E-Commerce Platform Documentation

Introduction

This document outlines the architecture of a modern e-commerce platform. Built with **Next.js** and **TypeScript**, the platform integrates **Sanity CMS** for content management, alongside robust APIs for payment and shipment tracking. Below is a detailed explanation of the system's structure and workflows.

System Overview

Frontend Design

- **Framework**: Next.js 14 with TypeScript for server-side rendering (SSR) and enhanced performance.
- **Primary Pages**:
    - **Homepage**: Displays featured collections and trending items.
    - **About Us**: Shares details about the brand and its story.
    - **Products**: Lists all available furniture pieces.
    - **Product Details**: Dynamically renders individual product details via route parameters.
    - **Cart**: A simplified interface for viewing and editing cart items.
    - **User Portal**:
        - **Login/Sign Up**: Manages user authentication.
        - **Dashboard**: Displays user-specific data, such as order history and shipping updates.

Core Components

- **Reusable Elements**:
    - **ProductTile.tsx**: A versatile card component to showcase furniture items.
    - **HeroBanner.tsx**: Highlights featured promotions or new collections.
    - **CategoryList.tsx**: Lists product categories for easy navigation.
- **Feature Components**:
    - **TrendingProducts.tsx**: Displays popular items dynamically.
    - **Testimonials.tsx**: Showcases customer reviews.

Content Management System (CMS)

- **Sanity Features**:
  - **Products**: Stores data such as pricing, descriptions, and stock levels.
  - **Orders**: Tracks customer purchases and fulfillment status.
  - **Users**: Manages user profiles and credentials.
  - **Inventory**: Maintains real-time stock updates.
  - **Analytics**: Tracks sales trends and product performance.
- **Data Handling**:
  - **Custom Schemas**: Define structures for products, orders, users, and inventory.
  - **Real-Time Queries**: Fetch up-to-date content using GROQ.

Backend API Integrations

Mock APIs

- Simulate backend operations for seamless testing:
  - `/api/products`: Fetch product details.
  - `/api/cart`: Handle cart-related operations.
  - `/api/orders`: Manage order processing.

Payment System

- **Stripe Integration**:
  - Facilitates secure transactions using **Stripe Elements**.
  - Mock payment functionality during development.

Shipment Tracking

- **Ship Engine API**:
  - **Endpoints**:
    - Create shipments and generate tracking numbers.
    - Fetch real-time tracking details for user convenience.

User Workflows

Customer Journey

1. **Browse Products**:
   - Items are dynamically rendered from Sanity's dataset.
2. **Add to Cart**:
   - Users can save selected products to a cart stored locally or linked to their account after logging in.
3. **Place Order**:

    - Users provide shipping and payment details via a streamlined checkout flow.
4. **Track Shipment**:
    - Shipment details and status updates are displayed in the user dashboard.

Technologies Used

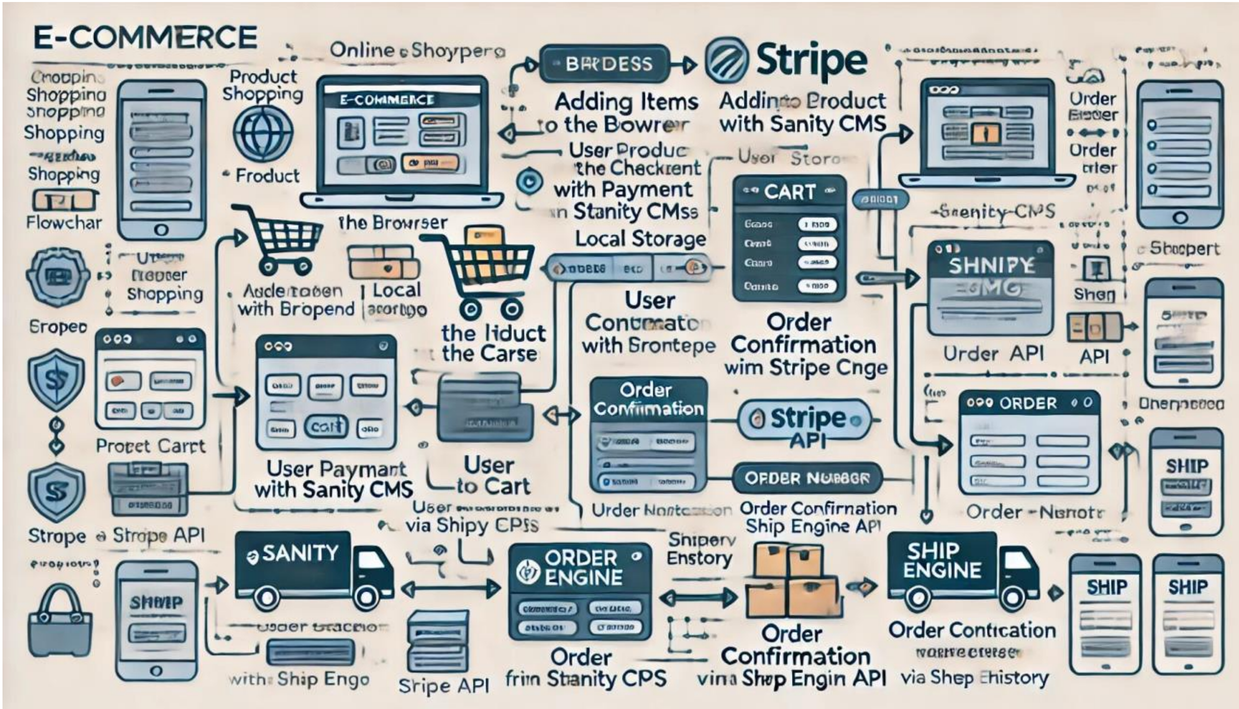
- **Frontend**: Next.js 14 with TypeScript for modular and scalable development.
- **CMS**: Sanity for real-time content management.
- **Payment Gateway**: Stripe for secure online transactions.
- **Shipping Integration**: Ship Engine API for efficient order tracking.
- **Hosting**: Vercel for seamless deployment and performance optimization.

Data Flow

1. **Products**: Sourced from Sanity, ensuring real-time updates on the frontend.
2. **Cart Data**: Temporarily stored in local storage, synced to the user account after login.
3. **Orders**: Persisted in Sanity, accessible to both users and admins.
4. **Payments**: Managed securely via Stripe API.
5. **Shipment Tracking**: Real-time updates fetched from the Ship Engine API.

Conclusion

This architecture offers a robust foundation for an e-commerce platform. By leveraging the flexibility of Next.js, the scalability of Sanity, and integrations with Stripe and Ship Engine, the platform is well-equipped to deliver a smooth and modern shopping experience.


E-COMMERCE
Online Shoypero
Stripe
Adding Items
to the Bowrer
Addinto Product
with Sanity CMS
Product
Shopping
E-COMMERCE
Product
The Browser
User Produc
the Checkrent
with Payment
Local Storage
CART
Flowchar
Local
User
Order
Confirmation
Order API
API
Proect Carrt
User Paymant
with Sanity CMS
User
to Cart
Order
Confirmation
Stripe
API
ORDER NUMBER
ORDER
SHIP
Order Confirnation
Ship Engine API
Strope
Strope API
SANITY
ORDER
ENGINE
SHIP
ENGINE
SHIP
SHIP
with Ship Engo
Sripe API
Order
frin Stanity CPS
Order
Confirmation
vina Shep Engin API
Order Contication
via Shep Ehistory