FAME

FASHION, ATTRACTION, MODE, EXPORT

Technical Foundation

Technical Plan:

a. Frontend Requirements:

1. User friendly environment for browsing
2. Appealing and attractive pictures
3. Responsive design for Mobile users
4. Essential Pages: Home, Product listing, Product Details, Cart, Checkout, Order confirmation, Order Tracking, Dashboard

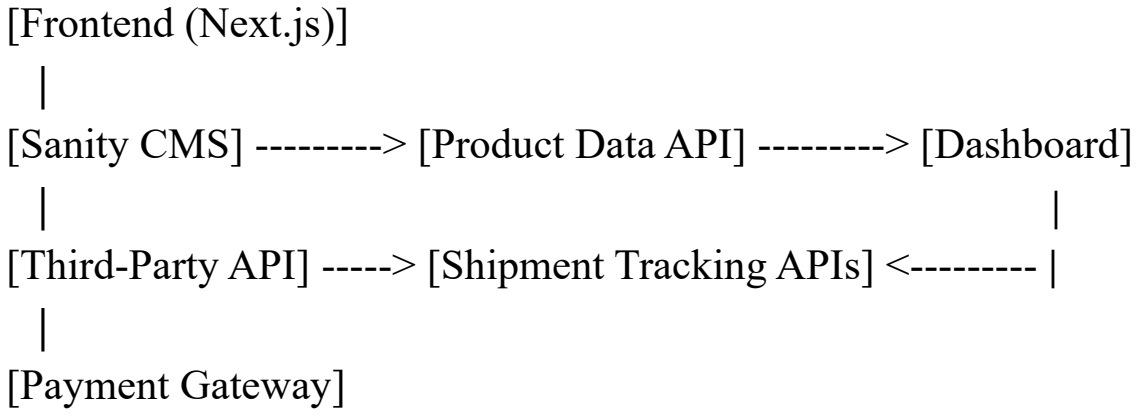
b. Sanity CMS For Backend:

1. Use Sanity CMS to manage product data, customer details, and order records, tracking and shipping. Sanity acts as the database for my marketplace.

c. Third Party APIs (Integration):

Integration of shipment APIs, tracking APIs, and other backend services.

SYSTEM ARCHITECHTURE:



APIs Requirements:

Endpoint Name: /products

1. Method: GET
2. Description: Fetch all available products from Sanity.
3. Response: Product details (ID, name, price, stock, image).

Endpoint Name: /orders

1. Method: POST
2. Description: Create a new order in Sanity.
3. Payload: Customer info, product details, payment status.

Endpoint Name: /shipment

1. Method: GET
2. Description: Track order status via third-party API.
3. Response: Shipment ID, order ID, status, expected delivery date.

Endpoint Name: /product listing

1. Method: POST
2. Description: Create a new product in sanity
3. Payload: Seller’s information, product Details, payment Methods

Sanity Schemas:

Order Schema:

```
export default {
    name: "order",
    type: "document",
    title: "Order",
    fields: [
      { name: "firstName", type: "string", title: "First Name" },
      { name: "lastName", type: "string", title: "Last Name" },
      { name: "city", type: "string", title: "City" },
      { name: "country", type: "string", title: "Country" },
      { name: "state", type: "string", title: "State / Province" },
      { name: "postalCode", type: "string", title: "Postal Code" },
      { name: "address1", type: "string", title: "Address Line 1" },
      { name: "address2", type: "string", title: "Address Line 2" },
      { name: "email", type: "string", title: "Email" },
      { name: "phoneNumber", type: "string", title: "Phone Number" },
      { name: "message", type: "text", title: "Message" },

      // Order Details
      {
        name: "orderItems",
        type: "array",
        title: "Order Items",
        of: [
          {
            type: "object",
            fields: [
              { name: "productName", type: "string", title: "Product Name" },
              {name: "productImage", type:"image" ,title: "Product Image"  ,
options: {
                hotspot: true
              },},
              { name: "quantity", type: "number", title: "Quantity" },
              { name: "price", type: "number", title: "Price" },
            ],
          },
        ],
      },

      { name: "totalAmount", type: "number", title: "Total Amount" },
      { name: "orderDate", type: "datetime", title: "Order Date", options: {
default: new Date().toISOString() } },],};
```

Product Schema:

```
import { defineType } from "sanity"

export const product = defineType({
    name: "product",
    title: "Product",
    type: "document",
    fields: [
        {
            name: "title",
            title: "Title",
            validation: (rule) => rule.required(),
            type: "string"
        },
        {
            title: 'Slug',
            name: 'slug',
            type: 'slug',
            options: {
              source: 'title',
              maxLength: 200, // will be ignored if slugify is set
              slugify: input => input
                                  .toLowerCase()
                                  .replace(/\s+/g, '-')
                                  .slice(0, 200)
            }
          },
        {
            name:"description",
            type:"text",
            validation: (rule) => rule.required(),
            title:"Description",
        },
        {
            name: "productImage",
            type: "image",
            validation: (rule) => rule.required(),
            title: "Product Image"
        },
        {
            name: "price",
            type: "number",
            validation: (rule) => rule.required(),
            title: "Price",
```

```
        },
        {
            name: "tags",
            type: "array",
            title: "Tags",
            of: [{ type: "string" }]
        },
        {
            name:"dicountPercentage",
            type:"number",
            title:"Discount Percentage",
        },
        {
            name:"isNew",
            type:"boolean",
            title:"New Badge",
        }
    ]
})
```