**Author**:

**Laraib Tabbassum**

**General Ecommerce Marketplace: Technical Foundation and Enhanced Workflow**

**System Architecture Overview**

**High-Level Diagram**

**[Frontend (Next.js)]**

**|**

**[Sanity CMS] <------> [Product Data API]**

**|**

**[Third-Party APIs] <------> [Shipment Tracking / Payment Gateway] Component Descriptions**

**Frontend (Next.js):**

For frontend next js frame work for making UI Ux.

**Backend (Sanity):**

For backend I preferred sanity to fetch the data and store in database.

**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 signup, login, and session management.

Integrates with Sanity CMS to store user data securely.

**Libraries:**

**React:** react use for add different icons.

Redux: for add to cart functionality.

Shadcn: for different types of components like carusel, dropdown menu and any other.

1. **Workflows:**
   * **Product Browsing:**
     1. User searches for products -> Frontend sends API request to Sanity CMS.
     2. CMS responds with product data -> Products displayed dynamically.
   * **Order Placement:**
     1. User adds items to the cart -> Sends order details to Sanity CMS.
     2. CMS saves the order -> Payment processed through Payment Gateway.
     3. Confirmation sent to user -> Shipment info fetched via third-party API.

**2.4 Shipment Tracking**

**Process:**

After order placement, shipment details are updated using ShipEngine.**2.5 Inventory Management**

**Process:**

**Category-Specific Instructions**

**3 General E-Commerce**

**Focus on:**

**Workflow Example:**

**Response Example:**

**JSON:**

[

{

"name": "Product Name",

"slug": "product-slug",

"image": "https://productimage.png",

"additionalImages": ["https://productimage2.png", "https://productimage3.png"],

"description": "Product Description",

"price": 150,

"discountPrice": 118,

"inStock": true,

"stock": 78,

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

Product stock levels are managed in Sanity CMS.

Real-time stock updates are fetched from Sanity CMS.

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

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

Standard product browsing, cart management, wishlist, and order placement.

Inventory management and real-time stock updates.

1. **Endpoint:** /products

2. **Method:** GET

3. **Purpose:** Fetch all product listings."rating": 4,

"reviews": 18

}

]

**API Endpoints**

**Endpoint : Method: Purpose: Response Example**

/productsGETFetch all product details

[ { "name": "Product Name", "slug": "product-slug", "price": 100 } ]

/order POST Submit new order details

{ "orderId": 123, "status": "success" }

/shipment-tracing GETFetch real-time tracking updates{ "trackingId": "AB123", "status": "In Transit" }

/delivery-status GETFetch express delivery tracking information{ "orderId": 456, "deliveryTime": "30 mins" }

/inventoryGETFetch real-time stock levels

{ "productId": 789, "stock": 50 }

/cart POST Add product to cart

{ "cartId": 101, "items": [...] }

/wishlist POST Add product to wishlist

{ "wishlistId": 202, "items": [...] }

**Sanity Schema Example**

**Product Schema:**

import { TrolleyIcon } from "@sanity/icons";

import { defineField, defineType } from "sanity";

export const productTypes = defineType({

name: "product",

title: "Products",

type: "document",

icon: TrolleyIcon,

fields: [

defineField({

name: "name",

title: "Product Name",type: "string",

validation: (Rule) => Rule.required(),

}),

defineField({

name: "slug",

title: "Slug",

type: "slug",

options: {

source: "name",

maxLength: 96,

},

validation: (Rule) => Rule.required(),

}),

defineField({

name: "image",

title: "Product Image",

type: "image",

options: {

hotspot: true,

},

validation: (Rule) => Rule.required(),

}),

defineField({

name: "additionalImages",

title: "Additional Images",

type: "array",

of: [{ type: "image", options: { hotspot: true } }],

}),

defineField({

name: "description",title: "Description",

type: "blockContent",

}),

defineField({

name: "price",

title: "Price",

type: "number",

validation: (Rule) => Rule.required().min(0),

}),

defineField({

name: "discountPrice",

title: "Discount Price",

type: "number",

description: "Discounted price of the product (optional)",

validation: (Rule) =>

Rule.custom((discountPrice, context) => {

const doc = context.document;

if (doc && typeof doc.price === "number") {

if (discountPrice && discountPrice >= doc.price) {

return "Discount price must be less than the original price";

}

}

return true;

}),

}),

defineField({

name: "inStock",

title: "In Stock",

type: "boolean",

description: "Indicates whether the product is currently in stock",initialValue: true,

validation: (Rule) => Rule.required(),

}),

defineField({

name: "stock",

title: "Stock Quantity",

type: "number",

description: "Number of items available in stock",

validation: (Rule) => Rule.required().min(0),

}),

defineField({

name: "rating",

title: "Rating",

type: "number",

description: "Rating from 0 to 5",

validation: (Rule) => Rule.min(0).max(5).precision(1),

}),

defineField({

name: "reviews",

title: "Reviews Count",

type: "number",

description: "Number of reviews for the product",

validation: (Rule) => Rule.required().min(0),

}),

],

preview: {

select: {

title: "name",

media: "image",

subtitle: "price",inStock: "inStock",

stock: "stock",

},

prepare({ title, subtitle, media, inStock, stock }) {

return {

title,

subtitle: `$${subtitle} | ${inStock ? `In Stock (${stock})` : "Out of Stock"}`,

media,

};

},

},

});

**Technical Roadmap**

**Development Phase:**

**Authentication:**

**Product Management:**

**Payment Integration:**

Implement user registration and login using Clerk.

Integrate Clerk with Sanity CMS for user data storage.

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 wishlist.

Display total bill and proceed to checkout button on cart page.

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 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 cart.

**Testing Phase:**

**End-to-End Testing:**

Test all workflows, including user registration, product browsing, cart management, checkout, and

shipment tracking.

Validate API responses and ensure data accuracy.

**Security Audits:**

Conduct security audits for sensitive data handling, including user authentication and 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 Bandage Online

Shopping platform. By following the enhanced workflow and technical roadmap, the platform will provide a

seamless eCommerce experience with robust authentication, inventory management, and shipment tracking

features.