### **Name: Usra Naz**

**Roll No: 00144665**

**Day/Time: Friday- 09:00AM -12:00PM**

### 

### 

### 

### 

### 

### **Marketplace Technical Foundation - [Clothing E-Commerce]**

### 

### 

### 

### 

### 

### 

### 

### 

### 

### **Technical Requirements for Clothing eCommerce Website**

This document outlines the technical requirements for developing a clothing eCommerce marketplace. It includes detailed descriptions of the frontend, backend, and integration with third-party services to ensure a seamless and scalable platform.

### 

### 

### 

### 

### 

### 

### 

### 

### **1. Frontend Requirements**

The frontend will serve as the user-facing part of the marketplace, designed to provide an intuitive and engaging shopping experience.

#### **1.1 User-Friendly Interface**

* The design will be clean and simple, making it easy for users to navigate and find products.
* Products will be categorized clearly into sections such as Men’s Clothing, Women’s Clothing, and Kids’ Clothing.

#### **1.2 Responsive Design**

* The website will adapt seamlessly to different screen sizes, ensuring compatibility with both mobile and desktop devices.

#### **1.3 Essential Pages**

The website will consist of the following core pages:

1. **Home Page**:  
   * **Purpose**: The first point of interaction for users; highlights the website's features and offerings.
   * **Key Sections**:
     + **Hero Banner**: Display featured collections, discounts, or promotions.
     + **New Arrivals**: Showcase the latest products.
     + **Bestsellers**: Highlight popular items.
     + **Categories**: Quick links to Men's, Women's, Kids' clothing.
     + **Footer**: Links to policies, contact information, and social media.
2. **Product Listing Page**:  
   * **Purpose**: Display all products in a specific category or based on user filters.
   * **Key Features**:
     + **Product Grid/List**: Show products with images, names, and prices.
     + **Filters**: Allow users to filter by size, color, price range, and brand.
     + **Sort Options**: Sort products by price, popularity, or newest arrivals.
     + **Pagination or Infinite Scrolling**: Load more products as users scroll.
3. **Product Details Page**:  
   * **Purpose**: Provide detailed information about a selected product.
   * **Key Features**:
     + **Product Images**: Zoom-in functionality for clear views.
     + **Description**: Detailed information about the product (fabric, care instructions, etc.).
     + **Price**: Show the current price and any discounts.
     + **Size and Color Options**: Allow users to select size and color.
     + **Stock Availability**: Indicate whether the item is in stock.
     + **Add to Cart Button**: Add the item to the cart with selected options.
     + **Reviews and Ratings**: Show user reviews and ratings for the product.
4. **Cart Page**:  
   * **Purpose**: Summarize items the user has added to their cart for purchase.
   * **Key Features**:
     + **Product List**: Show products with selected size, color, price, and quantity.
     + **Total Cost**: Display the subtotal, taxes, and final cost.
     + **Edit Options**: Allow users to update quantities or remove items.
     + **Proceed to Checkout Button**: Directs users to the checkout page.
5. **Checkout Page**:  
   * **Purpose**: Collect user details for order processing.
   * **Key Sections**:
     + **Shipping Details**: Collect user name, address, phone number, and email.
     + **Payment Information**: Display payment options (credit card, PayPal, etc.).
     + **Order Summary**: Summarize items being purchased.
     + **Place Order Button**: Complete the transaction and send order details to the backend.
6. **Order Confirmation Page**:  
   * **Purpose**: Confirm the successful placement of an order.
   * **Key Features**:
     + **Order Summary**: List purchased items with quantities and price.
     + **Tracking ID**: Display shipment tracking information (if available).
     + **Thank You Message**: Acknowledge the customer for their purchase.
     + **Next Steps**: Links to track the order or continue shopping.
7. **User Registration Page**:  
   * **Purpose**: Allow new users to create an account.
   * **Key Features**:
     + **Input Fields**: Name, email, password, and phone number.
     + **Sign Up Button**: Save user details in the backend.
8. **Login Page**:  
   * **Purpose**: Enable existing users to log in.
   * **Key Features**:
     + **Input Fields**: Email and password.
     + **Forgot Password Link**: Help users recover access to their account.
9. **User Profile Page**:  
   * **Purpose**: Allow users to view and manage their personal details.
   * **Key Sections**:
     + **Personal Information**: Name, email, phone number, and address.
     + **Order History**: List of previous orders with statuses.
     + **Wishlist**: Show saved products for later purchase.
10. **Wishlist Page**:  
    * **Purpose**: Display products that users have saved for future consideration.
    * **Key Features**:
      + **Product List**: Show saved items with prices and stock status.
      + **Move to Cart Option**: Allow users to add wishlist items to their cart.
11. **Search Results Page**:  
    * **Purpose**: Display products matching the user’s search query.
    * **Key Features**:
      + **Search Bar**: Input field for user queries.
      + **Product Grid/List**: Show results matching the search terms.
12. **About Us Page**:  
    * **Purpose**: Share information about your brand and story.
    * **Key Features**:
      + **Mission and Vision**: Explain your goals and values.
      + **Brand Story**: Describe how your business started.
      + **Contact Information**: Provide email, phone, and address details.
13. **Contact Us Page**:  
    * **Purpose**: Allow users to reach out with inquiries or issues.
    * **Key Features**:
      + **Contact Form**: Fields for name, email, subject, and message.
      + **Business Details**: Email, phone number, and physical address.
      + **Social Media Links**: Direct users to your social media platforms.
14. **FAQ Page**:  
    * **Purpose**: Address common customer queries.
    * **Key Sections**:
      + **Shipping**: Delivery timelines and fees.
      + **Returns**: Process for returning items.
      + **Payments**: Accepted payment methods and refund policies.
15. **Privacy Policy and Terms of Service Page**:  
    * **Purpose**: Inform users about your policies.
    * **Key Sections**:
      + **Data Collection**: Explain how user data is collected and used.
      + **Terms of Use**: Guidelines for using the website.

