# E-Commerce Website Project Proposal

Business Focus

Business Goals:

- Problem: Deliver a reliable, user-friendly marketplace for a wide range of products.
- Target Audience: Consumers seeking convenience and small/medium businesses aiming to sell products online.
- Unique Value Proposition: Competitive pricing, extensive product selection, seamless user experience, and fast delivery.

Data Schema:

- Entities: Products, Orders, Customers, Delivery Zones.
- Relationships:
  - Products linked to Orders via Product ID.
  - Orders linked to Customers via Customer ID.
  - Delivery Zones linked to Orders based on geographic area.

Key Features:

- Personalized recommendations, easy returns, loyalty programs, and advanced search filters.

---

E-Commerce Website Overview

Marketplace Type: General E-Commerce

Primary Purpose: Comprehensive marketplace offering a broad range of products, from everyday essentials to specialized items.

Business Goals:

- Problem to Solve: Simplify the shopping experience with quality products at affordable prices.
- Target Audience: Consumers and small/medium businesses.
- Products/Services: Electronics, clothing, groceries, household items, fast delivery, easy returns, and customer support.
- Unique Selling Points: Affordable prices, diverse product offerings, user-friendly interface, and reliable delivery services.

Data Schema:

- Entities: Products, Customers, Orders, Delivery Zones.
- Relationships: Products <-> Orders <-> Customers, Delivery Zones <-> Orders.

---

Technical Requirements

Frontend:

- UI Design: Clean, intuitive interface with navigation for browsing products.
- Responsive Design: Mobile-first design using Tailwind CSS for a seamless experience across all devices.

Core Pages:

- Home: Featured products, categories, and promotions.
- Product Listings: Grid-based display with filtering and sorting options.

- Product Details: Product information including images, descriptions, prices, and stock availability.
- Cart: Modify quantities, view total price.
- Checkout: Simple, secure checkout with payment and shipping options.
- Order Confirmation: Confirmation with estimated delivery time.

---

Backend: Sanity CMS

- Product Data: Store product details (name, price, description, images, stock).
- Customer Data: Store customer info (name, contact, address).
- Order Management: Track orders with details on products, quantities, customer info, and payment status.

Third-Party APIs:

- Shipment Tracking API: Use Shippo or ShipEngine for real-time order tracking.
- Payment Gateway API: Stripe or PayPal for secure payments.
- Other Services: Email (e.g., SendGrid) for order confirmations and promotional emails.

---

System Architecture

Components:

- Frontend (Next.js): Manages product browsing, order placement, and shipment tracking.
- Sanity CMS: Backend for managing products, customers, and orders.
- Third-Party APIs: Shipment tracking and payment gateways.
- Payment Gateway: Securely processes payments.

Workflow:

1. Frontend requests product data from Sanity CMS.

2. Sanity CMS returns product details, which are displayed on the frontend.

3. Frontend queries Shipment Tracking API for order status.

4. Payment Gateway securely processes payment and updates order status.

---

Key User Workflows:

1. Product Browsing:

   - Action: Browse categories, select products.

   - Data Flow: Frontend requests product data from Sanity CMS.

   - Display: Product details (name, price, stock).

2. Order Placement:

   - Action: Add items to cart and checkout.

   - Data Flow: Frontend sends order details to Sanity CMS.

   - Result: Order confirmation and processing in Sanity.

3. Shipment Tracking:

   - Action: Track order status.

   - Data Flow: Frontend queries Shipment Tracking API.

   - Display: Shipment status (e.g., "In Transit").

---

API Requirements

1. /products (GET): Fetch product details for browsing.

- Example Response:

{

"id": 1, "name": "Product A", "price": 100, "stock": 50, "image": "https://image-link.com/productA.jpg"

}

2. /orders (POST): Create an order with customer and product details.

- Payload Example:

{

"customerInfo": { "customerId": 123, "name": "John Doe", "email": "john@example.com", "address": "1234 Elm St" },

"products": [{ "productId": 1, "quantity": 2, "price": 100 }],

"paymentStatus": "Paid"

}

3. /shipment (GET): Track order status.

- Response Example:

{

"shipmentId": "ABC123", "orderId": 1234, "status": "In Transit", "expectedDeliveryDate": "2025-01-17T14:00:00"

}

---

Sanity Schema Example:

```
export default {
  name: 'product',
  type: 'document',
  fields: [
    { name: 'name', type: 'string', title: 'Product Name' },
    { name: 'price', type: 'number', title: 'Price' },
    { name: 'stock', type: 'number', title: 'Stock Level' },
    { name: 'image', type: 'image', title: 'Product Image' }
  ]
};
```

---

Technical Roadmap

1. Phase 1: Setup
   - Set up Sanity CMS and integrate with Next.js.
   - Design schemas for products and orders.
   - Implement basic frontend pages.

2. Phase 2: API Integration
   - Integrate shipment tracking and payment gateway APIs.
   - Implement product, order, and shipment API endpoints.

3. Phase 3: Testing & Deployment

- Test workflows and user experience.
- Deploy and conduct performance testing.

4. Phase 4: Final Review
   - Perform bug fixes and optimizations.
   - Prepare final documentation.