# DAY 2 PLANNING THE TECHNICAL FOUNDATION

Prepared by Ameen Alam

First published: PKT 11:08 AM, January 15, 2025

Document Revision Information

| Version | Date | Amendment | Author |
|---|---|---|---|
| 1.0 | 11:08 AM, January 15, 2025 | Initial release of Day1 | Ameen Alam |
| 1.1 | 03:00 PM, January 15, 2025 | Added Examples in Day1 | Ameen Alam |
| 2.0 | 01:00 PM, January 16, 2025 | Day 2 Release | Ameen Alam |

## Table of Contents

# Hackathon Day 2: Planning the Technical Foundation

## Day 2 Goal

The primary goal of Day 2 is to transition from the business-oriented planning of Day 1 to the technical preparation required to build your marketplace. Today, you will create a high-level technical plan, including system architecture, workflows, and API requirements, which will act as a blueprint for the implementation phase. This planning stage is directly connected to the business goals defined on Day 1, ensuring that your technical solutions align with the marketplace's purpose and provide a strong foundation for success. This step is essential for frontend developers as it ensures alignment with business goals while leveraging tools like Sanity CMS and third-party APIs to simplify backend requirements and focus on delivering a scalable and effective solution.

This structured approach mirrors industry best practices and ensures you are prepared to launch a functional marketplace within the hackathon timeline.

## Recap of Day 1: Business Focus

On Day 1, we emphasized the importance of focusing on business requirements before jumping into technical implementation. Here's what we achieved:

1. **Business Goals Defined:**
    - You identified the problem your marketplace aims to solve.
    - You defined your target audience and the unique value proposition (UVP) of your marketplace.
2. **Data Schema Drafted:**
    - Using paper and pencil, you created a preliminary data schema outlining entities like products, orders, customers, and their relationships.
3. **Single Focus:**
    - By concentrating on business requirements without technical distractions, you set a solid foundation for the technical phase.

These achievements are critical as they ensure your marketplace aligns with real-world business needs.

## Day 2 Activities: Transitioning to Technical Planning

### 1. Define Technical Requirements

The first step is to translate your business goals into clear technical requirements. For each feature identified on Day 1, outline the following:

- **Frontend Requirements:**
  - User-friendly interface for browsing products.
  - Responsive design for mobile and desktop users.
  - Essential pages: Home, Product Listing, Product Details, Cart, Checkout, and Order Confirmation.
- **Sanity CMS as Backend:**
  - Use **Sanity CMS** to manage product data, customer details, and order records. Sanity acts as the database for your marketplace.
  - Focus on designing schemas in Sanity to align with the business goals from Day 1.
- **Third-Party APIs:**
  - Integrate APIs for shipment tracking, payment gateways, and other required backend services.
  - Ensure APIs provide the necessary data for frontend functionality.

### 2. Design System Architecture

Create a high-level diagram showing how your system components interact. Use tools like pen and paper or software like Lucidchart, Figma or Excalidraw. For example, a more detailed architecture might include workflows such as:

```
[Frontend (Next.js)]
  |
[Sanity CMS] ---------> [Product Data API]
  |
[Third-Party API] -----> [Shipment Tracking API]
  |
[Payment Gateway]
```

In this architecture, a typical data flow could look like this:

1. A user visits the marketplace frontend to browse products.
2. The frontend makes a request to the **Product Data API** (powered by Sanity CMS) to fetch product listings and details, which are displayed dynamically on the site.

3. When the user places an order, the order details are sent to Sanity CMS via an API request, where the order is recorded.
4. Shipment tracking information is fetched through a **Third-Party API** and displayed to the user in real-time.
5. Payment details are securely processed through the **Payment Gateway**, and a confirmation is sent back to the user and recorded in Sanity CMS.

This detailed workflow ensures students understand how components interact in a real-world scenario and how data flows seamlessly between them.

This demonstrates how frontend interactions are supported by Sanity CMS for content management, third-party APIs for logistics, and payment gateways for transaction processing. Including such details will help you visualize dependencies and plan integrations effectively.

**Example System Architecture:**

```
[Frontend (Next.js)]
 |    |    |
[Sanity CMS]    [3rd Party APIs]
```

**Key Workflows to Include:**

1. **User Registration:**
   - User signs up -> Data is stored in Sanity -> Confirmation sent to the user.
2. **Product Browsing:**
   - User views product categories -> Sanity API fetches data -> Products displayed on frontend.
3. **Order Placement:**
   - User adds items to the cart -> Proceeds to checkout -> Order details saved in Sanity.
4. **Shipment Tracking:**
   - Order status updates fetched via 3rd-party API -> Displayed to the user.

