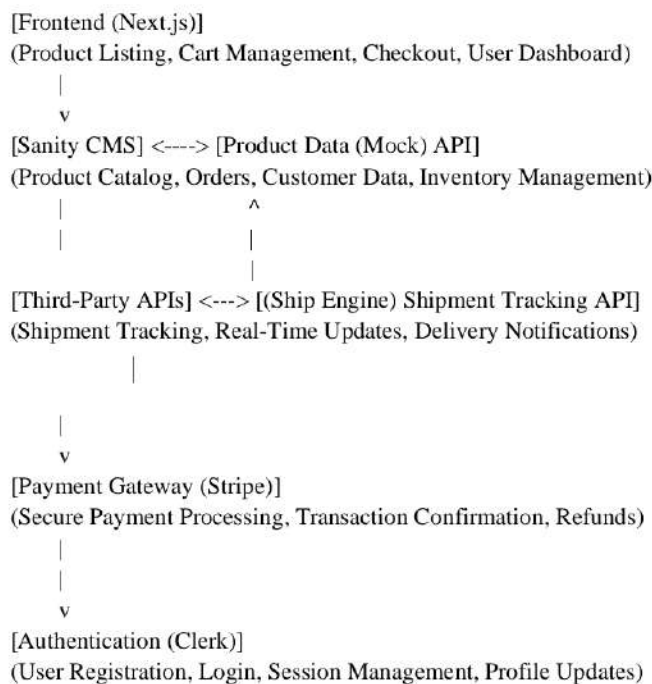


Name: Osama Bin Nadeem
Roll No: (00115125)
Website: Bandage *Online Store*
Email: osamanadeem4345@gmail.com

BANDAGE E-Commerc Platform: Technical Documentation

1. System Architecture Overview



2. Component Descriptions

➤ ***Frontend (Next.js):***

The frontend is like the storefront of the Bandage Online Store. It's built using Next.js, which makes it super-fast, smooth, and easy to use. Whether you're browsing sneakers, checking out the latest collections, or placing an order, everything feels quick and responsive. It's designed to work perfectly on both mobile and desktop, so you can shop anytime, anywhere.

➤ ***Sanity CMS:***

Think of Sanity CMS as the backbone of the store. It's where all the important stuff lives—product details, customer info, orders, and even inventory. It's super flexible and lets us manage everything in real-time. Whenever you see a product description or check your order status, it's Sanity working behind the scenes to make sure everything's up-to-date.

➤ **Third-Party APIs:**

○ **Shipment Tracking API (Ship Engine):**

Ever wondered where your order is? This API keeps track of your package in real-time. It gives you updates like “Out for Delivery” or “Arriving Tomorrow,” so you’re never left guessing.

○ **Payment Gateway (Stripe):**

When it’s time to pay, Stripe steps in to make sure your transaction is safe and secure. It handles everything from card payments to confirming your order, so you can shop with confidence.

➤ **Authentication (Clerk):**

Clerk is like the friendly bouncer at the door. It handles all the login and sign-up stuff, making sure only you can access your account. It also remembers your preferences, so every time you come back, it feels like the store knows you.

3. Key Workflows

i. User Registration

Process: Users sign up by entering their details (name, email, and password).

i. Behind the Scenes:

1. Registration details are securely stored in Sanity CMS.
2. A confirmation email is sent to verify the account.

ii. Outcome:

Users can log in, track orders, and save preferences for a personalized shopping experience.

ii. Product Browsing:

Process: Users browse through product categories, filter by size, color, or price, and view detailed product pages.

i. Behind the Scenes:

The frontend fetches product data (name, price, description, images) from Sanity CMS in real-time.

ii. Outcome:

Users can explore products effortlessly, just like in a physical store.

iii. Order Placement:

Process: Users add products to their cart, enter shipping details, and proceed to checkout.

i. Behind the Scenes:

1. Cart details are sent to Sanity CMS.
2. Stripe processes the payment securely.
3. An order confirmation email is sent, and the order is recorded in Sanity.

ii. Outcome:

A seamless and secure checkout experience for users.

iv. Shipment Tracking:

Process: After placing an order, users can track their shipment in real-time.

i. Behind the Scenes:

The *Ship Engine API* fetches shipment updates (e.g., “Out for Delivery” or “Arriving Tomorrow”).

ii. Outcome:

Users stay informed about their delivery status, reducing uncertainty.

v. Inventory Management:

Process: Product stock levels are updated in real-time.

i. Behind the Scenes:

1. Sanity CMS tracks inventory and updates product availability.
2. Out-of-stock products are added to the Wishlist instead of the cart.

ii. Outcome:

Users avoid ordering unavailable products and receive notifications when items are back in stock.

