DAY 2 PLANNING THE TECHNICAL FOUNDATION

Step 1: Define Technical Requirements

- **Next.js: A React-based framework for building server-rendered and static web applications. It provides excellent performance, SEO optimization, and flexibility with API routes, making it a strong foundation for e-commerce platforms.**

- **Sanity CMS: A versatile headless CMS used as the backend to manage product data, customer details, and order records. Its real-time collaboration features and flexible schema design are ideal for dynamic marketplaces.**

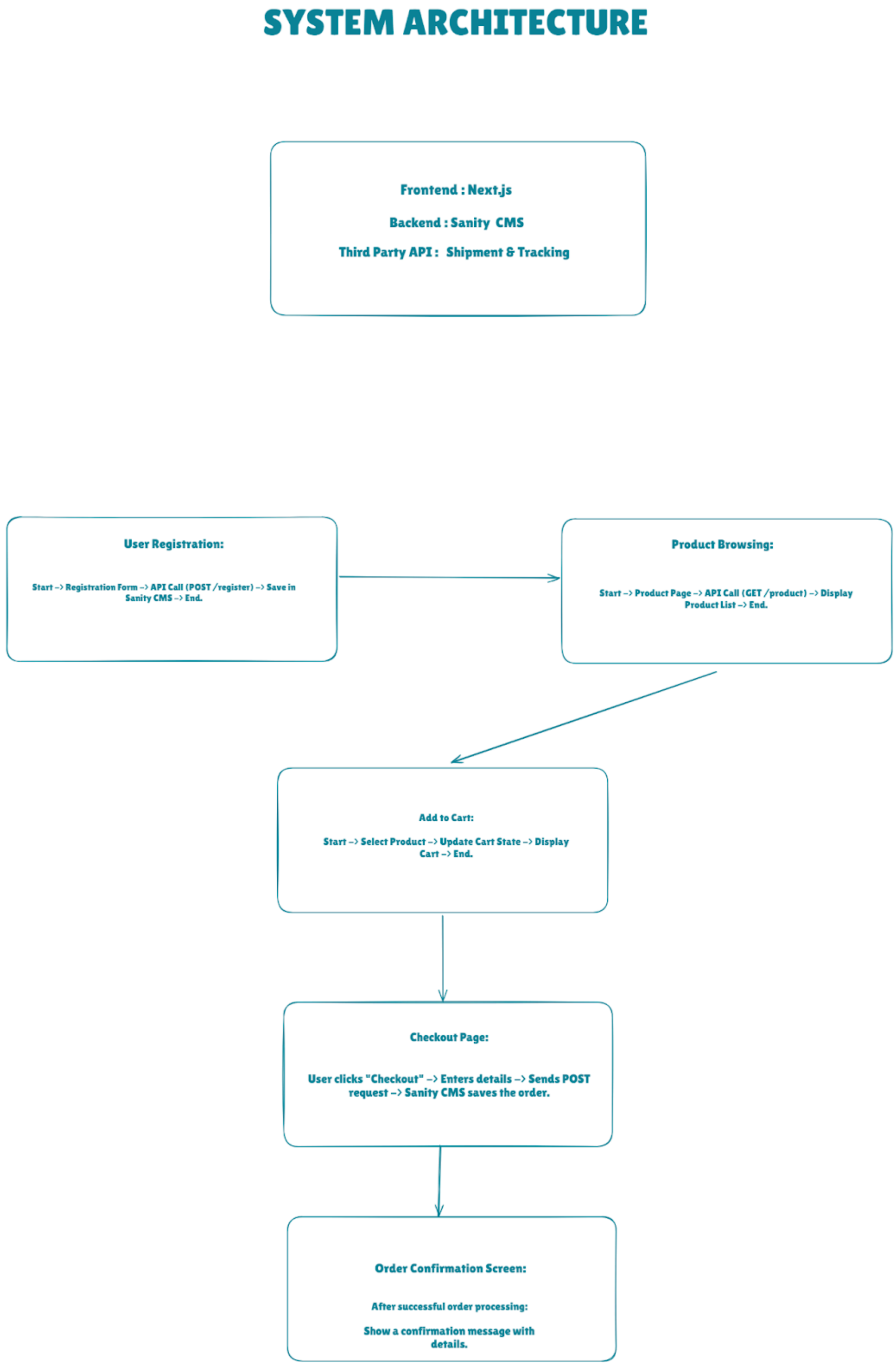
- **Tailwind CSS: A utility-first CSS framework that facilitates the rapid creation of responsive and customizable designs, ensuring a sleek and consistent user interface across all devices.**

- **Third-Party Shipment API: Integrates real-time shipment tracking and delivery updates to enhance the post-purchase experience and boost customer satisfaction.**

- **Vercel: A high-performance cloud platform for deploying and hosting web applications. It supports Next.js apps with quick deployment, scalability, and optimized performance.**

DAY 2 PLANNING THE TECHNICAL FOUNDATION

Step 2: Plan Design Architecture



SYSTEM ARCHITECTURE

Frontend : Next.js

Backend : Sanity CMS

Third Party API : Shipment & Tracking

User Registration:

Start –> Registration Form –> API Call (POST /register) –> Save in Sanity CMS –> End.

Product Browsing:

Start –> Product Page –> API Call (GET /product) –> Display Product List –> End.

Add to Cart:

Start –> Select Product –> Update Cart State –> Display Cart –> End.

Checkout Page:

User clicks "Checkout" –> Enters details –> Sends POST request –> Sanity CMS saves the order.

Order Confirmation Screen:

After successful order processing:

Show a confirmation message with details.

DAY 2 PLANNING THE TECHNICAL FOUNDATION

Step 3: Plan API Requirements

General E-Commerce :

List of Endpoints:

1. **User Signup**

   - ○ **Method: POST**
   - ○ **Endpoint: `/signup`**
   - ○ **Purpose: Creates a user account and stores user information in Sanity CMS.**

Payload:

```
 {

  "name": "username",

  "email": "user email",

  "password": "user password"

}
```

   - ○

2. **Fetch Products**

   - ○ **Method: GET**
   - ○ **Endpoint: `/product`**
   - ○ **Purpose: Fetches all product data from Sanity CMS.**

DAY 2 PLANNING THE TECHNICAL FOUNDATION

Response:

```
 [
  {
    "_id": "2214",
    "name": "product name",
    "description": "product description",
    "stock": 40,
    "image": "image url"
  },
  { ... },
  { ... }
]
```

○

3. **Submit Order**
   - **Method: POST**
   - **Endpoint: `/order`**
   - **Purpose: Submits order details to Sanity CMS.**

Payload:

```
 {
  "order_Id": 939,
  "order_Status": "Pending",
  "customerInfo": {
    "name": "any name",
    "phone": 12345678901,
    "address": "home address"
  },

  "productDetail": {

    "quantity": 3,

    "price": 390

  },

  "time_stamp": "new Date()",

  "paymentMethod": "COD"

 }
```

-

4. **Track Shipment**

    - **Method: GET**
    - **Endpoint:** `/shipment`
    - **Purpose: Retrieves shipment tracking information from a third-party API.**

Response:

```
{

 "shipment_Id": "239889sh",

 "order_Id": 939,

 "status": "In Transit",

 "deliveryDate": "new Date()"

}
```

-