

# Technical Requirements for My Smart Scale Marketplace

## 1. Frontend Requirements (Next.js + Tailwind CSS)

**Objective:** Create a fully functional and responsive e-commerce platform.

**Key Features:**

- **User-Friendly Interface:**
  - Clean design with easy navigation.
  - Interactive components for product filtering and selection.
- **Responsive Design:**
  - Mobile, tablet, and desktop compatibility.
  - Flexbox/Grid layout for optimal display on all screen sizes.

### Essential Pages:

- **Home Page:** Introduction, featured products, and highlights.
- **Product Listing Page:** Display all smart scales with filtering options.
- **Product Details Page:** Detailed view with product descriptions, reviews, and health metrics.
- **Cart Page:** Cart summary with quantity adjustments.
- **Checkout Page:** Secure checkout process integrated with a payment gateway.
- **Order Confirmation Page:** Display order summary and tracking details.
- **About Page:** Information about the business and mission.
- **Contact Page:** Contact form and company information.
- **Blog Page:** Articles and health-related content for audience engagement.
- **Sign-in/Sign-out Page:** Authentication for user accounts.

### Frontend Tools:

- **Framework:** Next.js
  - **Styling:** Tailwind CSS
  - **Interactivity:** TypeScript for type safety and better code management.
-

## 2. Sanity CMS as Backend

**Objective:** Manage product data, orders, and customer information with a structured backend.

### Sanity CMS Features:

- **Product Management:**
  - Product name, description, price, image, stock, supplier details.
- **Order Management:**
  - Customer information, ordered products, total amount, order status.
- **Customer Management:**
  - Name, contact details, order history.
- **Review Management:**
  - Product ratings and customer reviews.
- **Supplier Management:**
  - Supplier details and associated products.

### Key Requirements:

- Design **Sanity Schemas** aligning with the business goals.
  - Use **Sanity Studio** for data entry and management.
- 

## 3. Third-Party API Integration

**Objective:** Enhance platform functionality with external services for payments and logistics.

### Key Integrations:

- **Payment Gateway (Stripe API):**
  - Secure online payments.
  - Credit card and digital wallet support.
- **Shipment Tracking (Shippo API):**
  - Real-time shipment tracking for customer orders.
- **Health App Integration:**
  - Compatibility with **Fitbit, MyFitnessPal, and Apple Health**.

### API Considerations:

- Ensure **secure API calls** with proper authentication.
  - Handle **error management** for failed API requests.
- 

Here's the **System Architecture Design** customized for my **Smart Scale**

### System Components & Data Flow (Next.js, Sanity CMS, Stripe, Shippo)

[Frontend (Next.js + Tailwind CSS)]

|

[Sanity CMS] -----> [Product Data API]

|

[Third-Party API] -----> [Shippo API for Shipment Tracking]

|

[Payment Gateway] -----> [Stripe API for Payment Processing]

---

### Data Flow and Interaction:

#### 1. User Browsing Products:

- User visits the **Next.js frontend**.
- Product data (smart scales) is fetched from **Sanity CMS** via the **Product Data API**.
- Products are displayed dynamically with filters for categories, health metrics, and price ranges.

#### 2. User Registration:

- User registers through the **Sign-up page**.
- Customer data is sent to **Sanity CMS** and stored in the **customer schema**.
- A confirmation message is sent to the user's email.

#### 3. Product Browsing:

- User selects a smart scale from the product listing.

- Product details, including images, specifications, and reviews, are fetched from **Sanity CMS**.
  - 4. **Order Placement:**
    - User adds items to the **cart** and proceeds to the **checkout page**.
    - **Order details** (customer info, product IDs, total price) are sent to **Sanity CMS** for record-keeping.
    - **Payment request** is sent to the **Stripe API** for secure transaction processing.
  - 5. **Shipment Tracking:**
    - After successful payment, **Shippo API** is called to create a shipment.
    - Tracking information is stored in **Sanity CMS** and displayed on the **Order Confirmation** page.
- 

### **Key Workflows Included:**

1. **User Registration:**
  - User signs up → Data stored in **Sanity CMS** → Confirmation sent via email.
2. **Product Browsing:**
  - User views product categories → Product Data API fetches products → Products dynamically displayed on frontend.
3. **Order Placement:**
  - User adds smart scales to the cart → Proceeds to checkout → Order details saved in **Sanity CMS**.
4. **Shipment Tracking:**
  - Shipment status updated using **Shippo API** → Tracking details visible on **Order Confirmation** page.
5. **Payment Processing:**
  - User completes the payment → **Stripe API** processes the payment securely → Payment status saved in **Sanity CMS**.

