through the frontend.
   - **Step 2**: User data is stored in Sanity CMS.
   - **Step 3**: User receives a confirmation email (optional).

2. **Browsing Food Menu**:
   - **Step 1**: User browses food categories on the website.
   - **Step 2**: Frontend fetches food data from the Sanity CMS via the Food Data API.
   - **Step 3**: Display food items to the user.
3. **Placing an Order**:
   - **Step 1**: User selects food items and adds them to the cart.
   - **Step 2**: User proceeds to checkout and confirms the order.
   - **Step 3**: Order details are sent to Sanity CMS for storage.
   - **Step 4**: Payment Gateway processes the payment.
   - **Step 5**: Order confirmation is displayed to the user.
4. **Delivery Tracking**:
   - **Step 1**: After the order is placed, the delivery status is updated via the Delivery Tracking API.
   - **Step 2**: User can track the order status on the website in real-time.
   - **Step 3**: Delivery information is fetched and displayed to the user.

API Requirements

Endpoint	Method	Description
/api/products	GET	Fetch product data from Sanity CMS.
/api/shipping-label	POST	Generate a shipping label using ShipEngine.
/api/track-order	GET	Retrieve order status using ShipEngine label ID.
/api/checkout	POST	Integrate Stripe for payment processing.

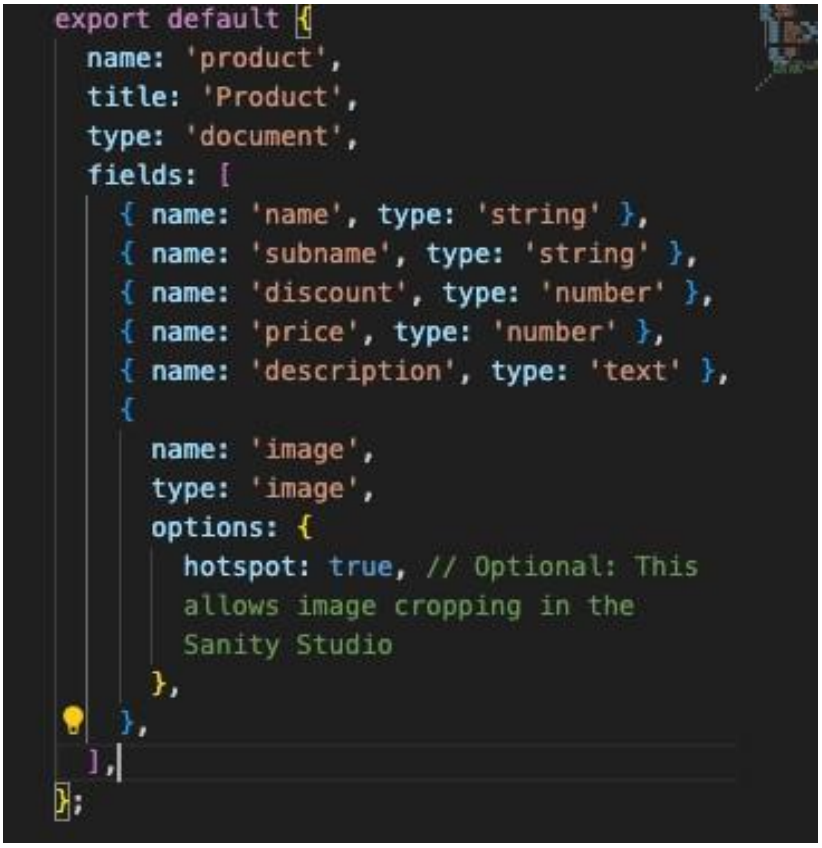
Tools & Libraries

- **Clerk:** Authentication.
- **Sanity CMS:** Content management.
- **ShipEngine API:** Shipping and tracking..
- **React Context API:** Cart functionality.
- Delivery/ Tracking

Data Models

Sanity Schemas

Product Schema:



```
export default {
  name: 'product',
  title: 'Product',
  type: 'document',
  fields: [
    { name: 'name', type: 'string' },
    { name: 'subname', type: 'string' },
    { name: 'discount', type: 'number' },
    { name: 'price', type: 'number' },
    { name: 'description', type: 'text' },
    {
      name: 'image',
      type: 'image',
      options: {
        hotspot: true, // Optional: This
        allows image cropping in the
        Sanity Studio
      },
    },
  ],
};
```

Deliverables

1. **System Architecture Diagram:** Shows component interaction.
2. **Sanity Schemas:** For products and orders.
3. **API Endpoints:** For Delivery, tracking, and payments.
4. **Frontend Pages:** Authentication, product browsing, cart management, and order confirmation.
5. **Portfolio-Ready Submission:** Polished project showcasing full-stack Q-commerce skills.