1. **System Overview**:

This architecture is designed to streamline backend and use Sanity CMS for content management.

* Frontend (Next.js + Tailwind CSS): Handles user interaction and UI/UX design.
* Sanity CMS (Headless CMS): Stores and manages data like products, categories, and other dynamic content.
* ShipEngine API: Manages shipping and logistics.
* Stripe API: Handles payment processing.

**Architecture Diagram**

USER INTERFACE (Frontend)

|

↓

Next.js + Tailwind CSS

|

↓

----------------------------------

BACKEND (Headless CMS + APIs)

----------------------------------

**Sanity CMS**

|

↓

Third-Party APIs:

- ShipEngine (Shipping)

- Stripe (Payments)

----------------------------------

**DATABASE (Sanity Data Store)**

----------------------------------

Entities:

- Users

- Products

- Orders

- Shipments

**Website Page List**

**Homepage**

1. **Products Listing Page**
2. **About Page**
3. **Contact Page**
4. **Admin Panel**
5. **Sign In Page**
6. **Sign Up Page**
7. **Category-Wise Product Pages**
8. **Add to Cart Page**
9. **Checkout Page**
10. **Payment with Stripe Page**
11. **Shipment with ShipEngine Page**

2. Workflows

A. Product Addition Workflow (Admin-Side)

Admin logs in to Sanity CMS via its dashboard.

Admin creates a new product with details like name, price, category, and images.

Product data is saved in Sanity's backend (which acts as your database).

B. Customer Order Workflow (Frontend-Side)

User browses the marketplace using Next.js frontend.

User adds products to their cart.

User clicks "Checkout" → cart details are sent to the backend through API calls.

Backend:

Retrieves product data from Sanity CMS.

Uses Stripe API to process the payment.

Uses ShipEngine API to generate a shipping label and track the shipment.

If payment and shipping are successful:

Save order details (products, user info, payment ID, and shipping status) in Sanity CMS.

Trigger a confirmation email or SMS (optional).

3. API Requirements

Sanity CMS (Headless CMS)

Get Products API: Fetch products dynamically for the frontend.

Add Product API: Admin adds product details.

Get Orders API: Retrieve orders to display admin-side order history.

Save Orders API: Save order data, including product IDs, user info, and payment status.

Stripe API (Payments)

Create Payment Intent: Generate a payment request.

Capture Payment: Finalize the payment process.

ShipEngine API (Shipping)

Create Shipping Label: Generate labels for the customer’s order.

Track Shipment: Track the order's delivery status.

**Frontend:**

Next.js: Handles UI and server-side rendering.

Tailwind CSS: Makes design look modern and responsive without much effort .

**Backend:**

Sanity CMS: Handles content and acts as a headless CMS for products, orders, and other dynamic content.

**APIs:**

Stripe API: Secure payment processing.

ShipEngine API: Manage shipping and logistics.

**By - Muhammad Shahroz**