DAY # 2

**Marketplace technical planning**

**\*Author\*: Rafia Samad**

**\*Role\*: Student Of GIAIC Web Developer.**

---

***## Overview***

This document outlines the \****technical foundation****\** and \****enhanced workflow***\* for the **\*General E-commerce Website\*.** It includes system architecture, key workflows, API endpoints, and a technical roadmap.

---

***## System Architecture***

### Component Descriptions

- \**Frontend (Next.js)\*:*

- Provides a responsive and interactive user interface for browsing products, managing orders, and handling user authentication.

- Fetches and displays data from the backend APIs in real-time.

*- \*Sanity CMS\*:*

- Centralized backend for managing product information, user data, order records, and inventory.

- Exposes APIs for dynamic data communication with the frontend.

*- \*Third-Party APIs\*:*

1. \***Shipment Tracking API (ShipEngine)\*:** Fetches real-time shipping updates and generates tracking details.

2. **\*Payment Gateway (Stripe)\*:** Processes secure transactions and confirms payment status.

- \**Authentication (Clerk)\*:*

- Handles user registration, login, and session management.

- Integrates with Sanity CMS to store user data securely.

---

***## Key Workflows***

*### 1. User Registration*

*- \*Process\*:*

- User signs up via the frontend using Clerk.

- Registration details are stored in Sanity CMS.

### 2. Product Browsing

*- \*Process\*:*

- User navigates through product categories on the frontend.

- Sanity CMS API fetches product data (name, price, stock, description, images).

- Dynamic product listings are displayed on the frontend.

### 3. Order Placement

*- \*Process\*:*

- User adds products to the cart and proceeds to checkout.

- Order details (products, quantities, shipping address) are sent to Sanity CMS.

- Payment is processed via Stripe.

- A confirmation message is sent to the user's email, and the order is recorded in Sanity CMS.

### 4. Shipment Tracking

- *\*Process\*:*

- After order placement, shipment details are updated using ShipEngine.

- Real-time tracking information is displayed to the user on the frontend.

### 5. Inventory Management

*- \*Process\*:*

- Product stock levels are managed in Sanity CMS.

- Real-time stock updates are fetched from Sanity CMS.

- Out-of-stock products are added to the wishlist instead of the cart.

- In-stock products can be added to the cart and proceed to checkout

***#Technical Roadmap***

This document outlines the technical roadmap for the \***General E-commerce Website \*.** It covers the development, testing, and launch phases, along with key features and workflows.

*## Development Phase*

***### Authentication***

- Implement user registration and login using **\*Clerk\*.**

- Integrate Clerk with **\*Sanity CMS\*** for user data storage.

***### Product Management***

- Create mock API for product data.

- Store product data in \***Sanity CMS\*.**

- Fetch and display product data on dynamic frontend pages.

***### Cart and Wishlist***

- Implement add-to-cart functionality with real-time stock checks.

- Allow out-of-stock products to be added to the wishlist.

- Display total bill and a "Proceed to Checkout" button on the cart page.

***### Payment Integration***

- Integrate **\*Stripe\*** for secure payments.

- Use Stripe test account for development.

- Handle payment success and failure scenarios.

***### Shipment Tracking***

- Integrate **\*ShipEngine\*** for shipment tracking.

- Generate tracking numbers and display them on the frontend.

- Allow users to track their orders in real-time.

***### Inventory Management***

- Create API for real-time stock updates in **\*Sanity CMS\*.**

- Update stock levels upon order placement.

- Prevent out-of-stock products from being added to the cart.

---

*## Testing Phase*

**### End-to-End Testing**

- Test all workflows, including:

- User registration.

- Product browsing.

- Cart management.

- Checkout process.

- Shipment tracking.

- Validate API responses and ensure data accuracy.

***### Security Audits***

- Conduct security audits for sensitive data handling, including:

- User authentication.

- Payment processing.

---

***## Launch Phase***

**### Deployment**

- Deploy the platform on a cloud hosting service **(e.g., \*Vercel, \*\*Netlify\*).**

- Monitor user feedback and optimize for performance.

**### Post-Launch**

- Collect user feedback for continuous improvement.

- Optimize API performance and frontend loading times.

- Scale infrastructure based on traffic and demand.

---

***## Conclusion***

This technical foundation outlines the architecture, workflows, and API endpoints for the \***General E-commerce Website \*.** The platform will provide a seamless eCommerce experience with:

- Robust authentication.

- Efficient inventory management.

- Real-time shipment tracking.

-