Marketplace Technical Foundation: General E-Commerce Hekto Furniture website

Table of Contents

Day 1 overview:

Marketplace Type:

I chose General E-Commerce to offer a wide range of furniture like armchairs, desk chairs, park benches, and sofas. This approach allows flexibility, scalability, and access to a larger audience.

Business Goals:

Problem: Make stylish, affordable, and customizable furniture easily available for homes and offices.

Target Audience: Homeowners, office managers, interior designers, and families.

What Makes hekto Special: Customization options, fast delivery, and competitive pricing.

Data Structure:

I designed a simple data schema to organize key elements:

Products: ID, name, price, stock.

Orders: ID, customer info, items, total, status.

Customers: ID, name, contact info.

Delivery Zones: Name, areas covered, assigned drivers.

Introduction:

- This documentation provides a technical overview of a general e-commerce website designed to facilitate online buying and selling.
- The platform includes:
- features for browsing products
    - managing a shopping cart
    - placing orders
    - handling payments.

Features:

Customer Features:

- User registration and login
- Product browsing and searching

- Product filtering and sorting
- Shopping cart and wishlist
- Secure checkout process
- Order tracking and history
- Payment integration

- **Admin Features:**
  - Product management (add, update, delete)
  - Order management
  - User management
  - Analytics dashboard

System Architecture:

Overview:

This document outlines the technical foundation for the Marketplace project, including:

- Routes
- API endpoints
- and order processing flow.

Technology Stack:

- **Frontend:** Next.js
- **Backend:** Sanity CMS
- **Third-Party APIs:**
  - Product listing: /products
  - Product details: /products/{product_id}

API Endpoints:

Sanity API Endpoints:

- **Customer Schema (/customer)**:
  - Create (POST)
  - Get (GET)
  - Update (PUT)
  - Delete (DELETE)
- **Order Schema (/order)**:
  - Create (POST)
  - Get (GET)
  - Update (PUT)
  - Delete (DELETE)
- **Cart Schema (/cart)**:
  - Create (POST)
  - Get (GET)
  - Update (PUT)
  - Delete (DELETE)

Workflow:

- **Homepage (/)**

- Fetch and display product listings from a third-party API.
- Users can navigate to individual product pages.

➢ Product Page (/products/{product_id})

- Fetch and display detailed product information.
- Option to add the product to the cart.

➢ Cart Page (/cart):

- Display the user's shopping cart.
- Allows adding, editing, and removing items.
- Stores data in the **Cart Schema** (Sanity CMS).

➢ Checkout Page (/checkout)

- Allows users to enter customer details and review their order.
- Displays the order total and a confirm button.
- **Backend Actions:**
    - Create a new customer record in Sanity.
    - Create a new order record in Sanity.
    - Assign **Shipping ID** upon successful checkout.

➢ Order Processing

1. **Processing:** Order received.
2. **Shipped:** Order dispatched, assign **Tracking ID**.
3. **Delivered:** Order successfully delivered.

➢ Order Tracking (/order/{order_id})

- Users can track their order using the **Tracking ID**.
- Fetches order details and current status from the **Order Schema** in Sanity.

Data Schemas:

➢ Product Schema (Sanity):

- **product_id**: Unique identifier
- **name:** Product name
- **image**: Product image URL
- **price**: Product price
- **description:** Product description
- **stock:** Available stock count

➢ Customer Schema (Sanity):

- **customer_id**: Unique identifier
- **name**: Customer's name
- **email**: Email address
- **address**: Shipping address
- **phone**: Contact number

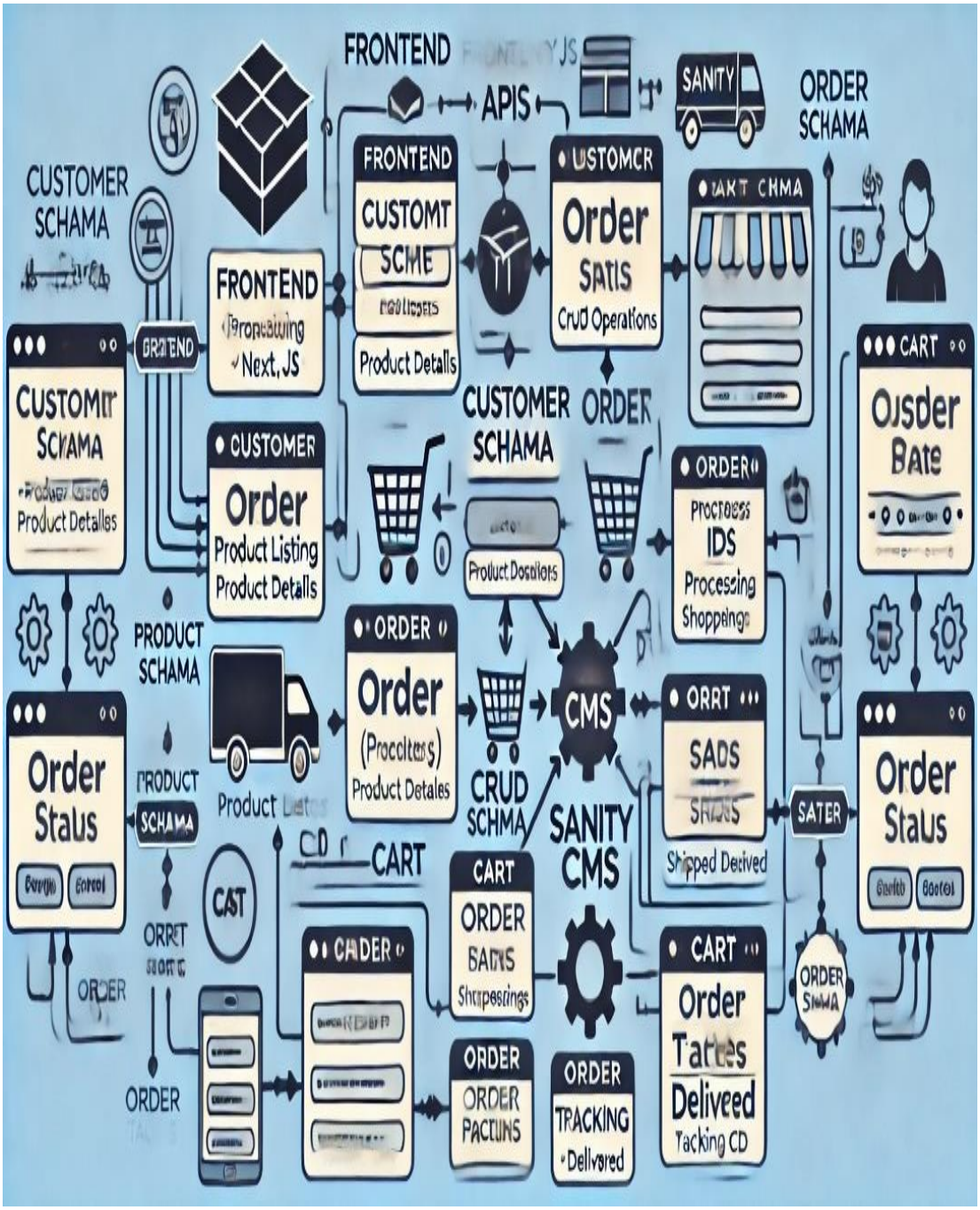
➢ Order Schema (Sanity):

- **order_id**: Unique identifier
- **customer_id**: Associated customer
- **items**: List of products purchased
- **total_price:** Total amount
- **status**: Processing | Shipped | Delivered
- **shipping_id:** Unique shipping identifier

- **tracking_id:** Tracking number assigned after dispatch

Technical Diagram

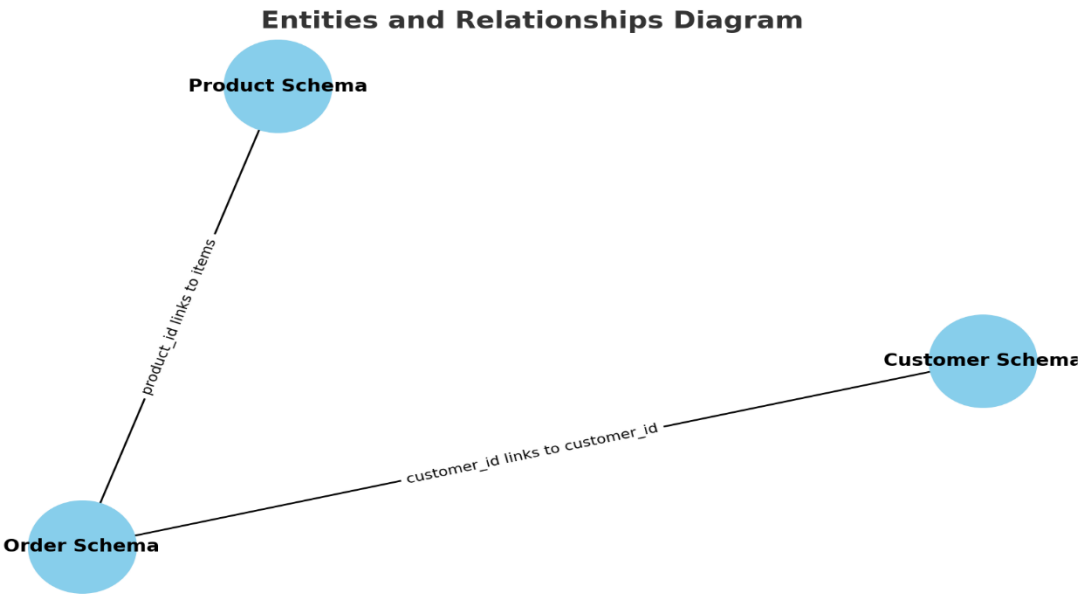
➢ Workflow Diagram



- **Homepage** (/):
  - Fetch product data from third-party API.
  - Display product listing.
- **Product Page** (/products/{product_id}):
  - Fetch product details from API.
  - Allow adding product to cart.
- **Cart Page** (/cart):
  - Display cart items stored in the **Cart Schema**.
  - Allow editing/removing items.
- **Checkout Page** (/checkout):
  - Create new customer and order records in Sanity CMS.
  - Generate a **Shipping ID**.
- **Order Tracking** (/order/{order_id}):
  - Display order details using **Order Schema**.
  - Update status from **Processing** to **Delivered**.

Entity-Relationship Diagram

Entities and Relationships:



1. **Product Schema:**
    - product_id links to items in the Order Schema.
2. **Customer Schema:**
    - customer_id links to customer_id in the Order Schema.
3. **Order Schema**:
    - Tracks order details, status, and shipping/tracking IDs.

Authentication and Security

❖ Features:

- **Secure Passwords:** Hashed using bcrypt.
- **JWT:** For user sessions.
- **Role-Based Access Control:** Separate access for admins and users.
- **HTTPS:** Encrypted data transmission.

Deployment and Hosting:

❖ Recommended Platforms:

- **Frontend:** Vercel or Netlify
- **Backend:** AWS EC2 or Heroku
- **Database:** AWS RDS or MongoDB Atlas

Testing and Quality Assurance:

❖ Types of Testing:

- **Unit Testing:** For individual functions/components.
- **Integration Testing:** For API and database interactions.
- **End-to-End Testing:** For the entire user journey using tools like Cypress.

Maintenance and Future Enhancements

❖ Maintenance:

- Regular updates for security patches.
- Database optimization for performance.

❖ Future Enhancements:

- AI-powered product recommendations.
- Multi-language support.
- Mobile application for iOS and Android.