# MARKETPLACE BUILDER HACKATHON 2025

HACKATHON DAY 2

PLANNING THE TECHNICAL FOUNDATION

# DEFINE TECHNICAL REQUIREMENTS

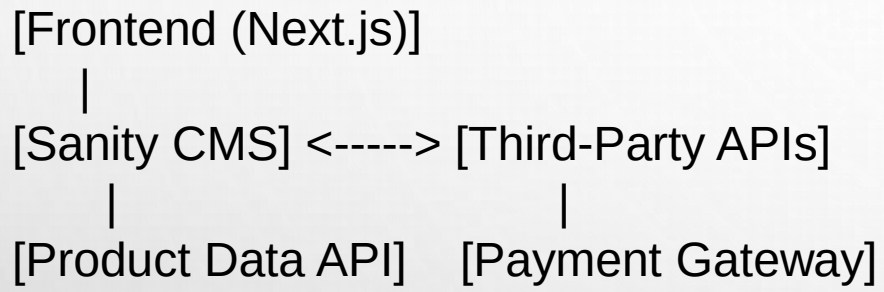
•**Frontend**: The client-side interface where users interact with the platform. It will be built with Next.js and will interact with the backend and CMS to display data.

•**Sanity CMS**: The content management system for managing products, orders, and customer data. Sanity platform will handle the dynamic content for products and customer data, providing an API for interaction.

•**Third-party APIs**: Services like payment gateways, shipping providers, and any external systems that integrate with the for additional functionality.

# DESIGN SYSTEM ARCHITECTURE

- A diagram for component interactions.




- Focus on workflows such as:
    1. Product browsing: frontend fetches data from sanity CMS via API.
    2. Order placement: order details sent to sanity and payment gateway.
    3. Shipment tracking: status fetched from a third-party API.

# API REQUIREMENTS

## API ENDPOINTS

| Endpoint | Method | Payload | Response |
|---|---|---|---|
| /api/products | GET | None | List of products with details (name, price, image) |
| /api/product/{id} | GET | Product ID | Product details (name, price, description, stock) |
| /api/cart | POST | { product ID, quantity } | Updated cart with product details and total price |
| /api/checkout | POST | { user ID, shipping Address, payment Details } | Order confirmation and payment receipt |
| /api/user/register | POST | { name, email, password } | Success message or error |
| /api/user/login | POST | { email, password } | JWT token or error message |

