**🔹 Key Achievements of Day 2:**

**1-Defining Technical Requirements.**

The foundation of any great product lies in a well-defined technical structure. Today, I outlined:

🔹 **Frontend Requirements:**

* Built with **Next.js** for a seamless and fast UI experience.
* A fully **responsive design** to ensure compatibility across devices.
* Core pages like **Homepage, Product Listing, Product Details, Cart, Checkout, and Order Confirmation.**

🔹 **Backend & Database (Sanity CMS):**

* **Sanity CMS** serves as the **backend database**, handling **product data, customer information, and orders** efficiently.
* Defined **schemas** in Sanity to align with the business goals.
* Structured content management for easy updates and dynamic product handling.

🔹 **Third-Party API Integrations:**

* **Payment Gateway Integration (Stripe, PayPal, etc.)** for smooth transaction processing.
* **Shipment Tracking API** to provide real-time order tracking for customers.

**2- Designing the System Architecture .**

A well-planned **system architecture** is the backbone of any application. Here’s how the components interact:

📌 **Frontend (Next.js)** – User-facing application handling product browsing, cart management, and checkout.  
📌 **Sanity CMS** – Serves as the **database** for product and order management.  
📌 **Third-Party APIs** – Handles external services like **shipment tracking and payments**.  
📌 **Data Flow:**  
1️⃣ User browses the marketplace on **Next.js frontend**.  
2️The frontend fetches product data from **Sanity CMS** via API requests.  
3️ When an order is placed, the details are stored in **Sanity CMS**.  
4️ **Shipment tracking API** fetches real-time order status updates.  
5️ **Payment gateway API** processes secure transactions.  
6️ The system updates and syncs data between **frontend, backend, and external APIs**.

A strong architecture ensures **scalability, efficiency, and seamless user experience!**

**3- Planning API Requirements**

A well-structured API is critical for smooth communication between different components. I outlined key Restful APIs:

📌 **GET /products** → Fetch all available products from **Sanity CMS**.  
📌 **POST /orders** → Create a new order with customer details, product data, and payment status.  
📌 **GET /shipment** → Retrieve order tracking details from the **shipment tracking API**.

Each API is designed to optimize **performance, data accuracy, and security**, ensuring a seamless shopping experience.

**4-Designing Key Workflows.**

To ensure smooth platform operations, I planned critical workflows:

✅ **User Registration:** User signs up → Data is stored in **Sanity CMS** → Confirmation email sent.  
✅ **Product Browsing:** User views products → Sanity API fetches data → Products displayed dynamically.  
✅ **Order Placement:** User adds items to cart → Proceeds to checkout → Order details stored in **Sanity CMS**.  
✅ **Shipment Tracking:** Order status updates fetched from **third-party API** → Displayed to the user.

Each step is carefully planned to **enhance user experience and system efficiency!**

**📍 Key Takeaways from Today:**

✅ Clear understanding of **how frontend, backend, and APIs communicate**.  
✅ Well-defined **system architecture** to ensure smooth data flow.  
✅ Properly planned **API structure and key workflows**.  
✅ The importance of **scalability, security, and efficiency** in technical planning.