## 3. Plan API Requirements

Based on your data schema, define the API endpoints needed. Include:

- **Q-Commerce Example:**
  - **Endpoint Name:** /express-delivery-status
  - **Method:** GET

- **Description:** Fetch real-time delivery updates for perishable items.
- **Response Example:** `{ "orderId": 123, "status": "In Transit", "ETA": "15 mins" }`

- **Rental eCommerce Example:**
  - **Endpoint Name:** /rental-duration
  - **Method:** POST
  - **Description:** Add rental details for a specific product.
  - **Payload:** `{ "productId": 456, "duration": "7 days", "deposit": 500 }`
  - **Response Example:** `{ "confirmationId": 789, "status": "Success" }`
- **General eCommerce Example:**
  - **Endpoint Name:** /products
  - **Method:** GET
  - **Description:** Fetch all product details.
  - **Response Example:** `{ "id": 1, "name": "Product A", "price": 100 }`

Ensure API documentation aligns with marketplace-specific workflows to provide clarity for implementation.

Based on your data schema, define the API endpoints needed. Include:

- **Endpoint Name:** /products
  - **Method:** GET
  - **Description:** Fetch all available products from Sanity.
  - **Response:** Product details (ID, name, price, stock, image).
- **Endpoint Name:** /orders
  - **Method:** POST
  - **Description:** Create a new order in Sanity.
  - **Payload:** Customer info, product details, payment status.
- **Endpoint Name:** /shipment
  - **Method:** GET
  - **Description:** Track order status via third-party API.
  - **Response:** Shipment ID, order ID, status, expected delivery date.

## 4. Write Technical Documentation

Document your system architecture, workflows, and API requirements in a structured format. Use headings, diagrams, and bullet points for clarity. Below are examples of documents commonly used in the industry to ensure professional standards:

- **System Architecture Document:** Describes the overall design and interaction between components.
- **API Specification Document:** Details the endpoints, methods, payloads, and responses.
- **Workflow Diagram:** Visualizes user interactions and data flows.
- **Data Schema Design:** Defines entities and relationships for databases or CMS.
- **Technical Roadmap:** Outlines the steps to complete the project, milestones, and deliverables.

When complete, your submission should be titled **"Marketplace Technical Foundation - [Your Marketplace Name]"** to reflect a professional approach. This document should include:

1. **System Architecture Overview:**
    - Include a clear diagram showing how the frontend interacts with Sanity CMS and third-party APIs.
    - Briefly describe the role of each component.
2. **Key Workflows:**
    - List user workflows, such as "User adds products to cart," with step-by-step details of interactions between components.
3. **Category-Specific Instructions:**
    - **Q-Commerce:** Focus on real-time inventory updates, delivery SLA tracking, and express delivery workflows. Example endpoint: /express-delivery-status to fetch real-time tracking.
    - **Rental eCommerce:** Include workflows for rental duration, condition reports, and return management. Example schema field: rentalDuration, depositAmount, conditionStatus.
    - **General eCommerce:** Cover workflows for product browsing, cart management, and order placement. Example endpoint: /products for fetching product listings.

4. **API Endpoints:**
   - Provide a table format:
     - **Endpoint:** /products
     - **Method:** GET
     - **Purpose:** Fetches all product details
     - **Response Example:** `{ "id": 1, "name": "Product A", "price": 100 }`
5. **Sanity Schema Example:**

Example for Product:

```
export default {
  name: 'product',
  type: 'document',
  fields: [
    { name: 'name', type: 'string', title: 'Product Name' },
    { name: 'price', type: 'number', title: 'Price' },
    { name: 'stock', type: 'number', title: 'Stock Level' }
  ]
};
```

This structured documentation will serve as a blueprint for your project and a professional reference for clients or stakeholders.

## 5. Collaborate and Refine

Collaboration is a key professional skill that will not only enhance the quality of your project but also prepare you for real-world teamwork. Follow these guidelines to collaborate effectively while ensuring individual submissions:

1. **Group Discussions:**
   - Organize brainstorming sessions with your peers to exchange ideas and solve common challenges.
   - Use tools like Slack, Discord, or Google Meet to facilitate discussions.
   - Focus on sharing innovative approaches to system architecture and API design.
2. **Peer Review:**
   - Share your technical plans with teammates and mentors for constructive feedback.
   - Review each other's workflows, data schemas, and documentation to identify potential improvements.

