

Open in app ↗

Medium

 Search Write

# Technical Requirements for Sofa and Furniture E-Commerce Website

Frontend



Huda Noor

6 min read · 1 day ago



1



## Technology Stack

- Framework: Next.js
- A React-based framework that enables server-side rendering (SSR) and static site generation (SSG), perfect for building high-performance and SEO-friendly applications.
- UI Library: shadcn/ui
- Provides pre-built, customizable UI components that work seamlessly with Tailwind CSS.
- Styling Framework: Tailwind CSS
- A utility-first CSS framework for fast and responsive design.

# Pages to Create

## 1. Home Page

- Features:
- Display promotional banners for seasonal sales or new arrivals.
- Showcase featured products (e.g., best-selling sofas).
- Highlight product categories (e.g., Sofas, Chairs, Tables, Outdoor Furniture).
- Libraries/Features to Use:
- shadcn/ui Components:
- Card for product display.
- Button for call-to-action (e.g., “Shop Now”).
- Carousel for promotional banners.
- Tailwind CSS for responsive design.

## 2. Product Listing Page

- Features:
- Show all available products with options for filtering (by category, price range, material) and sorting (by price, popularity).
- Libraries/Features to Use:
- shadcn/ui Components:

- Select for filtering options.
- Input for search functionality.
- Grid for product layout.
- Dynamic Routes:
- Use `pages/products/[category].tsx` for category-based listings.

### 3. Product Details Page

- Features:
- Display detailed product information including:
- Image gallery (multiple images of the sofa).
- Price.
- Description.
- Stock status (availability).
- Customization options (size, fabric, color).
- Libraries/Features to Use:
- shadcn/ui Components:
- Modal for displaying additional product information.
- Carousel for image gallery.
- Tabs for product details, reviews, and customization options.
- Image Optimization:

- Use Next.js `<Image>` component for optimized images.

## 4. Cart Page

- Features:
- View selected products with options to update quantities.
- Display a summary of the total price, including shipping costs.
- Libraries/Features to Use:
- shadcn/ui Components:
- Table for displaying cart items.
- Badge for showing the number of items in the cart.
- State Management:
- Persist cart data using localStorage or React Context API.

## 5. Checkout Page

- Features:
- Collect user information (shipping address, contact details).
- Process payments securely.
- Libraries/Features to Use:
- shadcn/ui Components:

- Form for user input.
- Input fields for address and payment details.
- Payment Gateway Integration:
- Integrate a payment gateway like Stripe for processing payments.

## 1. Products Schema

javascript

VerifyCopy [code](#)

```
1export default {
2  name: 'product',
3  type: 'document',
4  fields: [
5    { name: 'id', type: 'string', title: 'Product ID', readOnly: true },
6    { name: 'name', type: 'string', title: 'Product Name' },
7    { name: 'price', type: 'number', title: 'Price' },
8    { name: 'stock', type: 'number', title: 'Stock Level' },
9    { name: 'category', type: 'string', title: 'Category' },
10   { name: 'dimensions', type: 'string', title: 'Dimensions' },
11   { name: 'material', type: 'string', title: 'Material' },
12   { name: 'description', type: 'text', title: 'Description' },
13   { name: 'customizationOptions', type: 'array', title: 'Customization Options' },
14   { name: 'image', type: 'image', title: 'Product Image' }
15 ]
16};
```

## 2. Customers Schema

javascript

VerifyCopy code

```
1 export default {
2   name: 'customer',
3   type: 'document',
4   fields: [
5     { name: 'id', type: 'string', title: 'Customer ID', readOnly: true },
6     { name: 'name', type: 'string', title: 'Customer Name' },
7     { name: 'email', type: 'string', title: 'Email', validation: Rule => Rule.required },
8     { name: 'phone', type: 'string', title: 'Phone Number' },
9     { name: 'address', type: 'text', title: 'Address' },
10    { name: 'createdAt', type: 'datetime', title: 'Account Created At', readOnly: true },
11  ]
12};
```

### 3. Orders Schema

javascript

VerifyCopy code

```
1 export default {
2   name: 'order',
3   type: 'document',
4   fields: [
5     { name: 'id', type: 'string', title: 'Order ID', readOnly: true },
6     { name: 'customerID', type: 'reference', to: [{ type: 'customer' }], title: 'Customer' },
7     { name: 'items', type: 'array', title: 'Items' },
8     { name: 'total', type: 'number', title: 'Total' },
9     { name: 'status', type: 'string', title: 'Status' },
10    { name: 'createdAt', type: 'datetime', title: 'Order Created At', readOnly: true },
11  ]
12};
```

```

7   { name: 'orderDate', type: 'datetime', title: 'Order Date', readOnly: true },
8   { name: 'totalAmount', type: 'number', title: 'Total Amount' },
9   { name: 'status', type: 'string', title: 'Order Status', options: { list: ['Pending', 'Completed', 'Cancelled'] } },
10  ],
11};