---

## System Components and Flow (Text-Based Representation)

1. **Frontend (Next.js + Tailwind CSS)**
  - Fetches data from the **Product Data API**.
  - Manages user registration, browsing, cart, and checkout.
2. **Sanity CMS (Backend)**
  - Stores product, order, and customer data.
  - Handles product availability and order management.
3. **Product Data API**
  - Fetches product details from **Sanity CMS**.
4. **Third-Party API Integrations:**
  - **Shippo API:** Fetches real-time shipment tracking.
  - **Stripe API:** Secure payment gateway for transactions.

## Data Flow Example:

- **Step 1:** User browses products (data fetched from Sanity CMS).
- **Step 2:** User adds items to the cart and checks out.
- **Step 3:** Order data sent to **Sanity CMS** and payment is processed via **Stripe API**.
- **Step 4:** Shipment data is retrieved using **Shippo API** and shown on the confirmation page.

## Sanity Schemas

### // Product Schema

```
export default {
  name: 'product',
  title: 'Product',
  type: 'document',
  fields: [
    { name: 'name', title: 'Product Name', type: 'string' },
    { name: 'description', title: 'Description', type: 'text' },
```

```

    { name: 'price', title: 'Price', type: 'number' },
    { name: 'image', title: 'Image', type: 'image', options: { hotspot: true } },
    { name: 'stockQuantity', title: 'Stock Quantity', type: 'number' },
    { name: 'category', title: 'Category', type: 'string' },
    { name: 'supplier', title: 'Supplier', type: 'reference', to: [{ type: 'supplier' }] },
    { name: 'sku', title: 'SKU (Stock Keeping Unit)', type: 'string' },
    { name: 'weight', title: 'Weight (in kg)', type: 'number' },
    { name: 'dimensions', title: 'Dimensions (LxWxH)', type: 'object',
      fields: [
        { name: 'length', title: 'Length', type: 'number' },
        { name: 'width', title: 'Width', type: 'number' },
        { name: 'height', title: 'Height', type: 'number' }
      ]
    },
    { name: 'tags', title: 'Tags', type: 'array', of: [{ type: 'string' }] },
    { name: 'ratings', title: 'Ratings', type: 'number' },
    { name: 'reviews', title: 'Reviews', type: 'array', of: [{ type: 'reference', to: [{ type:
'review' }] }] }
  ]
};

```

## // Customer Schema

```

export default {
  name: 'customer',
  title: 'Customer',
  type: 'document',
  fields: [
    { name: 'firstName', title: 'First Name', type: 'string' },
    { name: 'lastName', title: 'Last Name', type: 'string' },

```

```
    { name: 'email', title: 'Email', type: 'string' },
    { name: 'phone', title: 'Phone Number', type: 'string' },
    { name: 'address', title: 'Address', type: 'text' },
    { name: 'createdAt', title: 'Account Created At', type: 'datetime' }
  ]
};
```

### // Order Schema

```
export default {
  name: 'order',
  title: 'Order',
  type: 'document',
  fields: [
    { name: 'customer', title: 'Customer', type: 'reference', to: [{ type: 'customer' }] },
    { name: 'orderItems', title: 'Order Items', type: 'array', of: [{ type: 'reference', to: [{ type:
'product' }] }] },
    { name: 'totalAmount', title: 'Total Amount', type: 'number' },
    { name: 'paymentStatus', title: 'Payment Status', type: 'string' },
    { name: 'orderDate', title: 'Order Date', type: 'datetime' },
    { name: 'trackingNumber', title: 'Tracking Number', type: 'string' },
    { name: 'shipmentStatus', title: 'Shipment Status', type: 'string' }
  ]
};
```

