DAY 2 PLANNING THE TECHNICAL FOUNDATION

Step 1: **Define Technical Requirements**

Next.js: A React-based framework for building server-rendered and static web applications. It ensures fast performance, SEO optimization, and flexibility with API routes.

Sanity CMS: A headless CMS that serves as the backend for managing product data, customer information, and order records. Its real-time collaboration and flexible schema design make it ideal for e-commerce.

Tailwind CSS: A utility-first CSS framework for creating responsive and customizable designs quickly, ensuring a modern and consistent user interface across devices.

Third-Party Shipment API: Enables real-time shipment tracking and order delivery updates, enhancing the post-purchase user experience.

Vercel: A cloud platform for deploying and hosting web applications, ensuring quick deployment and optimized performance for Next.js apps.

Step 2: **Design System Architecture**

1. User Registration

- Start: User lands on the registration page.
- Input: User fills in registration details (name, email, password, etc.).
- Process:
    1. Validate user input on the frontend.
    2. Send data to the Sanity CMS using a POST API.
- Output: User account is created in the Sanity CMS.

2. Product Browsing

- Start: User navigates to the product listing page.
- Process:
    1. Frontend sends a GET request to the Sanity CMS to fetch all products.
    2. Sanity CMS responds with product details (name, image, price, etc.).
- Output: Products are displayed to the user.

3. Add to Cart

- Start: User selects a product and clicks "Add to Cart."
- Process:
    1. Product ID and quantity are stored in the frontend through context API integration.
    2. The cart is updated dynamically.
- Output: Product appears in the cart.

4. Checkout and Place Order

Start:

- User clicks **"Checkout"** on the cart page.

Process:

1. **Enter Payment and Shipping Details:**
    - User provides a shipping address, payment method, and other necessary details on the checkout page.

2. **Send Order Payload:**
    - The frontend sends a **POST request** to the Sanity CMS with the following payload:
        - Order details (product IDs, quantities, and prices).
        - Customer information (name, phone number, address, etc.).
        - Payment method.

3. **Save Order:**
    - Sanity CMS processes the request and stores the order details in the database.
    - A unique **Order ID** is generated by the backend.

Output:

1. **Order Confirmation Screen:**

    - After receiving a successful response from the CMS, the frontend displays an **Order Confirmation Screen**:
        - Thank you message: "Thank you for your purchase!"
        - **Order ID**: Display the unique order ID for tracking purposes.
        - Summary of the order (product names, quantities, and total price).

Design System Architecture

User Registration:

Start -> Registration Form -> API Call (POST /register) -> Save in Sanity CMS -> End.

Product Browsing:

Start -> Product Page -> API Call (GET /product) -> Display Product List -> End.

Add to Cart:

Start -> Select Product -> Update Cart State -> Display Cart -> End.

Checkout Page:

User clicks "Checkout" -> Enters details -> Sends POST request -> Sanity CMS saves the order.

Order Confirmation Screen:

After successful order processing:

Show a confirmation message with details.

Step 3: **Plan API Requirements**

General eCommerce Example:

List of endpoints:
No.1:
Method:POST
Endpoint: “/signup”
Purpose: Creates user account and user created in sanity

Payload: {
name:”username”,
email:”user email”,
Password:”user password”,
}

No.2:
Method: GET
Endpoint: “/product”
Purpose: Fetch all data from Sanity
Response: [
{_id:”2214”,
name:”product name”,
description:”product description”,
Stock:40,
image:”image url”, …},
{...},
{...},...
]

No.3:
Method: POST
Endpoint: “/order”
Purpose: Send order details to the sanity
Payload: {
order_Id:939,
order_Status: “Pending”,
customerInfo:{name:”any name”, phone:12345678901,
address:”home address”, …},
productDetail:{quantity:3, price:390},
Time_stamp: new Date(),

paymentMethod:”COD”
 }

No.4:
Method: GET
Endpoint: “/shipment”
Purpose: Track order through third party API
Response:{
shipment_Id:”239889sh”,
order_Id:939,
status:”In Transit”,
DeliveryDate:”new Date()”}

GET `/product`:
Fetches all product data from Sanity CMS, including details like product ID, name, description, stock availability, and image URLs.
POST `/order`:
Submits order details to Sanity CMS. The payload includes order information such as order ID, customer details, product quantity, price, timestamp, and payment method.
GET `/shipment`:
Retrieves shipment tracking information from a third-party API. The response includes shipment ID, associated order ID, current status, and estimated delivery date.