* **HACKATHON 3 (DAY 2)**
* **Marketplace Technical Foundation**

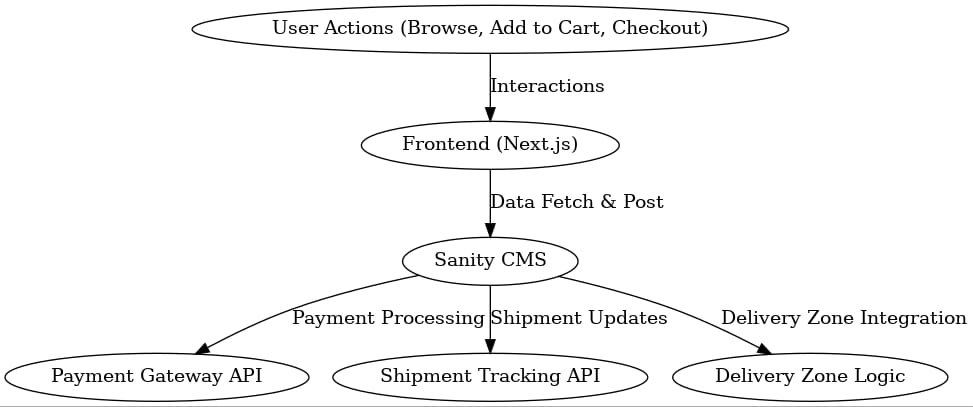
**|Marketplace Workflow Overview|**

**User Browses Products:** The user accesses the marketplace and browses the available products. They can view product details, add items to the cart, and proceed to checkout.

**User Places an Order:** After selecting products, the user places an order. The order is confirmed with customer and product details, along with the total price and shipping address.

**Payment Gateway and Delivery Zone:** The order is processed through the payment gateway, and a delivery zone is determined for shipping the product. The payment and delivery details are handled here.

**Shipment and Order Completion:** Once payment is confirmed, the order is shipped. The user can track the status of the shipment (shipped, delivered, etc.).

**System Architecture**

**System Architecture Overview**

**1. Frontend (Next.js UI):**

* **Role:** User interface for browsing products, adding to cart, and completing purchases using the Next.js framework.
* **Data Flow:** Retrieves product details, prices, and stock levels via API calls from the backend.

**2. Backend (Sanity CMS):**

* **Role:** Stores and manages core data, including products, orders, and customer details using Sanity CMS.
* **Data Flow:** Handles API requests, providing accurate and updated data to the frontend.

**3. Third-Party APIs (Shipping & Payment):**

* **Role:**
* **Shipping API:** Calculates delivery costs and provides shipment details.
* **Payment Gateway API:** Processes and verifies payments.
* **Data Flow:** Handles payment and shipping calculations and sends confirmation to the frontend.

**4. Delivery Zone:**

* **Role:** Calculates shipping costs based on regional zones with specific pricing.
* **Data Flow:** Determines shipping costs dynamically and includes them in the order summary.
* **API OVERVIEW**

**1. Fetch Products API (GET /products):**

* **Purpose:**
* This API will be used to retrieve product details. The Frontend will call this API to display a list of products.
* **Response:**
* After calling this API, you will receive product details like product name, price, stock level, and image.

**2. Create Order API (POST /orders):**

* **Purpose:**
* This API will store customer order details in the Backend. When a user places an order, this API will send the customer and selected product details to the Backend.
* **Payload:**
* The payload will include customer information (name, email) and the details of the products they selected.
* **Response:**
* This API will return the order confirmation details, such as the order status.

**3. Shipment Tracking API (GET /shipment):**

* **Purpose:**
* This API will track the status of the order shipment. The user will call this API to view the current status of their order.
* **Response:**
* The response will provide the current status of the order, such as whether the order has been shipped, delivered, or is still being processed.