

Technical Requirements for Furniture E-Commerce Marketplace

Frontend Requirements :

User-Friendly Interface: Intuitive browsing with clear navigation.

Responsive Design:

Seamless experience across mobile and desktop devices.

Essential Pages:

1. Home
2. Products
3. Pages
4. Cart
5. Checkout
6. Order Confirmation
7. About
8. Faq
9. Contact

Backend (Sanity CMS) :

Content Management:

Manage product data, customer details, and orders.

Create schemas for:

Products, Customers, Orders, and Categories.

Third-Party APIs:

Payment Gateway:

Secure integration for transaction processing.

Shipment Tracking:

Enable order tracking for customers

Tech Stack :

Frontend:

Next.js, TypeScript, Tailwind CSS.

Backend:

Sanity CMS.

System Architecture Visualized:

[Frontend (Next.js)]

[Sanity CMS]

— User Management:

— User Registration:

- Users sign up on the frontend.
- Data is securely stored in Sanity CMS.

— Product Management:

— Product Data API:

- Fetch product categories and details.

— Order Management:

- User adds items to cart on frontend.
- Order details stored in Sanity CMS when placed.
- Order status is updated based on shipment and payment events.

[Third-Party APIs]

— Shipment Tracking API: (Ship Engine)

- Tracks order progress (e.g., shipped, in transit, delivered).
- Updates fetched are displayed on the frontend order tracking page.

[Payment Gateway]

— Secure Transactions:

- Handles payment processing for various methods.
- Returns transaction status to the frontend.

— Integration with Sanity:

- On successful payment, updates the order status in Sanity CMS.

Detailed API Requirements Documented:

Product Browsing:

User Action:

User views available products.

API Request:

GET /products

API Response:

List of products with details like name, price, stock, image.

Example Response:

```
[  
  { "id": 1, "name": "Sofa Set", "price": 300, "stock": 20, "image": "sofa.jpg" },  
  { "id": 2, "name": "Dining Table", "price": 150, "stock": 15, "image": "dining_table.jpg" }  
]
```

Product Detail View:

User Action:

User clicks on a product to view detailed information.

API Request:

GET /product/{id}

API Response:

Detailed product information (description, price, stock, images).

Example Response:

```
{ "id": 1, "name": "Sofa Set", "description": "Comfortable 5-seater sofa", "price": 300, "stock": 20, "image": "sofa.jpg" }
```

Adding Product to Cart:

User Action:

User adds product to cart.

Action Required:

Client-side logic to manage the cart (not an API call, but may involve local storage).

Order Creation:

User Action:

User proceeds to checkout and places an order.

API Request:

POST /orders

Payload:

```
{  
  "customerInfo": { "name": "John Doe", "address": "123 Street, City", "contact": "123-456-7890" },  
  "products": [{ "productId": 1, "quantity": 2 }, { "productId": 2, "quantity": 1 }],  
  "paymentInfo": { "method": "Credit Card", "status": "Pending" }  
}
```

API Response:

Order ID and status.

```
{ "orderId": 123, "status": "Pending", "message": "Order created successfully" }
```

Order Tracking:

User Action:

User wants to track the status of the order.

API Request:

GET /shipment/{orderId}

API Response:

Shipment tracking info (shipment ID, current status, expected delivery date).

Example Response:

```
{ "shipmentId": "ABC123", "orderId": 123, "status": "In Transit", "expectedDeliveryDate": "2025-01-20" }
```

Sanity Schemas

1. Product Schema

Sanity Schemas

1. Product Schema

```
export default {  
  name: 'product',  
  type: 'document',  
  title: 'Product',  
  fields: [  
    { name: 'name', type: 'string', title: 'Product Name' },  
    { name: 'description', type: 'text', title: 'Description' },  
    { name: 'price', type: 'number', title: 'Price' },  
    { name: 'stock', type: 'number', title: 'Stock Level' },  
    { name: 'image', type: 'image', title: 'Product Image' },  
    {  
      name: 'category',  
      type: 'reference',  
      to: [{ type: 'category' }],  
      title: 'Category',  
    },  
  ],  
};
```


2. Order Schema

```
export default {
  name: 'order',
  type: 'document',
  title: 'Order',
  fields: [
    { name: 'orderId', type: 'string', title: 'Order ID' },
    { name: 'customer', type: 'reference', to: [{ type: 'customer' }], title: 'Customer' },
    {
      name: 'products',
      type: 'array',
      of: [
        {
          type: 'object',
          fields: [
            { name: 'product', type: 'reference', to: [{ type: 'product' }], title: 'Product' },
            { name: 'quantity', type: 'number', title: 'Quantity' },
          ],
        },
      ],
      title: 'Products',
    },
    { name: 'totalAmount', type: 'number', title: 'Total Amount' },
    { name: 'status', type: 'string', title: 'Order Status', options: { list: ['Pending', 'Shipped', 'Delivered'] } },
    { name: 'createdAt', type: 'datetime', title: 'Order Date' },
  ],
};
```

3. Customer Schema

```
export default {  
  name: 'customer',  
  type: 'document',  
  title: 'Customer',  
  fields: [  
    { name: 'name', type: 'string', title: 'Customer Name' },  
    { name: 'email', type: 'string', title: 'Email' },  
    { name: 'contactNumber', type: 'string', title: 'Contact Number' },  
    { name: 'address', type: 'text', title: 'Address' },  
  ],  
};
```

4. Category Schema

```
export default {  
  name: 'category',  
  type: 'document',  
  title: 'Category',  
  fields: [  
    { name: 'title', type: 'string', title: 'Category Title' },  
    { name: 'description', type: 'text', title: 'Description' },  
  ],  
};
```