# MARKETPLACE TECHNICAL FOUNDATION - URBANSTITCH

Prepared By: Mustafa Qazi

Student ID: 00010241

First published: PKT 10:00 PM, January 16, 2025

Document Revision Information

| Version | Date | Amendment | Author |
|---|---|---|---|
| 1.0 | 10:00 PM, January 16, 2025 | Initial release | Mustafa Qazi |
|  |  |  |  |

## Table of Contents

## Technical Plan Aligned with Business Goals

### 1. Introduction

This technical plan outlines the design, development, and implementation of a general eCommerce clothing store using Next.js, Sanity CMS, and ShipEngine API. The primary objective is to build a scalable, user-friendly, and feature-rich platform that aligns with the business goals of attracting customers, increasing sales, and enhancing operational efficiency.

### 2. Business Goals and Objectives

*Primary Business Goals:*

- Provide a seamless online shopping experience for customers.
- Facilitate efficient management of product inventory and promotions.
- Enable secure and fast order processing, shipping, and tracking.

*Technical Objectives:*

- Build a modern, responsive, and SEO-optimized web application.
- Use a headless CMS (Sanity CMS) for flexible and scalable content management.
- Integrate ShipEngine API for real-time shipping rates, label generation, and tracking.
- Ensure high performance, scalability, and robust security.

### 3. System Architecture Overview

***Frontend**:*

- Framework: **Next.js**
  - Server-Side Rendering (SSR) and Incremental Static Regeneration (ISR) for SEO and performance.
  - Dynamic routing for product categories and detail pages.
- Styling: **Tailwind CSS** for rapid UI development.
- State Management: **Context API** or for cart and user session handling.

***Backend**:*

- API Routes: Built into Next.js to handle server-side logic.
  - /api/products: Fetch product data from **Sanity CMS.**
  - /api/cart: Manage cart operations (add, remove, update).
  - /api/checkout: Process payments using Stripe / or any other payment gateway
  - /api/shipping: Integrate with ShipEngine for shipping operations.
  - /api/orders: Save and retrieve order details.

### *Content Management:*

- **Sanity CMS**
  - Product catalog management (name, description, pricing, images, inventory).
  - Promotional content (banners, discounts, campaigns).
  - Real-time updates via webhooks.

### *Shipping:*

- **Ship Engine API**
  - Fetch real-time shipping rates.
  - Generate shipping labels.
  - Provide tracking information to customers.

### *Payment Processing:*

- **Stripe API**
  - Secure payment handling with support for various **payment methods.**
  - Webhooks to handle payment status updates.

### *Database and Caching:*

- Primary Data Source: **Sanity CMS** (structured content storage).
- Caching: Redis (for frequently accessed data like product listings).

### *Hosting and Deployment:*

- Platform: **Vercel**
  - Supports Next.js features like ISR and serverless functions.
  - Built-in CI/CD pipelines for seamless deployments.

### *Monitoring and Analytics(Optional)*

- Tools: LogRocket and Sentry for error tracking.
- Google Analytics for user behavior insights.

## 4. Feature Breakdown

### *Customer-Facing Features:*

1. **Homepage**
   - Display featured collections, promotions, and trending products.
2. **Product Browsing**
   - Filters and sorting by category, price, size, and color.

3. **Product Detail Pages**
    - High-quality images, detailed descriptions, reviews, and size guides.
4. **Cart Management**
    - Add/remove items, quantity adjustments, and cart summary.
5. **Checkout Process**
    - Secure payment and shipping address collection.
6. **Order Tracking**
    - Customers can view real-time shipment status.