4. Category-Specific Instructions

- **General E-Commerce (Bandage):**

- **Focus on:** Standard product browsing, cart management, wishlist, and order placement.
- Inventory management and real-time stock updates.

- **Workflow Example:**

- **Endpoint:** /products
- **Method:** GET
- **Purpose:** Fetch all Nike product listings.

Response Example:

```
[
  {
    "name": "Nike Air Max",
    "slug": "nike-air-max",
    "image": "https://nikeairmax.png",
    "additionalImages": ["https://nikeairmax2.png", "https://nikeairmax3.png"],
    "description": "Comfortable and stylish sneakers.",
    "price": 150,
    "discountPrice": 118,
    "inStock": true,
    "stock": 78,
    "reviews": 18
  }
]
```

5. API Endpoints

Endpoint	Method	Purpose	Response Example
<i>/products</i>	GET	Fetch all Nike product details	[{"name": "Nike Air Max", "slug": "nike-air-max", "price": 150}]
<i>/order</i>	POST	Submit new order details	{"orderid": 123, "status": "success"}
<i>/shipment-tracking</i>	GET	Fetch real-time tracking updates	{"trackingid": "AB123", "status": "In Transit"}
<i>/shipment-tracking</i>	GET	Fetch express delivery tracking info	{"orderid": 456, "deliveryTime": "30 mins"}
<i>/inventory</i>	POST	Fetch real-time stock levels	{"productid": 789, "stock": 50}
<i>/cart</i>	POST	Add product to cart	{"wishlistid": 202, "items": [...]}

6. Sanity 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",
      stock: "stock",
    },
    prepare({ title, subtitle, media, inStock, stock }) {
      return {

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

```

7. Technical Roadmap

1. Development Phase:

1.1. Authentication:

This phase focuses on creating a secure and user-friendly system for signing up and logging in. It ensures that users can create accounts, verify their email, and recover passwords easily, while their data is stored securely.

1.2. Product Management:

This phase involves organizing and managing all product information, such as names, prices, descriptions, and images. It ensures that users can browse and view detailed product listings in real-time.

1.3.Cart and Wishlist:

This phase enables users to add products to their cart or save them to a Wishlist. It includes features like real-time stock checks, displaying the total bill, and allowing users to proceed to checkout seamlessly.

1.4.Payment Integration:

This phase ensures secure and hassle-free payment processing. It integrates a payment gateway (like Stripe) to handle transactions, confirm payments, and send order confirmation emails to users.

1.5.Shipment Tracking:

This phase provides real-time updates on the status of user orders. It integrates a shipment tracking API (like Ship Engine) to display delivery updates, such as “Out for Delivery” or “Arriving Tomorrow.”

1.6.Inventory Management:

This phase ensures that product stock levels are updated in real-time. It prevents users from ordering out-of-stock products and allows them to add such items to their Wishlist for future notifications.

2. Testing Phase:

2.1. End-to-End Testing:

- **Goal:** *Make sure everything works perfectly for users.*
- **Tasks:**
 - *Test if users can sign up, log in, and reset passwords.*
 - *Check if users can browse products and see details.*
 - *Verify users can add/remove items from the cart.*
 - *Ensure users can complete checkout and pay securely.*
 - *Test if users can track their orders in real-time.*
 - *Check if stock updates work correctly.*
 - *Make sure the platform works well on all devices.*

3. Launch Phase:

3.1. Deployment:

- **Goal:** *Make the platform live for everyone to use.*
- **Tasks:**
 - *Deploy the frontend on Vercel or Netlify.*
 - *Set up the backend (Sanity CMS) and connect all APIs.*
 - *Add tools to monitor performance and fix issues quickly.*

3.2. Post-Launch:

- **Goal:** *Keep improving the platform based on user feedback.*
- **Tasks:**
 - *Collect feedback from users through surveys and analytics.*
 - *Optimize the platform for better speed and performance.*
 - *Scale the system to handle more users as the store grows.*

Conclusion:

This roadmap outlines the steps we're taking to build a seamless and user-friendly Bandage Online Store. From creating a secure login system to ensuring smooth payments and real-time order tracking, every phase is designed to make your shopping experience enjoyable and hassle-free. Once the platform goes live, we'll continue to improve it based on your feedback, ensuring it stays fast, reliable, and easy to use. Thank you for being part of this journey—we're excited to bring you the best online shopping experience!

x-----x-----x