DAY 2

General E-commerce

DAY2_PLANNING_THE_TECHNICAL_FOUNDATION

PREPARED BY:
MUHAMMD HAMMAD

Day 2 Planning Technical Foundation

Recap Day 1

Primary Purpose:

The primary purpose of our business is to sell handmade
jewellery to local customers as well as international customers.

Business Goals:

We offer customizing jewellery on some of items, so
peoples will buy according to their needs that is plus point for them.

Target Audience:

Our target audience are generally females with young age girls

Marketplace Features:

- Realtime order tracking
- Flexible payment methods

Day 2

1. Technical Requirements

This document outlines the technical planning of each feature

1. Essential Pages:

Home Page: CTA , Hero Section, Featured Products, Top Categories, and some products

Products Page: Each and every product of our store list with filters like availability, rating, category, price, Sorting with best selling product

Product Detail Page: Product title, Image, Discription, Price, Stock Availability, Discounts, Color, Size, Rating, Reviews, Add to cart, Add to wishlist, FAQ

Cart Page: Product title, image, size, color, quantity, delete, add to wishlist, update quantity

Checkout Page: Order Summary, Payment details, SHIPPING zone

Order Confirmation Page: Order details, tracking, estimated delivery

About Page: Something about our store, Our popular products, key features

Contact Page: Contact info

2. Sanity CMS as Backend:

Sanity act as database for our customer, products, and order records

Product Schema:

Product ID

Name

Description

Price

Discount (if Applicable)

Badge (e.g New, Sales)

Inventory

Color

Size

FAQ

Product Schema:

Customer ID

Full Name

Email

Phone No.

Address

Order Schema:

Order ID

Customer ID

Product ID

Order Detail

Total Amount

Payment Schema:

Payment ID

Order ID

Amount Paid, Payment Method (e.g Credit Card, Wallet)

Payment Status (e.g Pending, Successfully)

Shipment Schema:

Shipment ID

Order ID

Courier Service

Estimated Delivery Date, Shipment Status

3. Third Party API Integration:

Payment Gateways:

Stripe: Features: Secure Payments, support multiple payment methods, and realtime transaction updates
Integration: Stripe SDKs and APIs for integration

Paypal: Features: Widely accepted payments
Integration: Paypal Rest API

Shipment Tracking:

Ship-engine: Features: Real-time tracking
Integration: Efficient shipment tracking

Additional APIs:

Google Maps API: Google maps API enables location-based service

Notification API: Notification API enables web push alerts

Frontend: NextJS and TailwindCSS

Backend: Sanity CMS

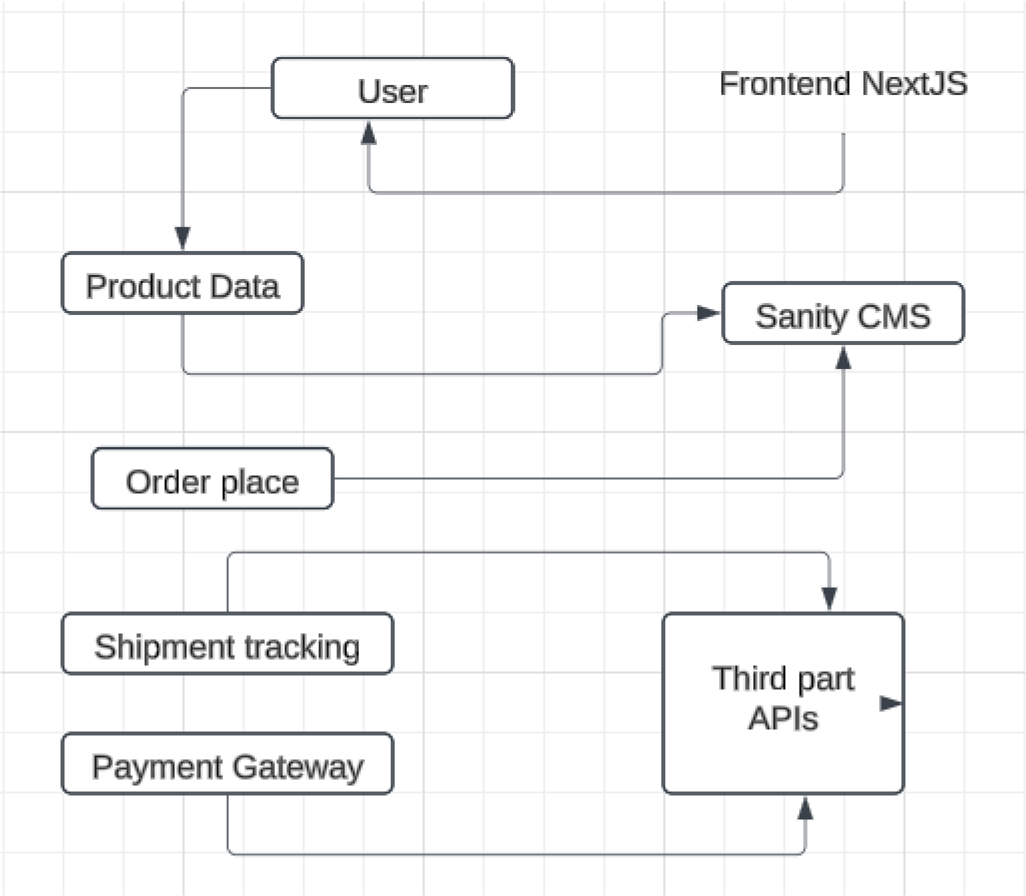
APIs: Shipment API and Payment API

Testing: Test responsiveness, security, and functionality

Deployment: Deployment on vercel or MS Azure

2. Design System Architecture

How components visualize and interact



- A user visits marketplace frontend to browse products.

