# **Hackathon Day 2: Technical Foundation**

## **Introduction**

## **Shop.co** is a cutting-edge, responsive e-commerce platform designed as the cornerstone of a robust online marketplace. Created during **Hackathon 3**, this project takes the UI/UX designs developed in **Hackathon 2** and brings them to life, transforming them into a fully operational and interactive website.

### Project Overview

The development of **Shop.co** centers around creating a flexible and scalable business infrastructure, with a strong emphasis on organizing data for products, customers, orders, payments, and shipments. By utilizing **Sanity CMS** for streamlined content management and **Next.js** for frontend development, Shop.co ensures smooth data flow and optimal performance across all aspects of the platform.

### **Our Vision**

**Shop.co** aims to offer a marketplace that is not only efficient but also intuitive and enjoyable for users. Our goals include:

* **Real-Time Updates**: Provide live tracking for orders, shipments, and payments, ensuring customers are always in the loop.
* **User-Friendly Navigation**: Deliver a seamless and engaging browsing experience with easy-to-use navigation.
* **Effortless Integration**: Achieve smooth communication between backend and frontend systems using the latest technologies, enhancing performance.
* **Enhanced Frontend Performance**: Focus on optimization using **Next.js** for server-side rendering, resulting in faster load times and better SEO.

## **1. Simple System Design Flowchart**

[User]

|

v

[Website (Next.js)] -----> [Sanity CMS (Product Storage)]

| |

v v

[Helpers (APIs)] [Payment Gateway (Payments)]

|

v

[Delivery Tracker (API)]

## **2. Easy System Architecture**

* **Website (Next.js):** The site where users shop and check out.
* **Sanity CMS:** Saves product info, orders, and customer details.
* **Helper APIs:** Help with shipping and payments.
* **Payment Gateway:** Handles online payments safely.
* **Delivery Tracker:** Shows the order’s delivery status.

## **3. How Everything Works**

### **a. User Registration (Sign-Up)**

[User] → [Sign-Up Form] → [Sanity CMS Saves Info] → [Sends Confirmation Email]

### **b. Browsing Products**

[User] → [Product Page] → [Sanity CMS] → [Products Show on Website]

### **c. Placing an Order**

[User] → [Add to Cart] → [Checkout] → [Order Saved in Sanity CMS] → [Payment Processed]

### **d. Tracking the Order**

[User] → [Order Status Page] → [Delivery API] → [Shows Tracking Info]

## **4. Website Features**

* **Sign-Up/Login:** Create and log into an account.
* **View Products:** Browse and check product details.
* **Place Orders:** Add to cart, checkout, and pay.
* **Track Delivery:** See where your order is.
* **Admin Panel:** Manage products and handle customer issues.

## **5. How Data Works (DSA)**

* **Getting Product Info:**
  + Use GET requests to show products from Sanity CMS.
* **Saving Orders:**
  + Use POST requests to store order details.
* **Tracking Orders:**
  + Use GET requests to check delivery status.
* **User Login/Sign-Up:**
  + Use POST requests to register or log in users.

## **6. Important API Links**

* **GET /products** → Show products.
* **POST /orders** → Save new orders.
* **GET /shipment** → Track delivery.
* **POST /auth/register** → New user sign-up.
* **POST /auth/login** → User login.

**7. Sanity CMS Product Example**

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 Left' },

{ name: 'description', type: 'text', title: 'Product Details' },

{ name: 'category', type: 'string', title: 'Category' },

{ name: 'image', type: 'image', title: 'Product Image' }

]

};

**Conclusion:**

**Shop.co** is a modern solution to the evolving e-commerce landscape, combining seamless user experiences with a powerful and scalable backend. By leveraging technologies like **Next.js**, **Sanity CMS**, and secure APIs, the platform ensures efficiency, security, and flexibility for both users and administrators.