Marketplace Technical Foundation - [General Ecommerce]

OVERVIEW:

SITSTYLE is a cutting-edge, responsive e-commerce platform designed specifically for the furniture marketplace. Whether you're furnishing a home, office, or restaurant, SITSTYLE offers a diverse range of premium furniture designed to meet all your needs.

Our platform connects you to high-quality furniture products, simplifying the online shopping experience with:

- **Convenience:** Browse, choose, and purchase furniture from the comfort of your home or office.

- **Quick Delivery:** Enjoy prompt and reliable delivery services.

- **Easy Customization:** Personalize your furniture with color, size, and other customization options.

- **Clear Pricing:** Transparent pricing with no hidden fees.

Built using Next.js for the frontend and Sanity CMS for backend management, SITSTYLE ensures that your shopping journey is efficient and smooth. It offers:

- **Real-Time Inventory Updates:** Always stay up-to-date with stock availability.

- **Order Tracking:** Know exactly when your items will arrive.

- **Dynamic Product Details:** Get detailed descriptions, images, and specifications for each product.
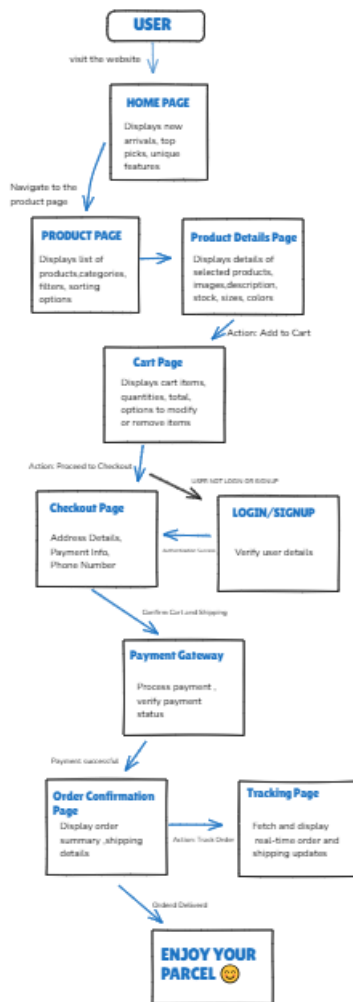
With fast load times and a mobile-first design, SITSTYLE provides a seamless user experience, ensuring that shopping for furniture has never been easier or more enjoyable.

Technologies Used

- **Next.js:** Fast, SEO-friendly frontend framework with dynamic routing and server-side rendering. Manages product pages, cart, checkout, and order confirmations.

- **Sanity CMS:** Backend for managing products, customer data, and orders. Supports real-time updates and customizable schemas for scalability.

- **Tailwind CSS:** Utility-first CSS for responsive, mobile-friendly design.

- **Shipment Tracking API:** Provides real-time order tracking and delivery updates.

- **Payment Gateway API:** Ensures secure payments and supports multiple payment methods like cards and digital wallets.
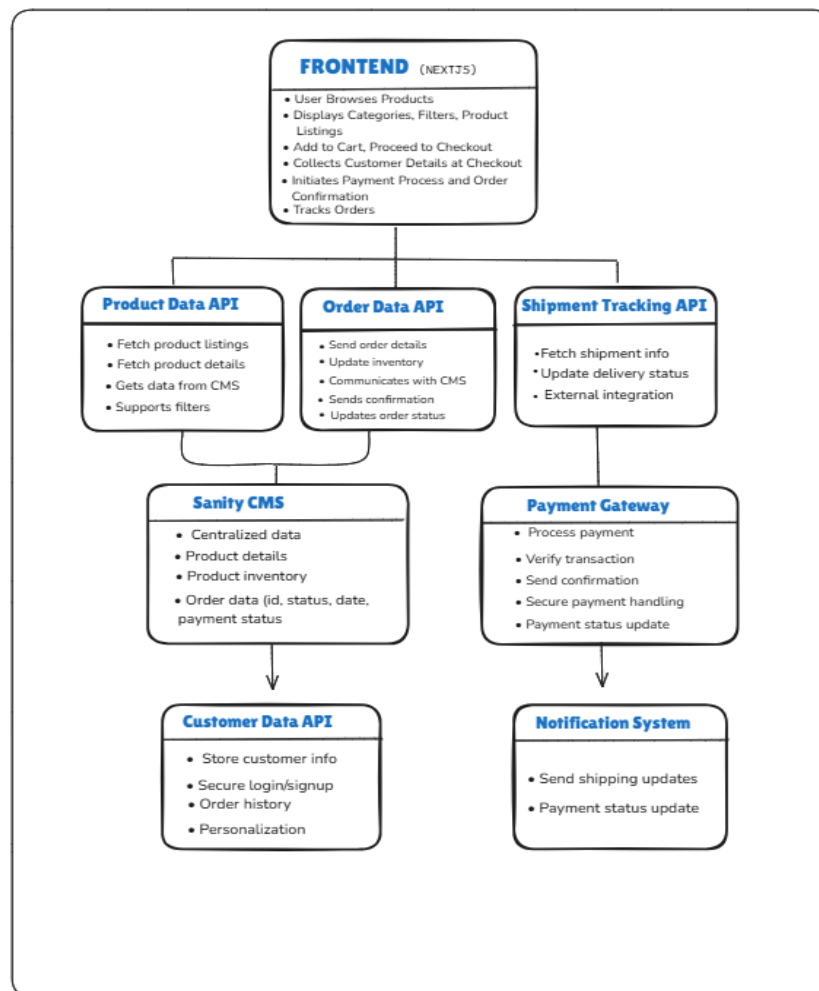
Component Architecture