- Frontend make a request to product API to fetch product listing and displayed dynamically.

- When the user places an order the order details are Sanity CMS via API request, where the order is recorded.

- Shipment tracking information is fetch through a Third-Part API and displayed the user in real-time.

- Payment details are securely processed through the Payment Gateway and the confirmation is sent back to the user and record in Sanity CMS.

3. Plan API Requirements

API endpoints are:

Endpoint Name	Method	Description	Request Body	Response Example
/api/users/register	POST	Register a new user	{ "username": "Hammad", "email": "hammad@gmail.com", "password": "password"}	{"status": "success", "message": "User registered successfully" }
/api/users/login	POST	Authenticate a user and generates a new user	{"email": "hammad@gmail.com", "password": "password"}	{"status": "success", "token": "jwt.token.here"}
/api/users/{id}	PUT	Update user	{"email": "update@gmail.com", "password": "password"}	{"status": "success", "message": "profile updated successfully"}
/api/users/categories	GET	List of all product categories	-	{"status": "success", "data": [{"id": 1, "name":"e.g jwellery"}]}
/api/users/categories/{id}/product	GET	Fetch product	-	{"status": "success", "data": [{"id": 10, "name":"e.g earrings"}]}
/api/users/products/{id}	GET	Detail of specific product	-	{"status": "success", "data": {"id": 10, "name":"e.g earrings"}}
/api/cart	GET	Current user cart	-	{"status": "success", "data": {"items": [{"id" : 10}]}}
/api/cart/add	POST	Add a product to cart	{"product_id": 1, "quantity": 2}	{"status": "success", "message": "item add successfully"}
/api/checkout	POST	Process payments	{"payment_method": "cart, "shipping_address": '---'}	{"status": "success", "message": "order place successfully"}
/api/orders/{id}	GET	Detail of specific order	-	{"status": "success", "data": { "order_id": 1234567890}}
/api/home	GET	Homepage include feature products	-	{"status": "success", "data": { "feature_products": "---"}}

4. Write technical documentation

1. System Architecture Overview

How components visualize and interact

1.1 Frontend

- NextJS for UI
- Tailwind for Responsive Design

1.2 Backend

- Sanity.io as backend
- JSON web tokens for secure authentication

1.3 Deployment

- Vercel

2. Key Workflows

2.1 Authentication and Authorization

1. **Signup:** User can registered by providing credentials

2. **Login:** User enter credentials to obtain JWT token

3. **Password Recovery:** User can reset password by token

2.2 Add Product to Cart

User can add product to cart

2.3 Checkout

User checkout by providing address and other details

2.4 Payment

User provide payment details or COD

3. Category Specific Instructions

- Hair/Hijab Accessories
- Jwellery

4. API Endpoints

Authentication

1./api/users/register – Register a new user.
2./api/users/login – Authenticate user and return a JWT token.

Categories

1./api/categories – Retrieve all categories.
2./api/categories/{id}/products – Retrieve products from a specific category.

Products

1./api/products/{id} – Retrieve detailed information about a product.

Cart

1./api/cart/add – Add a product to the cart.
2./api/cart – Retrieve cart details.

Orders

1./api/checkout – Process payment and create an order.

5. Sanity Schema Example

schemaTypes/product.ts

```
export const productSchema = defineType({
  name: "products",
  title: "Products",
  type: "document",
  fields: [
    {
      name: "title",
      title: "Product Title",
      type: "string",
    },
    {
      name: "price",
      title: "Price",
      type: "number",
    },
    {
      title: "Price without Discount",
      name: "priceWithoutDiscount",
      type: "number",
    },
    {
      name: "badge",
      title: "Badge",
      type: "string",
    },
    {
      name: "image",
      title: "Product Image",
      type: "image",
    },
{
      name: "category",
      title: "Category",
      type: "reference",
      to: [{ type: "categories" }],
    },
    {
      name: "description",
      title: "Product Description",
      type: "text",
    },
    {
      name: "inventory",
      title: "Inventory Management",
      type: "number",
    },
    {
      name: "tags",
      title: "Tags",
      type: "array",
      of: [{ type: "string" }],
      options: {
        list: [
          { title: "Featured", value: "featured" },
          {
            title: "Follow products and discounts on
Instagram",
            value: "instagram",
          },
          { title: "Gallery", value: "gallery" },
        ],
      },
    },
  ],
});
```