# Marketplace Technical Foundation - FurniSphere

Our marketplace will provide a platform to purchase high-quality furniture, such as sofas, tables, chairs, beds, and other furniture items, directly from trusted sellers. It aims to make furniture shopping more convenient with a wide variety of options, competitive prices, and seamless delivery services.

## 1. Technical Requirements :

### Frontend Requirements:

- The website layout will be optimized for both mobile and desktop users.
- The website will be user-friendly and attractive with smooth workflow.

**Essential Pages:**

- Home
- Contact
- Shop
- Product detail page
- Cart Page
- Account page
- Checkout
- Blog
- Order Confirmation Page

### Sanity CMS as Backend:

Sanity CMS is used as backend to manage product data, customer details, and order records.
**Detailed schema design for Product, Order and Customer :**

## PRODUCT SCHEMA:

```
export default {

 name: "product",

 type: "document",

 title: "Product",

 fields: [

  {

   name: "product_name",

   type: "string",

   title: "Product Name",

   description: "The name of the product",

  },

  {

   name: "product_description",

   type: "text",

   title: "Product Description",

   description: "A brief description of the product",

  },

  {

   name: "price",

   type: "number",

   title: "Price",
   description: "The price of the product",
  },
  {
   name: "ratings",
   type: "number",
   title: "Ratings",
   description: "The average rating of the product",
   validation: (Rule) => Rule.min(0).max(5),
  },
  {
   name: "reviews",
   type: "number",
   title: "Reviews",
   description: "The number of reviews",
  },
  {
   name: "stock",
   type: "number",
   title: "Stock",
   description: "The stock availability",
  },
  {
   name: "colors_available",
   type: "array",
   title: "Colors Available",
   description: "Hex codes for available colors",
   of: [{ type: "string" }],
  },
  {
   name: "sizes_available",
   type: "array",
   title: "Sizes Available",
   description: "Available sizes",
   of: [{ type: "string" }],
  },
  {
   name: "image",
   type: "image",
   title: "Image",
   description: "Product image",
   options: {
     hotspot: true,
   },
  },
  {
   name: "id",
   type: "string",
   title: "Product ID",
       description: "Unique identifier for the product",
     },
   ],
  };
```

**ORDER SCHEMA:**

```
export default {
  name: "order",
  type: "document",
  title: "Order",
  fields: [
    {
      name: "order_id",
      type: "string",
      title: "Order ID",
      description: "Unique identifier for the order",
    },
    {
      name: "customer_details",
      type: "object",
      title: "Customer Details",
      fields: [
        {
          name: "customer_id",
  type: "string",
  title: "Customer ID",
  description: "Unique identifier for the customer",
},
{
  name: "name",
  type: "string",
  title: "Customer Name",
  description: "Name of the customer",
},
{
  name: "address",
  type: "string",
  title: "Address",
  description: "Address of the customer",
},
{
  name: "phone",
  type: "string",
  title: "Phone",
  description: "Phone number of the customer",
},
{
  name: "email",
      type: "string",

      title: "Email",

      description: "Email address of the customer",

    },

  ],

},

{

  name: "products",

  type: "array",

  title: "Products",

  description: "List of products in the order",

  of: [

    {

      type: "object",

      fields: [

        {

          name: "product_id",

          type: "string",

          title: "Product ID",

          description: "Unique identifier for the product",

        },

        {

          name: "name",

          type: "string",
          title: "Product Name",
          description: "Name of the product",
        },
        {
          name: "quantity",
          type: "number",
          title: "Quantity",
          description: "Quantity of the product",
        },
        {
          name: "price",
          type: "number",
          title: "Price",
          description: "Price of the product",
        },
      ],
    },
  ],
},
{
  name: "order_price",
  type: "number",
  title: "Order Price",
  description: "Total price of the order",
    },
    {
      name: "status",
      type: "string",
      title: "Status",
      description: "Current status of the order",
      options: {
        list: [
          { title: "Pending", value: "pending" },
          { title: "Processing", value: "processing" },
          { title: "Shipped", value: "shipped" },
          { title: "Delivered", value: "delivered" },
          { title: "Cancelled", value: "cancelled" },
        ],
      },
    },
    {
      name: "date",
      type: "datetime",
      title: "Order Date",
      description: "Date when the order was placed",
    },
  ],
};
```

**CUSTOMER SCHEMA:**

```
export default {
  name: "customer",
  type: "document",
  title: "Customer",
  fields: [
    {
      name: "customer_id",
      type: "string",
      title: "Customer ID",
      description: "Unique identifier for the customer",
    },
    {
      name: "name",
      type: "string",
      title: "Customer Name",
      description: "Full name of the customer",
    },
    {
      name: "email",
      type: "string",
      title: "Email Address",
      description: "Email address of the customer",
    validation: (Rule) =>
      Rule.regex(
        /^[^\s@]+@[^\s@]+\.[^\s@]+$/,
        { name: "email" }
      ).error("Please enter a valid email address"),
  },
  {
    name: "phone",
    type: "string",
    title: "Phone Number",
    description: "Contact phone number of the customer",
    validation: (Rule) =>
      Rule.regex(
        /^[0-9]{10,15}$/,
        { name: "phone" }
      ).error("Please enter a valid phone number"),
  },
  {
    name: "address",
    type: "string",
    title: "Home Address",
    description: "Home address of the customer",
  },
],
};
```

## Third-Party APIs:

**1. Shipment APIs:**

Integrate shipment APIs like **ShipEngine** or **Shipoo** to manage real-time shipment tracking, delivery status, and estimated delivery times without building custom logistics systems.

**2. Payment Gateways:**

Integrate payment gatewas to securely process transactions and provide users with trusted and convenient payment options like **Easypaisa**, **JazzCash**, or **Stripe**.