#### **Summary of Pages**

* Home Page
* Product Listing Page
* Product Details Page
* Cart Page
* Checkout Page
* Order Confirmation Page
* User Registration Page
* Login Page
* User Profile Page
* Wishlist Page
* Search Results Page
* About Us Page
* Contact Us Page
* FAQ Page
* Privacy Policy and Terms of Service Page

### 

### 

### 

### **2. Sanity CMS as Backend**

Sanity CMS will act as the backbone for managing all dynamic content and data related to the website.

#### **2.1 Product Data Management**

* Products will be managed with fields such as:
  + Name, Description, Price
  + Available Sizes and Colors
  + Stock Levels
  + Product Images

#### **2.2 Customer Data Management**

* Maintain user profiles, including order history and personal details.

#### **2.3 Order Records**

* Store information about all orders placed, such as:
  + Product details (name, quantity, price).
  + Customer details (name, address, contact information).
  + Payment status and shipping details.

**3. Third-Party APIs**

To enhance functionality and automate backend tasks, the platform will integrate with the following APIs:

#### **3.1 Shipment Tracking API**

* Provides real-time updates on shipment status.
* Example APIs: **Shippo**, **EasyPost**.
* **Use Case**:
  + Display live shipment tracking information on the Order Confirmation Page.

#### **3.2 Payment Gateway API**

* Securely handle payment processing.
* Example APIs: **Stripe**, **PayPal**, **Razorpay**.
* **Use Case**:
  + Process transactions during checkout and return a confirmation status.

#### **3.3 Optional APIs**

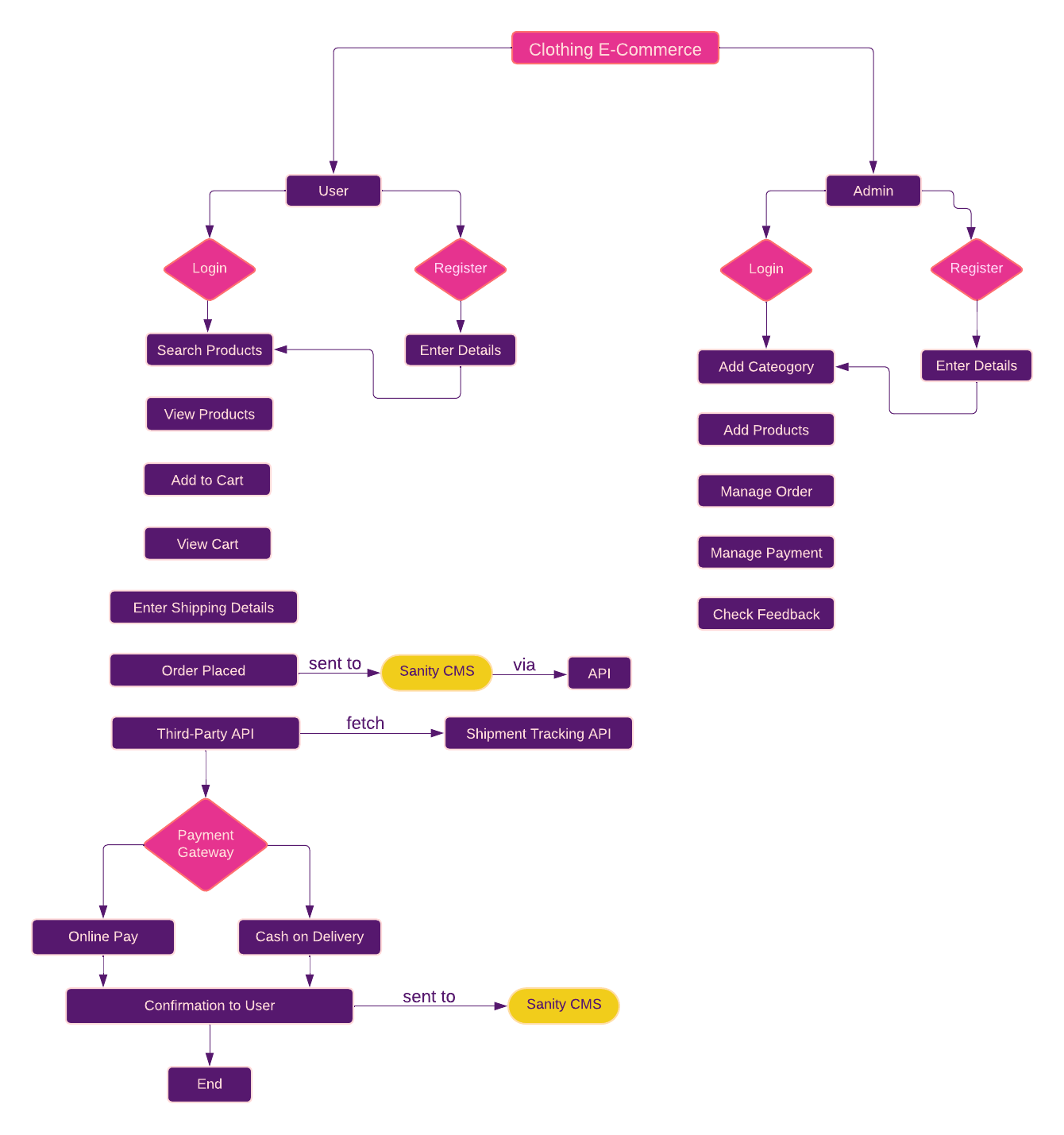
* **SMS/Email Notifications**: Notify customers about order confirmations, shipping updates, etc.
  + Example APIs: **Twilio**, **SendGrid**.