### // Review Schema

```
export default {
  name: 'review',
  title: 'Review',
  type: 'document',
```

```
fields: [  
  { name: 'product', title: 'Product', type: 'reference', to: [{ type: 'product' }] },  
  { name: 'customer', title: 'Customer', type: 'reference', to: [{ type: 'customer' }] },  
  { name: 'rating', title: 'Rating', type: 'number' },  
  { name: 'reviewText', title: 'Review Text', type: 'text' },  
  { name: 'createdAt', title: 'Review Date', type: 'datetime' }  
]  
};
```

## // Supplier Schema

```
export default {  
  name: 'supplier',  
  title: 'Supplier',  
  type: 'document',  
  fields: [  
    { name: 'name', title: 'Supplier Name', type: 'string' },  
    { name: 'contactInfo', title: 'Contact Information', type: 'text' },  
    { name: 'address', title: 'Address', type: 'text' },  
    { name: 'productsSupplied', title: 'Products Supplied', type: 'array', of: [{ type:  
'reference', to: [{ type: 'product' }] }] }  
  ]  
};
```

## API Requirements for Your Smart Scale Marketplace

### 1. Product Endpoints

- **Endpoint Name:** /api/products
- **Method:** GET
- **Description:** Fetch all available product details from **Sanity CMS**.
- **Response Example:**



```
[
  {
    "id": "p123",
    "name": "Smart Scale X",
    "price": 120,
    "stockQuantity": 50,
    "category": "Fitness",
    "image": "https://example.com/smart-scale.jpg"
  }
]
```

- **Endpoint Name:** /api/products/:id
- **Method:** GET
- **Description:** Fetch details of a single product based on its ID.
- **Response Example:**

```
{
  "id": "p123",
  "name": "Smart Scale X",
  "price": 120,
  "stockQuantity": 50,
  "category": "Fitness",
  "description": "A smart scale with body composition analysis.",
  "image": "https://example.com/smart-scale.jpg"
}
```

## 2. Customer Endpoints

- **Endpoint Name:** /api/customers
- **Method:** POST
- **Description:** Create a new customer record in **Sanity CMS**.
- **Payload Example:**

```
{
```

```
"firstName": "John",  
"lastName": "Doe",  
"email": "john@example.com",  
"phone": "123-456-7890"  
}
```

- **Response Example:**

```
{  
  "customerId": "c001",  
  "status": "Customer created successfully"  
}
```

- **Endpoint Name:** /api/customers/:id
- **Method:** GET
- **Description:** Fetch details of a single customer based on ID.
- **Response Example:**

```
{  
  "id": "c001",  
  "firstName": "John",  
  "lastName": "Doe",  
  "email": "john@example.com",  
  "phone": "123-456-7890"  
}
```

### 3. Order Endpoints

- **Endpoint Name:** /api/orders
- **Method:** POST
- **Description:** Create a new order in **Sanity CMS**.
- **Payload Example:**

```
{  
  "customerId": "c001",
```

```
"orderItems": [  
  { "productId": "p123", "quantity": 2 }  
],  
"totalAmount": 240,  
"paymentStatus": "Paid"  
}
```

- **Response Example:**

```
{  
  "orderId": "o001",  
  "status": "Order created successfully"  
}
```

- **Endpoint Name:** /api/orders/:id
- **Method:** GET
- **Description:** Fetch order details based on the order ID.
- **Response Example:**

```
{  
  "orderId": "o001",  
  "customerId": "c001",  
  "orderItems": [  
    { "productId": "p123", "quantity": 2 }  
  ],  
  "totalAmount": 240,  
  "paymentStatus": "Paid"  
}
```

#### 4. Payment Endpoints (Stripe Integration)

- **Endpoint Name:** /api/payments
- **Method:** POST
- **Description:** Process payment using **Stripe API**.
- **Payload Example:**

```
{
  "orderId": "o001",
  "paymentMethod": "credit_card",
  "amount": 240
}
```

- **Response Example:**

```
{
  "paymentId": "pay001",
  "status": "Payment Successful"
}
```

## 5. Shipment Tracking Endpoints (Shippo Integration)

- **Endpoint Name:** /api/shipment/:trackingNumber
- **Method:** GET
- **Description:** Fetch shipment tracking information via **Shippo API**.
- **Response Example:**

```
{
  "shipmentId": "s001",
  "trackingNumber": "TR123456",
  "status": "In Transit",
  "estimatedDelivery": "2025-01-20"
}
```

## Summary of Key API Requirements:

1. **Product API:** Fetch all products and individual product details.
2. **Customer API:** Create and fetch customer records.
3. **Order API:** Create and retrieve order details.
4. **Payment API:** Secure payment processing using **Stripe**.
5. **Shipment API:** Real-time shipment tracking using **Shippo**.

