Day 2: Planning the Technical Foundation - Ali's Store

1. Define Technical Requirements

- **Frontend:**
    - **Build a user-friendly interface using Next.js.**
    - **Create essential pages: Home, Product Listing, Product Details, Cart, Checkout, and Order Confirmation.**
- **Backend:**
    - **Use Sanity CMS to manage product data, customer details, and order records.**
    - **Design Sanity schemas for products, orders, and customers.**
- **Third-Party APIs:**
    - **Integrate a Mock API for testing shipment tracking and payment processing.**

What I Have Done:

- **Defined the frontend structure and backend requirements.**
- **Drafted Sanity schemas for products and orders.**

2. Design System Architecture

- Create a high-level system architecture diagram showing:
    1. **Frontend (Next.js)**: Handles user interactions.
    2. **Sanity CMS**: Manages product and order data.

3. **Mock API**: Simulates shipment tracking and payment processing.
- Define key workflows:
    1. **Product Browsing**: Fetch product data from Sanity and display it on the frontend.
    2. **Order Placement**: Save order details in Sanity and process payments via a Mock API.
    3. **Shipment Tracking**: Fetch real-time updates from the Mock API and display them to the user.

What I Have Done:

- Drafted the system architecture diagram.
- Outlined workflows for product browsing, order placement, and shipment tracking.

---

3. Plan API Requirements

- Define API endpoints for:
    - **Fetching Products**: `/products` (GET) to retrieve product details.
    - **Creating Orders**: `/orders` (POST) to save order details.
    - **Tracking Shipments**: `/shipment` (GET) to fetch shipment status.
- Document the API requirements, including methods, payloads, and responses.

What I Have Done:

- Listed API endpoints and their purposes.
- Drafted example responses for each endpoint.

4. Write Technical Documentation

- Write a structured technical document including:
    - **System Architecture Overview**: Diagram and description of components.
    - **Key Workflows**: Step-by-step details of user interactions.
    - **API Endpoints**: Table with methods, purposes, and example responses.
    - **Sanity Schemas**: Example schema for products.
- Save the document in a **Documentation** folder in my repository.

What I Have Done:

- Drafted the technical document with system architecture, workflows, and API details.
- Created a Sanity schema for products.

5. Collaborate and Refine

- Share my technical plan with peers and mentors for feedback.
- Refine the system architecture, API requirements, and documentation based on feedback.
- Use **GitHub** to track changes and collaborate on diagrams or drafts.

What I Have Done:

- Shared my draft with peers for initial feedback.
- Incorporated suggestions to improve the technical plan.

Key Outcome of Day 2

What I Have Achieved:

1. **Technical Plan**:
    - Aligned with the business goals of **Ali's Store**.
2. **System Architecture**:
    - Designed a clear diagram showing interactions between **Next.js**, **Sanity CMS**, and **Mock API**.
3. **API Requirements**:
    - Documented endpoints for fetching products, creating orders, and tracking shipments.
4. **Sanity Schemas**:
    - Drafted schemas for products and orders.
5. **Collaborative Feedback**:
    - Refined the technical plan based on input from peers and mentors.

Next Steps

- On **Day 3**, I will use the provided API or create my own schema in **Sanity CMS** for **GET**, **POST**, **UPDATE**, **PATCH**, and **DELETE** operations.
- Focus on implementing workflows for product browsing, order placement, and shipment tracking.