Day 2 Hackathone

Market Place : Rental E Commerce

**Frontend Requirements**

* **User Interface (UI)**:
  + Design a user-friendly interface to allow customers to browse, select, and purchase/rent items effortlessly.
  + Include pages such as:
    - Home Page.
    - Product Listings.
    - Product Details.
    - Cart.
    - Checkout.
    - Order Confirmation.
  + Ensure a **responsive design** for both mobile and desktop users.

**Backend Requirements (Sanity CMS)**

* Use **Sanity CMS** for managing data such as:
  + Product inventory.
  + Customer details.
  + Order records.
* Design schemas for:
  + Products (name, price, stock, category).
  + Orders (customer info, product details, status).
  + Shipment tracking (order ID, status).

**Third-Party API Integrations**

* Integrate APIs for:
  + Payment gateways to handle secure transactions (e.g., Stripe, PayPal).
  + Shipment tracking to provide real-time order status updates.
  + Any other essential functionality specific to your marketplace type.

### **Design System Architecture**

Draw a high-level diagram to represent how different components interact in your system. Here’s an example structure:

#### ****System Architecture Diagram (Example)****

* **Frontend**: Built using Next.js or a similar framework for a seamless user experience.
* **Sanity CMS**: Handles backend data storage and management.
* **Third-Party APIs**:
  + Payment Gateway for processing payments.
  + Shipment Tracking API for order updates.

#### ****Workflow Example****:

1. **Product Browsing**:
   * Frontend fetches data via the Sanity CMS API to display product listings.
2. **Order Placement**:
   * Customer places an order → Frontend sends data to Sanity CMS → Order recorded.
3. **Shipment Tracking**:
   * Frontend retrieves shipment status from the tracking API and displays it to the customer.

#### ****Plan API Endpoints****

Define key API endpoints based on your marketplace’s requirements:

#### ****Example Endpoints****:

1. **Fetch Products**
   * Endpoint: /products
   * Method: **GET**
   * Description: Retrieve all available products from Sanity CMS.
   * Response

{

"id": 1,

"name": "Product A",

"price": 100,

"stock": 20

}

**Create Order**

* Endpoint: /orders
* Method: **POST**
* Description: Save order details in Sanity CMS.
* Payload:

{

"customerName": "Rameez",

"productId": 1,

"quantity": 2,

"totalAmount": 200

}

{

"orderId": 101,

"status": "Success"

}

**Track Shipment**

* Endpoint: /shipment
* Method: **GET**
* Description: Fetch real-time order shipment status using a third-party API.
* Response:

{

"shipmentId": "ABC123",

"status": "In Transit",

"ETA": "2 hours"

}

#### ****Structure of the Documentation****:

1. **System Architecture Overview**:
   * Include a diagram (hand-drawn or software-based).
   * Briefly explain the interaction between components (Frontend, Sanity CMS, APIs).
2. **Workflow Descriptions**:
   * Provide step-by-step details for key workflows:
     + Product Browsing: User → Sanity CMS → Products displayed.
     + Order Placement: User → Order saved in Sanity CMS → Confirmation sent.
     + Shipment Tracking: User → Shipment data retrieved from API → Displayed in UI.
3. **API Specifications**:
   * List endpoints with details (name, method, payload, response).

**Sanity Schema Example**:

* Example schema for **Products**:

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' },

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

],

};