### *Admin Features:*

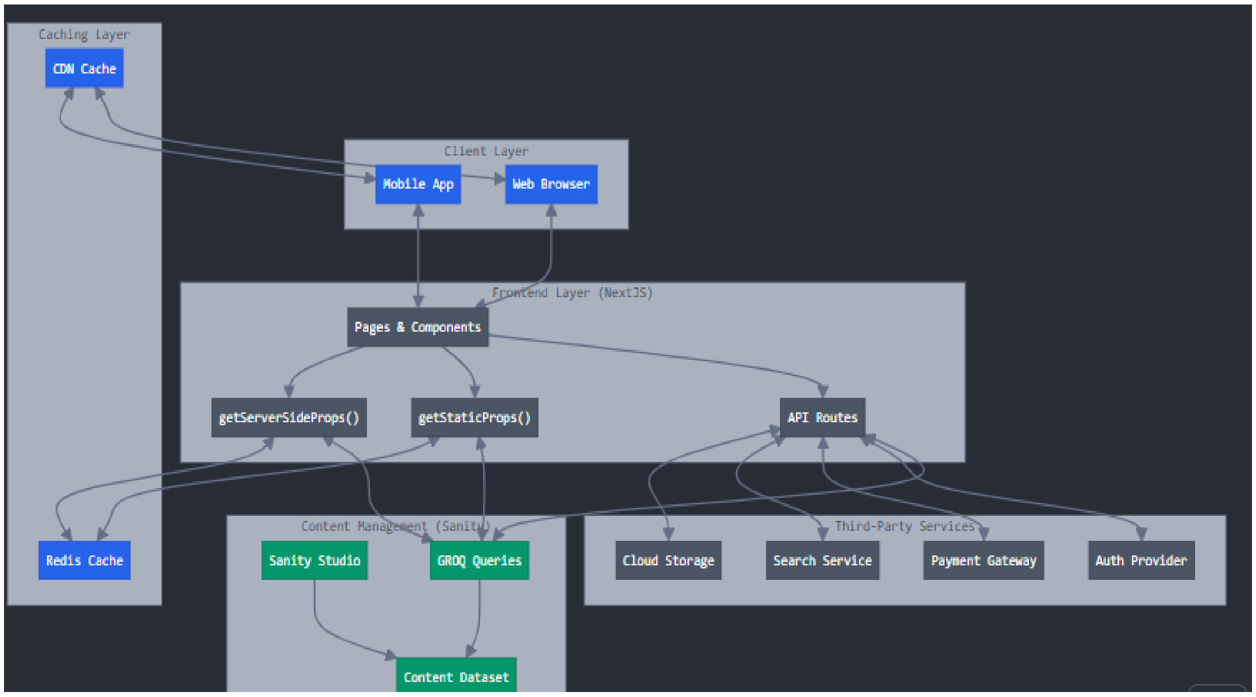
1. **Content Management**
    - Manage product details, inventory, and promotions through Sanity Studio.
2. **Order Management**
    - View and update order statuses.
3. **Shipping Management**
    - Generate and manage shipping labels via ShipEngine API.
4. **Analytics Dashboard**
    - Monitor sales, user engagement, and inventory levels.

## 5. Conclusion

This technical plan ensures that the eCommerce platform is aligned with the business's goals of providing an exceptional shopping experience, efficient operations, and scalability. By leveraging Next.js, Sanity CMS, and ShipEngine API, the platform will be robust, secure, and future-proof.

## 2. System Architecture Visualized

```
[Frontend (Next.js)] -- API Requests --> [Sanity CMS]
    |                                   ^
    |                                   |
    v                                   |
[Third-Party APIs] <-- Interactions -- [Sanity CMS]
```



## 3. Data Flow / Key Workflows

I've created a comprehensive data flow diagram that illustrates how data moves through the system. Here's a breakdown of the key data flows:

1. **Product Data Flow**

- Admins manage products and content through the Admin Panel
- Content updates flow through Sanity CMS to the product pages
- Product data is served to users through Next.js API

2. **User Session Flow**

- Authentication data is stored in SanityCMS (persistent)

- Session data is cached in Redis (temporary)
- Cart data is maintained in Redis for performance

3. **Order Processing Flow**

- Cart data converts to order data
- Payment processing through payment gateway
- Shipping calculations via ShipEngine
- Order data stored in SanityCMS

4. **Real-time Updates**

- Shipping status updates from ShipEngine
- Order status updates to users
- Cart updates in real-time

## 4. Detailed API Requirements Document

### Overview

- **Platform Name**: UrbanStitch Marketplace API
- **Frontend**: Next.js
- **Backend & Content Management**: Sanity CMS
- **Shipping Integration**: ShipEngine API
- **Payment Integration**: Stripe API
- **Authentication**: JSON Web Tokens (JWT) or API Key (based on user roles)
- **Base URL**:
    - Development & Production: vercel link here …………
- **Content Type**: JSON (application/json)

