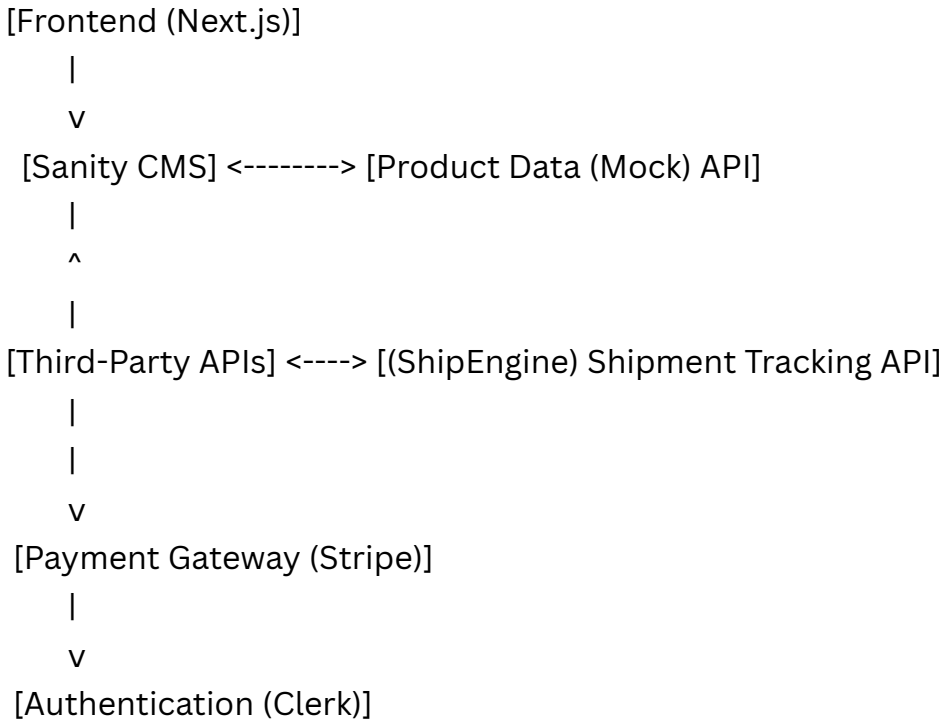
Building a Seamless E-Commerce Experience

The Technical Blueprint of Shop.co's Enhanced Online Shopping Platform

Overview

This document outlines the technical foundation and enhanced workflow for the Shop.co Online Shopping Platform. It includes system architecture, key workflows, API endpoints, and a technical roadmap designed specifically for a seamless e-commerce experience.

High-Level Diagram



Component Descriptions

Frontend (Next.js)

- **User Interface:** Creates a user-friendly interface for browsing products, managing orders, and logging in.
- **Dynamic Display:** Displays data dynamically by connecting to backend APIs.

Sanity CMS

- **Data Management:** Manages product details, user information, orders, and inventory in one place.
- **Data Sharing:** Shares data with the frontend through APIs.

Third-Party APIs

- **Shipment Tracking API (ShipEngine):** Provides live shipping updates and tracking information.
- **Payment Gateway (Stripe):** Processes payments securely and confirms transactions.

Authentication (Clerk)

- **User Management:** Handles user sign-up, login, and session management.
- **Security:** Works with Sanity CMS to securely store user details.

Key Workflows

1. User Registration

- **Sign-Up Process:** Users sign up through the frontend using Clerk.
- **Data Security:** Registration details are securely stored in Sanity CMS.

2. Product Browsing

- **Exploration:** Users explore product categories on the frontend.
- **Data Retrieval:** Sanity CMS API provides product details like name, price, stock, description, and images.
- **Dynamic Display:** The frontend displays dynamic product listings.

3. Order Placement

- **Cart Management:** Users add products to their cart and proceed to checkout.
- **Order Submission:** Order details (products, quantities, shipping address) are sent to Sanity CMS.
- **Secure Payment:** Payments are securely processed through Stripe.

4. Shipment Tracking

- **Real-Time Updates:** Shipment details are updated using ShipEngine after the order is placed.
- **User Notification:** Real-time tracking information is shown to the user on the frontend.

5. Inventory Management

- **Stock Control:** Stock levels are managed in Sanity CMS.
- **Availability Check:** Real-time updates ensure only in-stock products can be added to the cart.

API Documentation

Endpoint	Method	Purpose	Response Example
/products	GET	Fetch all product details	[{ "name": "Product Name", "slug": "product-slug", "price": 100 }]
/order	POST	Submit new order details	{ "orderId": 123, "status": "success" }
/shipment-tracking	GET	Fetch real-time tracking updates	{ "trackingId": "AB123", "status": "In Transit" }
/delivery-status	GET	Fetch express delivery information	{ "orderId": 456, "deliveryTime": "30 mins" }
/inventory	GET	Fetch real-time stock levels	{ "productId": 789, "stock": 50 }
/cart	POST	Add product to cart	{ "cartId": 101, "items": [...] }

Development Phase

1. Authentication

- **User Registration & Login:** Implement user registration and login using **Clerk**.
- **Clerk Integration with Sanity CMS:** Link **Clerk** for user data storage in **Sanity CMS**.

2. Product Management

- **Mock API:** Create a mock API to manage product data.
- **Store Products in Sanity CMS:** Save product details such as name, price, description, etc., in **Sanity CMS**.

- **Display Products:** Fetch and display product data dynamically on frontend pages.

3. Cart and Wishlist

- **Add-to-Cart Functionality:**
  - Implement functionality to add products to the cart.
  - Use **Redux** to manage cart state for handling multiple product additions and removing items from the cart.
- **Stock Check:** Ensure that users can only add products to the cart if they are in stock.
- **Out-of-Stock to Wishlist:** Allow users to add out-of-stock products to their wishlist.
- **Cart Summary:** Display the total bill and provide a **"Proceed to Checkout"** button on the cart page.

4. Payment Integration

- **Stripe Integration:** Integrate **Stripe** for secure payment processing.
- **Test Account:** Use the **Stripe test account** during development to simulate payments.
- **Payment Handling:** Manage payment success and failure scenarios, and show relevant feedback to users.

5. Shipment Tracking

- **ShipEngine Integration:** Connect **ShipEngine** API to track shipments in real-time.
- **Tracking Numbers:** Generate and display tracking numbers for orders.
- **Order Tracking:** Allow users to track their shipments in real-time.

6. Inventory Management

- **Real-Time Stock API:** Create an API that fetches real-time stock data from **Sanity CMS**.
- **Stock Updates:** Automatically update product stock levels when an order is placed.
- **Out-of-Stock Prevention:** Prevent users from adding out-of-stock items to their cart.

Conclusion

This technical foundation outlines the architecture, workflows, and API endpoints for the **Shop.co** eCommerce platform. The platform is designed to offer a seamless online shopping experience with:

- **Robust Authentication:** Secure user registration and login using Clerk.
- **Efficient Inventory Management:** Real-time stock updates and product management through Sanity CMS.
- **Real-Time Shipment Tracking:** Integration with ShipEngine for tracking orders and providing users with up-to-date shipment information.
- **Smooth Shopping Experience:** Features like adding to the cart, wishlist, secure payment processing via Stripe, and easy checkout.

The goal of **Shop.co** is to provide users with a hassle-free shopping journey while maintaining a highly scalable and performant eCommerce solution.