### **How These Requirements Align with Business Goals**

1. **User-Friendly Design**:  
   * A clean interface and responsive design attract and retain customers, ensuring a pleasant shopping experience.
2. **Sanity CMS**:  
   * Centralized data management simplifies inventory updates and ensures scalability as the business grows.
3. **Third-Party APIs**:  
   * Automates essential tasks like payment processing and shipment tracking, improving operational efficiency and customer satisfaction.

This technical plan forms the foundation for a robust eCommerce platform that meets customer needs while supporting business scalability.

### **Key Workflow**



### **System Architecture Overview**

#### **Frontend (Next.js) Interaction with Sanity CMS**

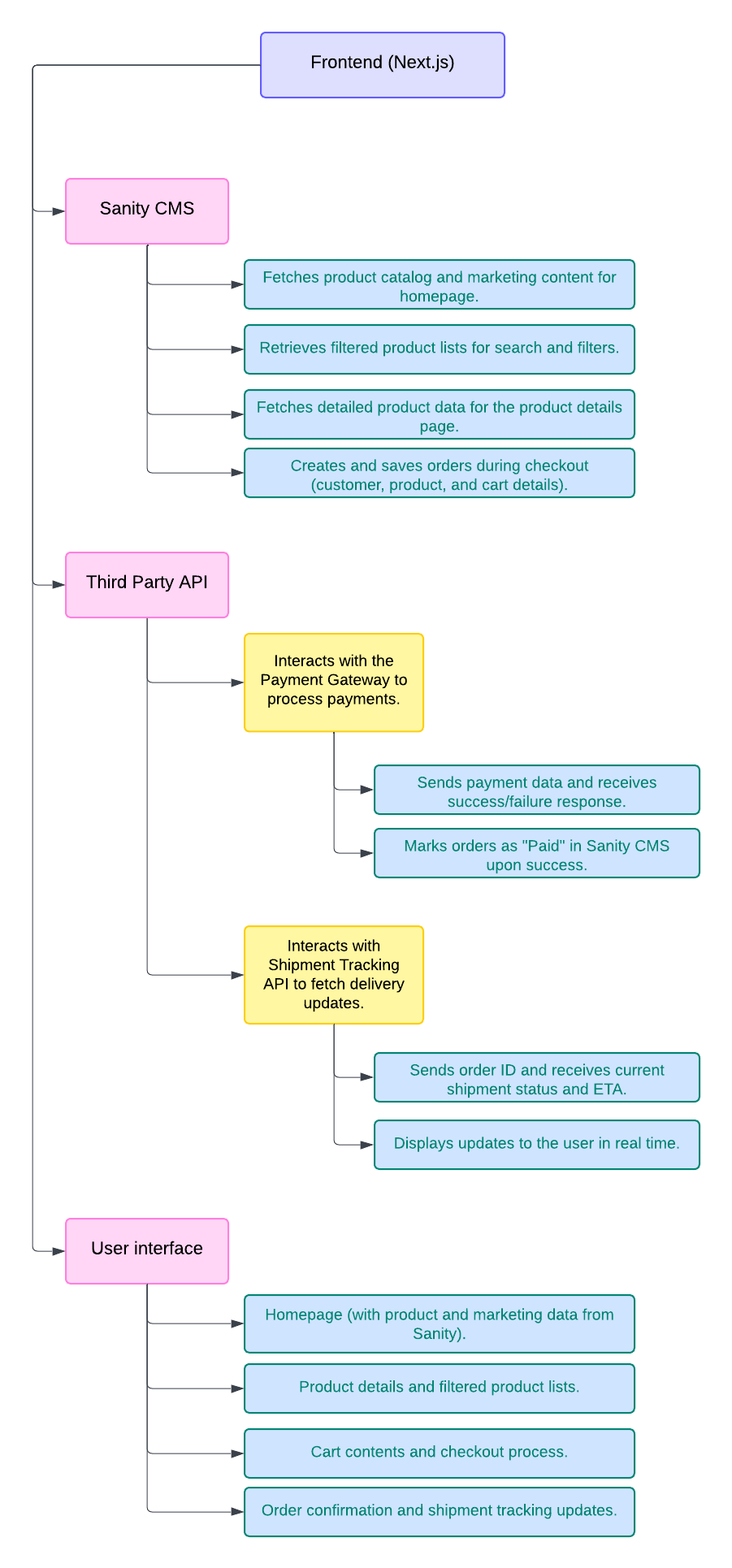
1. **Home Page/Product List Loading:**
   * User visits the website.
   * **Frontend Request:** Sends a GET request to Sanity CMS to fetch:
     + Product catalog (ID, name, image, price, stock status).
     + Promotions, banners, and featured products.
   * **Sanity CMS Response:** Returns structured data (JSON) for products and homepage content.
   * **Frontend Action:** Dynamically renders the homepage with product information and marketing content.
2. **Search/Filter Products:**
   * User interacts with search or filter options.
   * **Frontend Request:** Sends a query to Sanity CMS, including:
     + Keywords (search term).
     + Selected filter criteria (e.g., category, price range).
   * **Sanity CMS Response:** Returns a refined list of products matching the criteria.
   * **Frontend Action:** Updates the UI to display filtered products.
3. **Product Details Page:**
   * User clicks on a product to view details.
   * **Frontend Request:** Sends a GET request to Sanity CMS with the product ID.
   * **Sanity CMS Response:** Returns detailed product information:
     + Name, description, price, stock, reviews, and images.
   * **Frontend Action:** Renders the product details page with all the fetched information.
4. **Cart Management:**
   * User adds/removes products from the cart.