| Feature | EndPoint | Method | Description | Parameters | Response Example |
|---|---|---|---|---|---|
| Authentication | /auth/login | POST | User Login | email, password, userid, name, role | {<br>  "email": "user@example.com",<br>  "password": "password123"<br>}<br>Response (200):<br>{<br>  "token": "ey12345exampletoken",<br>  "expires_in": 3600,<br>  "user": {<br>    "id": "u1",<br>    "name": "Ali Asghar",<br>    "email": "aliasghar@gmail.com",<br>    "role": "customer"<br>  }<br>}<br>Response (401):<br>{<br>  "error": "Invalid email or password"<br>} |
| Product Management | /products | GET | Get All Products | id, name, desc, price, category, size, colors, image, instock | Response (200):<br>[<br>  {<br>    "id": "p1",<br>    "name": "Casual T-Shirt",<br>    "description": "100% cotton unisex t-shirt",<br>    "price": 19.99,<br>    "category": "T-Shirts",<br>    "sizes": ["S", "M", "L", "XL"],<br>    "colors": ["White", "Black"], |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | "image":<br>"https://cdn.example.com/tshirt1.jpg",<br>"in_stock": true<br>},<br>{<br>"id": "p2",<br>"name": "Denim Jacket",<br>"description": "Premium quality denim jacket",<br>"price": 59.99,<br>"category": "Jackets",<br>"sizes": ["M", "L"],<br>"colors": ["Blue"],<br>"image":<br>"https://cdn.example.com/jacket1.jpg",<br>"in_stock": false<br>}<br>] |
| Product Management | /products/{id} | GET | Get Products by ID | id, name, desc, price, category, size, colors, image, instock | • Response (200):<br>• {<br>• "id": "p1",<br>• "name": "Casual T-Shirt",<br>• "description": "100% cotton unisex t-shirt",<br>• "price": 19.99,<br>• "category": "T-Shirts",<br>• "sizes": ["S", "M", "L", "XL"],<br>• "colors": ["White", "Black"],<br>• "image":<br>"https://cdn.example.com/tshirt1.jpg",<br>• "in_stock": true<br>• } |

| | | | | | |
|---|---|---|---|---|---|
| Product Management | /products | POST | Create Product (Admin Only) | | Method: POST Request Body: { "name": "Leather Boots", "description": "Genuine leather boots for men and women", "price": 129.99, "category": "Footwear", "sizes": ["8", "9", "10", "11"], "colors": ["Brown", "Black"], "image": "https://cdn.example.com/boots1.jpg", "in_stock": true } Response (201): { "message": "Product created successfully", "product": { "id": "p3", "name": "Leather Boots", "price": 129.99 } } |
| Category Management | /categories | GET | Get All Categories | | Response (200): [ { "id": "c1", "name": "T-Shirts" }, { "id": "c2", "name": "Jackets" }, { "id": "c3", "name": "Footwear" } ] |

| | | | | | |
|---|---|---|---|---|---|
| | /categories | POST | Create Category (Admin Only) | | Request Body:<br>{<br>  "name": "Accessories"<br>}<br>Response (201):<br>{<br>  "message": "Category created successfully",<br>  "category": {<br>    "id": "c4",<br>    "name": "Accessories"<br>  }<br>} |
| Cart Management | /cart/add | POST | Add to Cart | | Request Body:<br>{<br>  "userId": "u1",<br>  "productId": "p1",<br>  "quantity": 2<br>}<br>Response (200):<br>{<br>  "message": "Product added to cart successfully",<br>  "cart": {<br>    "userId": "u1",<br>    "items": [<br>      {<br>        "productId": "p1",<br>        "quantity": 2<br>      }<br>    ]<br>  }<br>} |
| Order Management | /orders | POST | Create Order | | Request Body:<br>{<br>  "userId": "u1",<br>  "products": [<br>    { "productId": "p1", "quantity": 2 },<br>    { "productId": "p2", "quantity": 1 }<br>  ],<br>  "shippingAddress": {<br>    "line1": "123 Fashion St",<br>    "city": "Los Angeles",<br>    "state": "CA",<br>    "zip": "90001",<br>    "country": "US"<br>  }<br>} |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | Response (201):<br>{<br>"message": "Order created successfully",<br>"order": {<br>"id": "o1",<br>"totalAmount": 99.97<br>}<br>} |
| Payment Processing (Stripe API) | /payments/create-intent | POST | Create Payment Intent | | Request Body:<br>{<br>"amount": 99.97,<br>"currency": "USD"<br>}<br>Response (200):<br>{<br>"clientSecret": "pi_123456789_secret_abcdefgh"<br>} |