3. **Version Control:**
    - Use GitHub or similar platforms to track changes and collaborate on diagrams or drafts.
    - Make sure to commit changes regularly and write clear commit messages to maintain transparency.
4. **Divide and Conquer:**
    - While brainstorming can be done as a group, ensure that each team member works on their own document for submission.
    - Encourage diverse perspectives by allowing individual creativity within a shared framework.
5. **Submission Requirements:**
    - Remember, submissions must be individual. Even if you collaborate, ensure that your final document reflects your unique understanding and approach.

By practicing professional collaboration, you will learn how to balance teamwork and individual accountability, a critical skill for your career.

Work with your peers to review and refine your technical plan. Discuss:

- Innovative approaches to solving common challenges.
- Feedback on your system architecture or API design.
- Suggestions for improving scalability or performance.

## Key Outcome of Day 2

By the end of Day 2, students should have achieved the following outcomes:

1. **Technical Plan Aligned with Business Goals:**
    - A comprehensive plan that reflects the unique requirements of their marketplace type (Q-Commerce, Rental eCommerce, or General eCommerce).
2. **System Architecture Visualized:**
    - A clear diagram illustrating how the frontend interacts with Sanity CMS and third-party APIs.
    - Architecture should include specific workflows tailored to their marketplace type:

        - **Q-Commerce:** Real-time inventory updates and SLA tracking.
        - **Rental eCommerce:** Rental duration, deposit handling, and condition tracking.
        - **General eCommerce:** Standard workflows like product browsing, cart management, and order placement.

3. **Detailed API Requirements Documented:**
    - A list of endpoints, methods, and expected responses, including category-specific examples.
4. **Sanity Schemas Drafted:**
    - Schemas designed to handle key data entities such as products, orders, customers, and additional fields relevant to the specific marketplace type.
5. **Collaborative Feedback Incorporated:**
    - Input from peers and mentors to refine and enhance their technical documentation and plans.
6. **Portfolio-Ready Submission:**
    - A polished, professional document that can be used in job interviews or to showcase their skills in startup or client presentations.

This preparation ensures that students have a solid foundation to begin implementation on Day 3, setting them up for success in building a fully functional and tailored marketplace.

## Industry Best Practices to Emphasize

1. **Plan Before You Code:**
    - Always create a clear roadmap before starting technical work to avoid unnecessary rework.
2. **Use the Right Tools:**
    - Leverage tools like Sanity CMS and APIs to reduce backend complexity while focusing on frontend innovation. Sanity CMS offers customizable schemas, built-in APIs, and real-time collaboration, allowing frontend developers to manage data effectively. These features reduce repetitive backend tasks, enabling teams to focus on building dynamic and user-friendly interfaces.

3. **Collaboration:**
    - Share your plans with peers and mentors for feedback.
4. **Focus on User Experience:**
    - Ensure that the technical foundation supports a seamless and intuitive user journey.

## Submission Guidelines

**Submission Title:**

- **"Marketplace Technical Foundation - [Your Marketplace Name]"**

**Submission Approach:**

1. **Repository Submission:**
    - Create a dedicated folder named "Documentation" in your repository.
    - Upload your technical document and any associated diagrams or schemas to this folder.
2. **Document Structure:**
    - Follow the industry-standard format provided above.
    - Include collaboration notes if you worked in a group, briefly summarizing how ideas were shared and feedback was incorporated.
3. **File Naming Convention:**
    - Ensure all files are named clearly, e.g., SystemArchitecture_Day2.pdf, APIEndpoints.xlsx, SanitySchema.js.
4. **Review and Quality Check:**
    - Double-check your diagrams, schemas, and written content for clarity and accuracy.
    - Collaborate with peers or mentors for feedback to refine your submission.

This structured submission will help you showcase your ability to create professional-level technical documents, a critical skill for future projects and client interactions.

## What's Next?

On Day 3, we will provide the API as a reference point. This API can be utilized to check schema structure or migrate the schema into your Sanity project. While the provided

API will be readonly, you have the flexibility to create your own data schema in Sanity for GET, POST, UPDATE, PATCH, and DELETE operations. Specific guidance includes:

- **Q-Commerce:** Use the provided API schema to manage perishable goods and SLA tracking.
- **Rental eCommerce:** Design schemas to handle rental-specific fields like duration, deposit, and condition reports.
- **General eCommerce:** Implement standard workflows like inventory management, order placement, and payment integration.

The choice is yours—use the provided API schema or innovate with your own to best suit your marketplace's unique requirements.

Stay focused, think critically, and aim for innovation!