Components Workflow

- **Home Page:** Users view new arrivals, top picks, and product categories.

- **Product Listing Page:** Users filter and sort products based on preferences.

- **Product Details Page:** Users view detailed information, images, and can add products to the cart.

- **Cart Page:** Users modify quantities, view the total, and proceed to checkout.

- **Checkout Page:** Users enter shipping details, select payment options, and confirm the order.

- **Authentication Page:** Users sign in or create an account to complete their purchase.

- **Payment Gateway:** Users enter payment details, and the payment is processed.

- **Order Confirmation:** Users receive a summary of their order, shipping details, and estimated delivery.

- **Tracking Page:** Users track their order's shipping status and real-time updates.

System Architecture

FRONTEND (NEXTJS)

- User Browses Products
- Displays Categories, Filters, Product Listings
- Add to Cart, Proceed to Checkout
- Collects Customer Details at Checkout
- Initiates Payment Process and Order Confirmation
- Tracks Orders

Product Data API

- Fetch product listings
- Fetch product details
- Gets data from CMS
- Supports filters

Order Data API

- Send order details
- Update inventory
- Communicates with CMS
- Sends confirmation
- Updates order status

Shipment Tracking API

- Fetch shipment info
- Update delivery status
- External integration

Sanity CMS

- Centralized data
- Product details
- Product inventory
- Order data (id, status, date, payment status)

Payment Gateway

- Process payment
- Verify transaction
- Send confirmation
- Secure payment handling
- Payment status update

Customer Data API

- Store customer info
- Secure login/signup
- Order history
- Personalization

Notification System

- Send shipping updates
- Payment status update

Workflow Explanation

Frontend (Next.js): Users interact with the website to browse products, view categories, add items to their cart, and proceed through checkout. The frontend sends API requests for product details, orders, and payments.

Product Data API: Retrieves product listings and details from the CMS, supporting filters like category and price for a seamless user experience.

Order Data API: Handles order creation, updates inventory, communicates with the CMS, sends order confirmations, and manages order status.

Sanity CMS: Acts as the centralized database for product details, inventory, and order data.

Third-Party Payment Gateway: Processes payments, verifies transactions, and updates payment status securely.

Third-Party Shipment Tracking API: Fetches shipping details, updates delivery status, and integrates external tracking for customer convenience.

Customer Data API: Manages customer information, secure authentication, order history, and personalized data.

Notification System: Sends confirmations, shipping updates, and payment status notifications to customers.

API Endpoint

Endpoint Name	Method	Description	Response Example	Status Code
/products	GET	Fetch all available products	[{ "id": 1, "name": "Sofa", "price": 100, "price": 349.99 , "image": "image_url" }]	200 (OK)
/products/[id]	GET	Fetch a single product by ID	{ "id": 1,"name": "Modern Gray Sofa", "price": 599.99, "stock": 20, "category": "Living Room Furniture","tags": ["sofa", "modern", "gray", "comfortable"], "rating": 4.5, "image": "image_url" }	200 (OK)
/orders	POST	Create a new order	{ "orderId": "789", "status": "Success" }	201 (Created)
/orders/[id]	GET	Get details of an order by ID	{"orderId": "ORD123", "customerInfo": { "customerId": "123" }, "productDetails": [{ "productId": "PROD001" }, { "productId": "PROD002" }], "status": "Shipped", "timestamp": "2025-01-16T10:00:00Z}	200 (OK)
/customers	POST	Create a new customer	{"customerId": "124","name": "Jane Smith", "email": "jane@example.com", "phone": "+987654321", "address": "456 Elm St, Springfield", "zipCode": "67890", "orderHistory": ["ORD789"]}	201 (Created)
/customers/[id]	GET	Get customer details by ID	{"customerId": "123", "name": "John Doe", "email": "john@example.com", "phone": "+123456789", "address": "123 Main St, Springfield", "zipCode": "12345", "orderHistory": ["ORD123", "ORD456"]},	200 (OK)
/shipment	GET	Get shipment status	{ "status": "In Transit", "ETA": "1 day", "trackingNumber": "12345" }	200 (OK)
/shipment/[id]	GET	Get detailed shipment information by shipment ID	{ "shipmentId": "12345", "orderId": "789", "status": "Delivered", "deliveryDate": "2025-01-17", "trackingNumber": "12345" }	200 (OK)