**3. User Authentication:**

APIs like **Clerk** to quickly implement secure user login, sign-up, and profile management features, ensuring a smooth and reliable authentication experience.

## 2. Design System Architecture :

- ### Detailed Workflow Explanation:

**1. User Enters Marketplace**

- A visitor navigates to the marketplace platform.

**2. Login/Signup**

- If not already logged in, the user is prompted to login or sign up.
- Credentials are validated via a third-party authentication service such as Clerk.

**3. Third-Party API Call (Clerk)**

- The marketplace makes an API request to Clerk to authenticate the user.
- On success, Clerk returns user data, confirming authentication.

**4. User Enters Home Page**

- The authenticated user is redirected to the home page.
- Product data is dynamically fetched from Sanity CMS and displayed.

**5. Browsing Products**

- The user browses through the list of products fetched from Sanity CMS.

**6. Product Detail Page**

- When a product is clicked, the user is navigated to a dynamic product detail page.
- Product details are fetched from Sanity CMS via an API request.

**7. Add to Cart**

- The user adds a product to the cart.
- The cart's state is updated on the frontend.

**8. Place Order**

- The user places an order.
- Order details (including user ID, products, prices, etc.) are sent to Sanity CMS via an API request and stored.

**9. Checkout and Payment**

- The user is directed to the checkout page to confirm their order.
- Payment is processed via a payment gateway (e.g., Stripe, PayPal, Easypaisa, JazzCash).
- The selected payment method triggers the respective API for payment processing.

**10. Order Confirmation**

**=> After successful payment:**

- The order is confirmed.
- Confirmation details are sent to Sanity CMS.
- The user receives a success message.

**11. Track Shipment**

- The user clicks on "Track Shipment."
- A third-party shipment API is called to fetch real-time tracking information.

**12. Create Label**

- The user clicks "Create Label."
- The system makes an API call to retrieve order details and generate a shipping label.
- The label and relevant details are displayed to the user.

**13. Completion**

- The process concludes with the user receiving shipment tracking information and shipping labels.

- ## System Architecture:

```
┌──────────────────────────┐
│ User Enters Marketplace │
└────────────┬─────────────┘
     ▼
 ┌─────────────────┐
 │ Login / Signup │
 └────────┬────────┘
     ▼
┌──────────────────────────┐
│ Clerk API Authentication │
└────────────┬─────────────┘
     ▼
  ┌───────────────┐
```

```
│ User Home Page │
└─────────┬─────────┘
        ▼
┌──────────────────────────┐
│ Browse Products (CMS) │
└────────────┬─────────────┘
        ▼
┌───────────────────────────────┐
│ Product Details Page Open │
└──────────────┬────────────────┘
        ▼
┌──────────────────┐
│ Add to Cart    │
└─────────┬────────┘
        ▼
┌──────────────────────────────────┐
│ Place Order (API to CMS)    │
└──────────────┬───────────────────┘
        ▼
┌──────────────────┐
│ Checkout Page │
└─────────┬────────┘
        ▼
┌────────────────────────┐
```

```
 │ Payment Gateway API │
 └─────────┬──────────────┘
           ▼
┌──────────────────────────────────┐
│ Order Confirmation (CMS) │
└──────────────┬───────────────────┘
            ▼
┌─────────────────────────────┐
│ Track Shipment (API) │
└─────────────┬───────────────┘
            ▼
┌──────────────────────────┐
│ Create Label (API) │      - - - - -> SUCESSFULLY PLACED ORDER!
└──────────────────────────┘
```

## 3. Plan API Requirements:

### General E-Commerce

**Endpoint**: /products
**Method**: GET
**Purpose**: Fetch all products
**Response**: [

{

"product_name": "Trenton modular sofa_3",

"product_description": "A modern, versatile modular sofa designed for comfort and style. Its clean lines and customizable layout make it perfect for any living space.",

```
        "price": 25000,

        "ratings": 4.5,

        "reviews": 7,

        "stock": 8,

        "colors_available": { "#6C757D", "#F1F3F5", "#2A2C2B" },

        "sizes_available": { "L",  "XL","XS" },

        "image":
"https://res.cloudinary.com/dqc4xmj4g/image/upload/v1736284448/Trenton_modular_sofa_3_1_s5q1vn.png",

        "id": "1"

      },

]
```

**Endpoint**:  /orders
**Method**:  POST
**Purpose**: Create new order

**Payload:**

```
{

  "customer_info": {

    "customer_id": "CUST123",

    "name": "John Doe",

    "email": "john.doe@example.com",

    "phone": "1234567890",

    "address": "123 Main St, New York, NY 10001"

  },

  "product_details": [

    {

      "product_id": "PROD001",
      "name": "Product A",

      "quantity": 2,

      "price_per_unit": 50

    },

    {

      "product_id": "PROD002",

      "name": "Product B",

      "quantity": 1,

      "price_per_unit": 100

    }

  ],

  "payment_status": {

    "status": "Paid",

    "payment_method": "Credit Card",

    "transaction_id": "TXN987654321",

    "amount_paid": 200

  }

}
```

**Response**: {

```
    "sale_id": "sale_id 17",

    "quantity_sold": 51,

    "sale_price": 89,

    "date_of_sale": "date_of_sale 17",

    "customer_id": "customer_id 17",

    "id": "17"

}
```

**Endpoint**:  /customer
**Method**:  GET
**Purpose**: Fetching a Specific Customer's Information
**Response:**  {

```
  "id": 2,

  "name": "Jane Smith",

  "email": "jane.smith@example.com",

  "phone": "9876543210",

  "address": "456 Elm St, Los Angeles, CA 90001"

 }
```

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -