HACKATHON 3

Day 2 Assignment: Planning the Technical Foundation for the SHOP.CO General E-commerce store.

System Architecture

Frontend (Next.js)

- User-facing interface for the marketplace.
- Handles product browsing, cart functionality, and order placement.

Sanity CMS

- Centralized content management system for storing product data, user details, orders, and dynamic content (replacing the need for an additional database).

Third-Party APIs

- **Shipment Tracking API**: Fetches real-time shipping updates.
- **Email/SMS API**: Sends transactional notifications to users.

Payment Gateway

- Handles secure payment processing (e.g., Stripe, PayPal).

High-Level Data Flow

- **Frontend (Next.js)**: User interacts with the UI, browsing products, adding to the cart, and placing orders.
- **Sanity CMS**: Handles the dynamic content, including product details, user data, and order history.
- **Third-Party APIs**: Manage external data like shipment tracking and email/SMS notifications.

- **Payment Gateway**: Processes payment and sends confirmation back to the frontend and CMS.

Key Workflows and Interactions

1. **User Registration**
    a. User submits registration details.
    b. Data is stored in Sanity CMS.
    c. Confirmation email is sent via Email API.
2. **Product Browsing**
    a. User selects a category and views products.
    b. Data is dynamically fetched from Sanity CMS.
3. **Order Placement**
    a. User adds products to the cart and checks out.
    b. Order details are sent to Sanity CMS.
    c. Payment is processed via Payment Gateway.
    d. Order confirmation is saved in Sanity CMS.
4. **Shipment Tracking**
    a. The order tracking ID is sent to the Shipment Tracking API for real-time updates.

Example Architecture Diagram

```
[Frontend (Next.js)]
      |
[Sanity CMS] <---------> [Product Data API]
      |
[Third-Party APIs] ----> [Shipment Tracking API]
      |
[Payment Gateway]
```

Key Tools and APIs

Frontend (Next.js)

- **Next.js**: A framework for building server-rendered React applications. It helps in creating dynamic and static web pages with ease, providing features like routing, pre-rendering, and API routes.

- **Tailwind CSS**: A utility-first CSS framework that allows for rapid and responsive design. It helps to style the application by applying classes directly to HTML elements.

Sanity CMS

- **Sanity CMS**: A content management system that stores and manages all dynamic content for your site. It is used here as the **database** for your project. It manages product information, user data, and order history, which can be fetched and updated via APIs.

Third-Party APIs

- **Shipment Tracking API**: These APIs allow you to track the status of a shipment in real-time. Examples include:
  - **Shippo**: Provides shipment tracking, label generation, and shipping rate comparison.
  - **AfterShip**: Offers tracking services for various carriers, sending users automatic tracking updates.
- **Email/SMS API**: These APIs are used for sending notifications, such as order confirmations, shipping updates, and promotional messages:
  - **Twilio**: A cloud communication platform that allows for sending SMS, voice messages, and more.
  - **SendGrid**: A cloud-based email delivery service that helps send transactional and marketing emails, like registration confirmation and order receipts.

Payment Gateway

- **Payment Gateway**: These services process online payments securely by handling transactions between customers, merchants, and financial institutions:
  - **Stripe**: A popular payment processing platform for online businesses. It supports card payments, subscriptions, and more.
  - **PayPal**: A widely used payment service that allows users to make secure online payments via credit/debit cards or their PayPal accounts.
  - **Razorpay**: A payment gateway offering services like payment collection, refunds, and subscription handling for businesses.

1. Fetch All Available Products

- **Endpoint Name**: `/products`
- **Method**: `GET`
- **Description**: Fetch all available products from Sanity CMS.
- **Response Example**:

```
[
  {
    "id": 1,
    "name": "Product A",
    "price": 100,
    "stock": 50,
    "image": "https://example.com/images/product-a.jpg"
  },
  {
    "id": 2,
    "name": "Product B",
    "price": 200,
    "stock": 30,
    "image": "https://example.com/images/product-b.jpg"
  }
]
```

2. Create New Order

- **Endpoint Name**: `/orders`
- **Method**: `POST`
- **Description**: Create a new order in Sanity CMS.
- **Payload**:

```
{
  "customerId": 123,
  "products": [
    { "productId": 1, "quantity": 2 },
    { "productId": 2, "quantity": 1 }
  ],
  "paymentStatus": "Pending",
  "shippingAddress": "123 Main St, City, Country"
```

```
}
```

- **Response Example**:

```
{
  "orderId": 456,
  "status": "Order Created"
}
```

3. Track Order Status

- **Endpoint Name**: `/shipment`
- **Method**: `GET`
- **Description**: Track order status via third-party Shipment Tracking API.
- **Response Example**:

```
{
  "shipmentId": "789",
  "orderId": 456,
  "status": "In Transit",
  "expectedDeliveryDate": "2025-01-20"
}
```

4. User Registration (Optional)

- **Endpoint Name**: `/register`
- **Method**: `POST`
- **Description**: Register a new user and store their details.
- **Payload**:

```
{
  "name": "John Doe",
  "email": "john.doe@example.com",
  "password": "password123"
}
```

- **Response Example**:

```
json
```

```
CopyEdit
{
  "userId": 123,
  "status": "Registration Successful"
}
```

5. User Login (Optional)

- **Endpoint Name**: `/login`
- **Method**: `POST`
- **Description**: Authenticate user login.
- **Payload**:

```
{
  "email": "john.doe@example.com",
  "password": "password123"
}
```

- **Response Example**:

```
{
  "userId": 123,
  "status": "Login Successful",
  "token": "jwt-token"
}
```

6. Fetch Product Details (Optional)

- **Endpoint Name**: `/products/{id}`
- **Method**: `GET`
- **Description**: Fetch detailed information for a specific product.
- **Response Example**:

2. Key Workflows

- **User Registration**:
    - User enters registration details.

    - Frontend sends a POST request to Sanity CMS (or optional database).
    - User details are stored in Sanity CMS.
    - A confirmation email is sent via the Email API.
- **Product Browsing**:
    - User selects a category.
    - Frontend sends a GET request to Sanity CMS for the category-specific products.
    - Products are displayed on the UI.
- **Order Placement**:
    - User adds products to the cart.
    - At checkout, order details are sent to Sanity CMS.
    - Payment is processed via the payment gateway (Stripe/PayPal).
    - Order confirmation is displayed and stored in Sanity CMS.
- **Shipment Tracking**:
    - The order tracking ID is passed to the Shipment Tracking API.
    - Shipment status is fetched and displayed in real time on the frontend.


```
{
  "id": 1,
  "name": "Product A",
  "price": 100,
  "description": "Detailed product description here.",
  "stock": 50,
  "image": "https://example.com/images/product-a.jpg"
}
```


General E-Commerce:

- **Product browsing, cart management, and order placement**: Standard workflows for browsing products, adding to the cart, and placing orders.
- **Example Endpoint**: `/products` to fetch all available products.

4. API Endpoints

Endpoint	Method	Purpose	Response Example

/products	GET	Fetches all product details	{ "id": 1, "name": "Product A", "price": 100 }
/orders	POST	Creates a new order in Sanity CMS	{ "orderId": 456, "status": "Order Created" }
/shipment	GET	Track order status via third-party API	{ "shipmentId": "789", "status": "In Transit" }
/express-delivery-status	GET	Fetch real-time delivery status for perishable items	{ "orderId": 123, "status": "In Transit", "ETA": "15 mins" }