```

## 4. OrderItems Schema

javascript

VerifyCopy code

```

1export default {
2  name: 'orderItem',
3  type: 'document',
4  fields: [
5    { name: 'id', type: 'string', title: 'Order Item ID', readOnly: true },
6    { name: 'orderId', type: 'reference', to: [{ type: 'order' }], title: 'Order' },
7    { name: 'productId', type: 'reference', to: [{ type: 'product' }], title: 'Product' },
8    { name: 'quantity', type: 'number', title: 'Quantity' },
9    { name: 'pricePerUnit', type: 'number', title: 'Price Per Unit' }
10  ],
11};

```

## Third-Party APIs

### APIs Required:

#### 1. Payment Gateway API:

- Purpose: For secure transaction processing.
- Example:
- JazzCash API or EasyPaisa API for processing payments in Pakistan.

#### 1. Shipment Tracking API:

- Purpose: To provide real-time shipment updates.
- Example:
- TCS API or DHL API for tracking shipments within Pakistan.

#### 1. Email Notification API:

- Purpose: For sending order confirmations and notifications to customers.
- Example:
- SendGrid API or Mailgun API for sending transactional emails.

#### 1. Product Information API:

- Purpose: To fetch product details and availability.
- Example:
- Custom API that connects to your Sanity CMS to retrieve product data.

## System Architecture Design



VerifyCopy **code**



## 9/16

- Users interact with the frontend to browse products, add items to the cart, and proceed to checkout.

## 1. Frontend (Next.js):

- The Next.js application handles user interactions, displays product listings, and manages the shopping cart.

### 1. Fetch Product Data:

- The frontend fetches product data from the Sanity CMS, which stores all product information, including details like name, price, stock, and customization options.

### 1. Fetch Order Data:

- When a user places an order, the frontend sends the order details to the Sanity CMS for processing.

### 1. Sanity CMS:

- Acts as the backend content management system where product and order data is stored and managed.

### 1. API Integration:

- Integrates with third-party APIs for:
- Payment Processing: Securely handle transactions using the selected payment gateway API (e.g., JazzCash or EasyPaisa).

- **Shipment Tracking:** Use the shipment tracking API (e.g., TCS or DHL) to provide real-time updates to customers about their orders.

## 1. Third-party APIs:

- **Email Notification API:** Sends order confirmation emails to customers using services like SendGrid or Mailgun.

# Workflows

## 1. Product Data Fetching:

- **Process:**
- The frontend requests product data from Sanity CMS.
- Sanity CMS retrieves and returns the data in JSON format.

## 2. Order Processing:

- **Process:**
- The frontend sends order details to the API endpoint.
- The API saves the order in Sanity CMS.
- Notification and payment processing APIs are triggered.

## 3. Shipment Integration:

- **Process:**
- The frontend fetches the shipment status from the Shipment Tracking API.
- The status is displayed to the user on the Order Details page.

# API Endpoints Plan

## 1. Product Endpoints

- `/products` (GET)
- Description: Fetch all product details.
- Response Example:
- `json`
- VerifyCopy code
- ```
1[ 2 { 3 "id": "1", 4 "name": "Outdoor Sofa", 5 "price": 499.99, 6 "stock": 20, 7 "material": "Patex Wood", 8 "durability": "5 years guarantee", 9 "customizationOptions": ["Size", "Fabric", "Color"] 10 } 11]
```

## 2. Order Endpoints

- `/orders` (POST)
- Description: Save new order details.
- Request Example:
- `json`
- VerifyCopy code
- ```
1{ 2 "customerId": "123", 3 "products": ["1", "2"], 4 "total": 999.98 5}
```

## 3. Shipment Endpoints

- `/shipment` (GET)

- **Description:** Fetch the shipment status for an order.
- **Response Example:**
- **json**
- **VerifyCopy code**
- ```
1{ 2 "orderId": "123", 3 "status": "In Transit" 4}
```

## Sofa and Furniture E-Commerce Platform

Your platform specializes in offering high-quality outdoor furniture, including customizable sofas made from premium Patex wood and equipped with Master Molty Foam for maximum comfort and durability. Each product comes with a 5-year guarantee on durability and a 1-year bug spray service to ensure your furniture remains in pristine condition.

### Key Features:

- **Customizable Sizes:** Customers can choose the size of their sofas to fit their outdoor spaces perfectly.
- **Durability Assurance:** All products are crafted with Patex wood and Master Molty Foam, ensuring long-lasting quality.
- **Seamless Shopping Experience:** Users can easily browse, customize, and purchase their desired outdoor furniture through a user-friendly interface.
- **Reliable Customer Support:** Dedicated support to assist customers with their inquiries and after-sales services.