| | | | | | |
|---|---|---|---|---|---|
| Shipping Management (Ship Engine API) | /shipping/rates | POST | Get Shipping Rates | | Request Body: { "from": { "postalCode": "90001", "countryCode": "US" }, "to": { "postalCode": "10001", "countryCode": "US" }, "weight": { "value": 2.5, "unit": "pounds" } } Response (200): [ { "carrier": "UPS", "service": "Ground", "rate": 12.99, "estimatedDeliveryDays": 3 }, { "carrier": "FedEx", "service": "Express", "rate": 24.99, "estimatedDeliveryDays": 1 } ] |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
|  | /shipping/create-label | POST | Create Shipping Label |  | {<br>  "shipment": {<br>    "fromAddress": {<br>      "line1": "Warehouse 1",<br>      "city": "Los Angeles",<br>      "state": "CA",<br>      "zip": "90001",<br>      "country": "US"<br>    },<br>    "toAddress": {<br>      "line1": "123 Fashion St",<br>      "city": "New York",<br>      "state": "NY",<br>      "zip": "10001",<br>      "country": "US"<br>    },<br>    "weight": {<br>      "value": 2.5,<br>      "unit": "pounds"<br>    },<br>    "carrier": "UPS",<br>    "serviceCode": "ground"<br>  }<br>}<br>Response (200):<br>{<br>  "label": {<br>    "labelUrl": "https://cdn.example.com/shipping_label.pdf",<br>    "trackingNumber": "1Z12345E1234567890"<br>  }<br>} |
| Error Handling |  |  | General Error Format |  | Response (4xx/5xx):<br>{<br>  "error": "Error description",<br>  "details": "Optional details about the error"<br>} |

## 5. Sanity Schemas Design

**1. Product Schema**

Defines the structure for products in the marketplace, including key attributes like name, price, category, sizes, and colors.

### 1. Product Schema

| Field Name | Type | Title | Validation | Additional Options |
|---|---|---|---|---|
| name | string | Product Name | Required, max 100 characters | N/A |
| slug | slug | Slug | Required, max length 96 | Source: name |
| description | text | Description | Required, max 500 characters | N/A |
| price | number | Price | Required, minimum 0 | N/A |
| category | reference | Category | Required | References category type |
| sizes | array | Available Sizes | N/A | List: XS, S, M, L, XL, XXL |
| colors | array | Available Colors | N/A | Of type string |
| inStock | boolean | In Stock | N/A | Default value: true |
| images | array | Product Images | N/A | Of type image, hotspot enabled |

**2. Category Schema**

Organizes products into categories, such as "T-Shirts," "Jackets," or "Footwear."

| Field Name | Type | Title | Validation | Additional Options |
|---|---|---|---|---|
| name | string | Category Name | Required, max 50 characters | N/A |
| slug | slug | Slug | Required, max length 96 | Source: name |
| description | text | Description | Max 200 characters | N/A |

**3. Customer Schema**

Manages customer profiles, including contact information and addresses.

| Field Name | Type | Title | Validation | Additional Options |
|---|---|---|---|---|
| name | string | Full Name | Required | N/A |
| email | string | Email | Required, must be a valid email | N/A |
| phone | string | Phone Number | Required, regex validation for phone numbers | Format: ^\+?\d{10,15}$ |
| addresses | array | Addresses | N/A | Array of objects with fields: line1, city, state, zip, country |

**4. Order Schema**

Tracks customer orders, including product details, shipping, and payment status.

| Field Name | Type | Title | Validation | Additional Options |
|---|---|---|---|---|
| orderId | string | Order ID | Required | N/A |
| customer | reference | Customer | Required | References customer type |
| products | array | Products | N/A | Array of objects with fields: product, quantity, price |
| totalAmount | number | Total Amount | Required, minimum 0 | N/A |
| shippingAddress | object | Shipping Address | N/A | Fields: line1, city, state, zip, country |
| status | string | Order Status | N/A | Options: Pending, Processing, Shipped, Delivered, Canceled |
| paymentStatus | string | Payment Status | N/A | Options: Pending, Paid, Failed, Refunded |

**5. Review Schema**

Stores customer reviews for products.

| Field Name | Type | Title | Validation | Additional Options |
|---|---|---|---|---|
| product | reference | Product | Required | References product type |
| customer | reference | Customer | Required | References customer type |
| rating | number | Rating | Required, minimum 1, maximum 5 | N/A |

| Field Name | Type | Title | Validation | Additional Options |
|---|---|---|---|---|
| reviewText | text | Review Text | Max 300 characters | N/A |

## 6. Banner Schema

For Homepage Promotions

| Field Name | Type | Title | Validation | Additional Options |
|---|---|---|---|---|
| title | string | Title | Required | N/A |
| image | image | Banner Image | N/A | Hotspot enabled |
| link | url | Redirect Link | N/A | N/A |

**6. Collaboration Notes**

**Challenges Faced:**

**Data Modeling:** Defining flexible yet robust schemas in Sanity for dynamic data, such as products, categories, and customer profiles, required significant planning and iteration.

**Complex Relationships:** Managing references between schemas (e.g., linking products to categories and orders to customers) presented challenges in maintaining data integrity.

**Performance Issues:** Optimizing page load times while rendering dynamic content, especially for product images, required additional effort.

**Feature Prioritization:** Balancing feature requests, such as advanced filtering and search, with tight deadlines was a recurring challenge.