# EXAMPLE API RESPONSES

**GET /api/products**

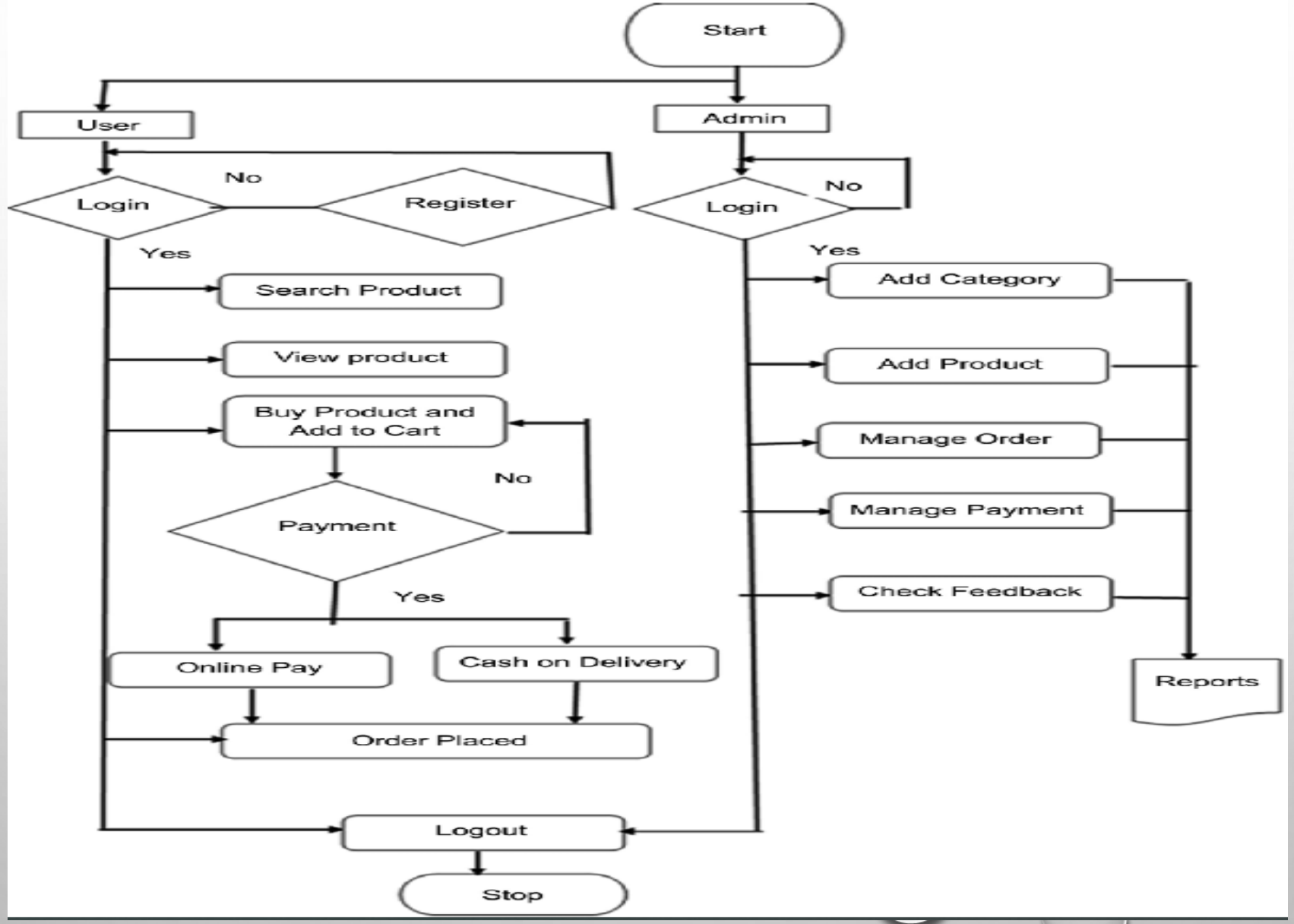

[ { "ID": 1, "NAME": "CERAMIC PLANT POT",
"PRICE": 20.99,
"IMAGE": "/IMAGES/POT.JPG",
"DESCRIPTION": "A BEAUTIFUL CERAMIC PLANT POT.",
"STOCK": 50 },

{ "ID": 2, "NAME": "WOODEN TABLE",
"PRICE": 120.99,
"IMAGE": "/IMAGES/TABLE.JPG",
"DESCRIPTION": "A STURDY WOODEN TABLE FOR DINING.",
"STOCK": 30 } ]

**POST /api/checkout**

{ "STATUS": "SUCCESS",
"ORDER ID": "12345",
"TOTAL": 141.98,
"MESSAGE": "ORDER PLACED SUCCESSFULLY!" }

# KEY WORK FLOW



# KEY WORKFLOWS

## User registration workflow

- **User visits the registration page**: the user is prompted to enter their details (e.g., Name, email, password).
- **Form validation**: the system validates the form data (e.g., Checks for valid email format, strong passwords).
- **Account creation**: if valid, the system creates a new account and stores the data in the database.
- **Confirmation email**: the user receives a confirmation email to verify their email address.
- **User is redirected to the login page**: once verified, the user can log in with their credentials.

## Product browsing workflow

- **User browses products**: users can search, filter, and view product categories (e.g., Plant pots, ceramics).
- **Product details**: clicking on a product brings up the product details page, including images, description, price, and availability.
- **Add to cart**: users can add products to their shopping cart, and the cart's contents will be updated dynamically.

## Order placement workflow

- **User reviews cart**: before checkout, users can review their cart and modify quantities or remove items.
- **Checkout process**: the user enters shipping and payment details.
- **Payment processing**: the system integrates with a payment gateway (e.g., Stripe or PayPal) to process payments.
- **Order confirmation**: after successful payment, the user receives an order confirmation, and an order entry is created in the database.

# SANITY SCHEMA

PRODUCT AND CATEGORY

```
anity > schemaTypes > TS product.ts
import { defineType, defineField } from "sanity"

export const product = defineType({
    name: "product",
    title: "Product",
    type: "document",
    fields: [
        defineField({
            name:"category",
            title:"Category",
            type:"reference",
            to:[{
                type:"category"
            }]
        }
        ),
        defineField({
            name: "name",
            title: "Title",
            validation: (rule) => rule.required(),
            type: "string"
        }),
        defineField({
            name: "slug",
            title: "Slug",
            validation: (rule) => rule.required(),
            type: "slug"
        }),
        defineField({
            name: "image",
            type: "image",
            validation: (rule) => rule.required(),
            title: "Product Image"
        }),
        defineField({
            name: "price",
            type: "number",
            validation: (rule) => rule.required(),
            title: "Price",
        }),
        defineField({
            name: "quantity",
            title: "Quantity",
            type: "number",
            validation: (rule) => rule.min(0),
          }),
```

```
            type: "number",
            validation: (rule) => rule.required(),
            title: "Price",
        }),
        defineField({
            name: "quantity",
            title: "Quantity",
            type: "number",
            validation: (rule) => rule.min(0),
          }),
        defineField({
            name: "tags",
            type: "array",
            title: "Tags",
            of:[{
                type: "string"
            }]
        }),
        defineField({
            name: 'description',
            title: 'Description',
            type: 'text',
            description: 'Detailed description of the product',
          }),
          defineField({
            name: 'features',
            title: 'Features',
            type: 'array',
            of: [{ type: 'string' }],
            description: 'List of key features of the product',
          }),
          defineField({
            name: 'dimensions',
            title: 'Dimensions',
            type: 'object',
            fields: [
              { name: 'height', title: 'Height', type: 'string' },
              { name: 'width', title: 'Width', type: 'string' },
              { name: 'depth', title: 'Depth', type: 'string' },
            ],
            description: 'Dimensions of the product',
          }),
    ]
})
```

```
import { defineType,defineField } from "sanity";

export const Category = defineType({
    name: "category",
    title: "Category",
    type: "document",
    fields:[
        defineField({
            name: "name",
            title: "Name",
            type: "string",
            validation: (rule) => rule.required(),
        }),
        defineField({
            name: "slug",
            title: "Slug",
            type: "slug",
            validation: (rule) => rule.required(),
            options: {
                source: "name",
            }
        })
    ]
})
```

## Collaboration notes

- Worked closely with peers to define the data schema and ensure proper workflows for order placement and product management.
- Faced challenges around integrating third-party APIs for payments and shipment tracking, but received valuable feedback on improving the flow.
- Incorporated suggestions on better api documentation and improving the clarity of system architecture diagrams.

## CONCLUSION

This document outlines the technical foundation of the e-commerce platform for selling plant pots, ceramics, tables, and chairs. The system is designed with scalability and ease of integration in mind, ensuring smooth user experiences and robust backend management through sanity CMS.