Day 2:

Planning The Technical Foundation

1. Define Technical Requirements

The first step is to translate the business goals you defined on Day 1 into clear technical requirements. This ensures that your technical infrastructure supports the business objectives of your marketplace.

Frontend Requirements

- **User-friendly Interface:** The marketplace should be intuitive, with easy navigation for users. Features like search bars, category filters, and quick access to key pages (like Home, Product Listing, Cart, Checkout, etc.) should be prioritized.
- **Responsive Design:** This means your website should work on different screen sizes, such as mobile phones, tablets, and desktops. This will require using responsive design techniques such as CSS media queries and fluid layouts.
- **Essential Pages:**
  - Home Page: Displays an overview of the marketplace, such as featured products, categories, and promotions.
  - Product Listing Page: Displays products based on categories or search filters.
  - Product Details Page: Shows detailed information for each product, including images, description, price, and availability.
  - Cart: A page showing all the products added to the cart, allowing the user to modify quantities, remove products, or proceed to checkout.
  - Checkout Page: Where users provide payment and shipping details to complete their order.
  - Order Confirmation: A page that confirms the order has been placed successfully.

Sanity CMS as Backend

- **Sanity CMS:** Sanity is a headless content management system that allows you to manage content (like product data, customer details, and orders). It acts as your backend, storing all of this data and making it available via an API to be fetched by your frontend.

  - **Schemas:** You'll need to define the types of data you will be working with (e.g., products, orders, users) in Sanity CMS. Each schema defines how the data should be structured.
  - **Example Schema:** For a product schema, you might define fields like name, price, description, image, and stock.

Third-Party APIs

- Integrating external APIs will provide additional functionality, such as:
  - Payment Gateway APIs: To handle payment transactions securely (e.g., Stripe, PayPal).
  - Shipment Tracking APIs: To provide real-time tracking of shipments (e.g., FedEx, UPS).
  - Other APIs: Depending on the business needs, you might integrate other APIs, such as for reviews, analytics, or inventory updates.

2. Design System Architecture

System architecture is a high-level diagram or design that shows how the different components of your marketplace interact with each other. This design helps ensure smooth data flow and efficient operations.

System Architecture Overview

Here's how the different components will interact:

- **Frontend (Next.js):** The frontend is built with Next.js, a React-based framework. It will handle displaying data fetched from Sanity CMS and interacting with third-party APIs.
- **Sanity CMS:** Acts as your database, storing products, orders, and customer details. Your frontend will make API calls to Sanity to retrieve product data or submit order information.
- **Third-Party APIs:** These will be used for functionalities outside of the marketplace, such as payment processing and shipment tracking.
- **Payment Gateway:** A payment service (such as Stripe) that processes payments securely. The frontend will send payment details, and the payment provider will return a response, which is then recorded in the system.

Workflow

- A user visits the marketplace frontend and browses products.

- The frontend makes a request to the Product Data API (powered by Sanity CMS) to fetch the product listings and details, which are displayed on the site.
- The user places an order, and the order details are sent to Sanity CMS via an API request, where the order is recorded.
- Shipment tracking information is fetched through a third-party API and displayed to the user in real-time.
- Payment details are securely processed through a payment gateway, and a confirmation is sent back to the user and recorded in Sanity CMS.

3. Plan API Requirements

Defining the API endpoints is essential for your system's functionality. These endpoints will allow different components of the system (frontend, CMS, third-party services) to communicate with each other.

Example Endpoints

- /products: A GET request to this endpoint will fetch all product details (e.g., name, price, stock).
- /orders: A POST request to this endpoint will create a new order in Sanity, including customer information, product details, and payment status.
- /shipment: A GET request to this endpoint will fetch the status of an order from a third-party shipment tracking API.

Example API Request/Response

- ***/express-delivery-status:***
  - Method: GET
  - Description: Fetch real-time delivery updates for perishable items.
  - Response:

```json
{
  "orderId": 123,
  "status": "In Transit",
  "ETA": "15 mins"
}
```

- **/rental-duration:**
  - Method: POST
  - Description: Add rental details for a specific product.
  - Payload

```json
{
  "productId": 456,
  "duration": "7 days",
  "deposit": 500
}
```

  - Response:

```json
{
  "confirmationId": 789,
  "status": "Success"
}
```

4. Write Technical Documentation

Documenting the system architecture, API requirements, workflows, and schemas is crucial for ensuring clarity and smooth implementation. Professional documentation serves as a reference for all team members and stakeholders.

Types of Documents

- **System Architecture Document:** This document describes the overall design, with diagrams and explanations of how the frontend, backend, and third-party APIs interact.
- **API Specification Document:** This document details all the API endpoints, including their methods, parameters, and response examples.
- **Workflow Diagram:** A visual representation of how users will interact with the marketplace and how data will flow between components.
- **Data Schema Design:** This document defines the entities (e.g., products, orders) and their relationships within your CMS (Sanity).
- **Technical Roadmap:** This outlines the steps, milestones, and deliverables needed to complete the project, including deadlines for each phase.

5. Collaborate and Refine

Collaboration is key to building a successful project. Working with peers, sharing ideas, and gathering feedback will ensure that your technical foundation is solid and your marketplace is well-constructed.

Collaboration Steps:

1. **Group Discussions:** Organize sessions with your peers to brainstorm ideas, solve challenges, and refine your system architecture and API design.
2. **Peer Review:** Share your technical plan with teammates and mentors for feedback. Review each other's work to improve and identify potential issues.
3. **Version Control:** Use GitHub or similar tools to track changes and work together on drafts.
4. **Divide and Conquer:** Split tasks, such as documenting different parts of the system architecture or API design, while keeping in mind that the final submission must be individual.

Conclusion:

In conclusion, planning the technical foundation of your marketplace ensures a structured approach to development, aligning system architecture, API design, and documentation with business goals. Effective collaboration and clear documentation will guide successful implementation and scalability.