# Technical DOCUMENTATION

## Marketplace Technical Foundation - Smart Scale Marketplace

### 1. System Architecture Overview

#### Components:

- **Frontend (Next.js + Tailwind CSS):** Manages the user interface, browsing, and interaction.
- **Sanity CMS (Backend):** Manages product, order, customer, and supplier data.
- **Third-Party APIs:**
  - **Stripe API:** Manages secure payment processing.
  - **Shippo API:** Provides shipment tracking and logistics information.

#### System Interaction Flow:

1. **User Interaction:** User accesses the marketplace via the frontend.
  2. **Product Data Fetching:** The frontend fetches product data from **Sanity CMS**.
  3. **Order Management:** Order details are stored in **Sanity CMS**.
  4. **Payment Processing:** Payments are securely processed via **Stripe API**.
  5. **Shipment Tracking:** Real-time shipping details are fetched using **Shippo API**.
- 

### 2. Key Workflows

#### User Registration:

- User signs up through the frontend.
- Data is stored in **Sanity CMS** with confirmation sent to the user's email.

#### Product Browsing:

- User views the product catalog.
- Frontend calls `/api/products` endpoint.
- **Sanity CMS** returns the product listing.

#### Order Placement:

- User adds products to the cart and proceeds to checkout.
- Order data is stored in **Sanity CMS**.

- Payment is processed via **Stripe API**.

#### **Shipment Tracking:**

- Shipment information is fetched from **Shippo API**.
  - Updated shipment data is stored in **Sanity CMS** and shown on the frontend.
- 

### **3. API Specification Document**

#### **Product Endpoints:**

- **Endpoint:** /api/products
- **Method:** GET
- **Description:** Fetch all products.
- **Response Example:**

```
{  
  "id": "p123",  
  "name": "Smart Scale X",  
  "price": 120  
}
```

#### **Order Endpoints:**

- **Endpoint:** /api/orders
- **Method:** POST
- **Description:** Create a new order.
- **Payload:**

```
{  
  "customerId": "c001",  
  "orderItems": [{ "productId": "p123", "quantity": 2 }],  
  "totalAmount": 240  
}
```

---

### **4. Data Schema Design**

#### **Product Schema:**

- name: String
- description: Text
- price: Number
- stockQuantity: Number
- category: String
- supplier: Reference to Supplier

#### **Customer Schema:**

- firstName: String
- lastName: String
- email: String
- phone: String

#### **Order Schema:**

- customer: Reference to Customer
- orderItems: Array (Product references)
- totalAmount: Number
- paymentStatus: String

#### **Review Schema:**

- product: Reference to Product
- customer: Reference to Customer
- rating: Number
- reviewText: Text

---

## **5. Technical Roadmap**

### **Milestone 1: Setup Project Environment**

- Initialize **Next.js** and **Sanity CMS**.
- Configure version control with **GitHub**.

### **Milestone 2: Develop Core Pages**

- Build **Home**, **Product Listing**, **Product Details**, **Cart**, **Checkout**, **About**, **Contact**, **Blog** pages.

### Milestone 3: Backend Configuration

- Set up **Sanity CMS** schemas for products, orders, customers, suppliers, and reviews.

### Milestone 4: API Integration

- Integrate **Stripe API** for payment processing.
- Integrate **Shippo API** for real-time shipment tracking.

### Milestone 5: Testing & Deployment

- Perform unit and integration testing.
- Optimize performance for scalability.
- Deploy to **Vercel** for production.

---

## 6. Security & Best Practices

- **Data Protection:** Implement SSL encryption for all transactions.
- **User Privacy:** Ensure compliance with GDPR standards.
- **Version Control:** Use **GitHub** for collaborative development.

---

This comprehensive technical foundation ensures a professional, market-ready product for the Smart Scale Marketplace.