   * **Frontend Action:** Updates the cart state locally (e.g., in Redux or localStorage).
   * Optionally syncs the cart state with Sanity CMS for logged-in users.
5. **Checkout Process:**
   * User initiates checkout.
   * **Frontend Action:**
     + Collects user information (e.g., name, email, shipping address).
     + Gathers cart details (product IDs, quantities, total price).
   * **Frontend Request:** Sends a POST request to Sanity CMS to create a new order.
   * **Sanity CMS Response:**
     + Saves order data.
     + Returns an order confirmation ID.

#### **Frontend Interaction with Third-Party APIs**

1. **Payment Gateway:**
   * During checkout, user provides payment details.
   * **Frontend Request:**
     + Sends payment data (amount, card details, user info) to the third-party payment gateway.
   * **Payment Gateway Response:**
     + Returns payment status (success or failure).
     + If successful:
       - Returns a transaction ID and confirmation receipt.
       - **Frontend Action:** Updates the UI with a success message.
       - **Sanity CMS Request:** Marks the order as "Paid."
     + If failed:
       - Returns an error message.
       - **Frontend Action:** Prompts the user to retry payment.
2. **Shipment Tracking:**
   * After order placement, shipment status becomes important.
   * **Frontend Request:** Sends the order ID to the third-party shipment tracking API.
   * **Shipment Tracking API Response:**
     + Provides real-time shipment updates:
       - Current status (e.g., "Dispatched," "In Transit").
       - Expected delivery date.
       - Tracking URL for further details.
     + **Frontend Action:** Displays shipment updates on the user dashboard or order history page.
3. **Real-Time Updates:**
   * The frontend periodically polls the third-party APIs for updates or uses WebSockets for real-time communication (if supported by APIs).
   * **Frontend Action:** Reflects real-time changes in:
     + Payment status (e.g., awaiting payment confirmation).
     + Shipment tracking (e.g., updated ETA).

### 

### **System Architecture Diagram**



### **API Endpoint**

### **Schema**

