Technical Roadmap:

Overview

This documentation provides a comprehensive guide to transforming your static e-commerce website into a fully functional dynamic platform using Sanity CMS as the backend, integrating third-party APIs, and ensuring user-friendly workflows. The components covered include Home, About, Contact, Shop, Cart, Checkout, Blog, My Account, and Single Product.

Setup and Static Website Refinement

Goal:

Enhance the static website's structure and prepare it for dynamic content integration.

Sanity CMS Integration

Goal:

Set up Sanity CMS and configure schemas to manage product data, customer details, and orders.

Steps:

Define Schemas:

Create the following schemas:

- Product Schema
- Order Schema
- User Schema

Populate Initial Data:

Add dummy data for products in the Sanity Studio.

Expose APIs:

Use GROQ queries or Sanity's APIs to fetch data.

Test endpoints using tools like Postman.

Frontened and Backened Integration

Goal:

Connect the website's frontend with Sanity CMS to enable dynamic content.

Steps:

Connect Sanity to the Existing Frontend Project:

Install the necessary Sanity client library in your Next.js project.

Configure the Sanity client by providing the project ID and dataset in a new `sanity.js` file.

Fetch and Display Products:

Fetch product data from Sanity in the Shop component.

Map and display the products dynamically in the Shop component.

Implement Single Product View:

Fetch individual product data using dynamic routes.

Display product details dynamically in the Single Product component.

Cart Functionality:

Use React Context or Redux to manage the cart state.

Update the Cart component to list items dynamically and calculate totals.

Checkout Functionality:

Send order data to Sanity for processing and storage.

Third-Party API Integration

Goal:

Integrate third-party APIs for payment processing and shipment tracking.

Steps:

Payment Gateway Integration:

Select a gateway (e.g., Stripe, PayPal).

Implement the payment workflow:

Frontend collects payment details.

Backend processes the payment via the gateway.

Payment confirmation is displayed to the user.

Shipment Tracking API Integration:

Choose a tracking API (e.g., EasyPost, Shippo).

Fetch shipment statuses and display them in the "My Account" or dedicated tracking page.

Testing:

Test payment transactions with sandbox/test accounts.

Verify shipment tracking updates display correctly.