# Hackathon-3
# Day#02

# MarketPlace Technical Foundation - Shoes

## 1. System Architecture Document

### 1.Overview

The marketplace website for NikeShoes is designed to allow customers to browse and purchase stylish footwear. The system consists of several key components:

- **Frontend**: A React-based web application that provides an intuitive interface for customers to explore and purchase Nike shoes.

- **Backend**: A robust API that handles data processing, including fetching shoe details, managing user interactions, and performing CRUD operations

- **CMS**: A CMS to manage and store content such as shoe details, categories, promotional offers, and other relevant information.

- **Database**: The database is integrated with the CMS to store data related to shoes, customer profiles, and order histories.

## 1.1. Frontend

- **Homepage**: A visually engaging page showcasing promotional banners, featured shoes, categories (e.g., Running, Casual, Training), and easy navigation.

- **Product Listing**: Displays a grid of Nike shoes with filters such as size, color, category, price range, and special editions.

- **Product Details:** A detailed page showing information for each shoe, including images (from various angles), descriptions, sizes available, and reviews.

- **Cart:** A section to review and modify selected items before proceeding to checkout.

- **Checkout**: A secure and user-friendly interface to finalize orders, enter payment information, and select delivery options.

- **Track Order**: A dedicated page to track the real-time delivery status of placed orders.

## 2. Sanity CMS

- **Product Management:** Store and manage shoe details such as name, price, size, color, category, and descriptions.
- **Order Management:** Track customer orders, payment statuses, and shipping details.

## 3.Third-Party APIs

### 1. Payment Gateway

Integrate Stripe, PayPal, or similar payment gateways for secure and reliable payment processing.

**Features include:**

Credit/debit card payment support.

Multi-currency support for international transactions.

Refund and dispute management for seamless customer service.

### 2. Shipment Tracking

Use AfterShip or similar courier APIs to provide real-time tracking information for Nike shoe orders.

**Features include:**

Real-time status updates (e.g., Shipped, Out for Delivery, Delivered).

Notifications for order movement to keep customers informed.

Integration with multiple courier services for global delivery.

## 3. Mock API

Create a Mock API for development and testing to simulate data and API responses without relying on live APIs.

**Features include:**

Simulating product data for Nike shoes, including styles, sizes, and inventory.

Order status updates for testing the complete order lifecycle.

Shipment tracking updates for seamless testing of delivery processes.

# System Workflow

### Login/Signup:

- User login/signup by filling the form.
- Frontend sends the data to the backend API.
- Backend stores the user data in the database.
- User login/signup and access their personalized dashboard.

- **Home page:**

  User lands on the home page after login or signup.

- **Product Page:**

  User views a list of available products.

- **Single Product Page:**

  User clicks on a product to view details

- **Add to Cart**

  Users add a product to their cart.

- ## Cart Page

  User navigates to the cart page by clicking the cart icon.

- ## Checkout

  User clicks the "Place Order" button and provides order details.

- ## Track Order

  Users track their order status.

# Third-Party Services

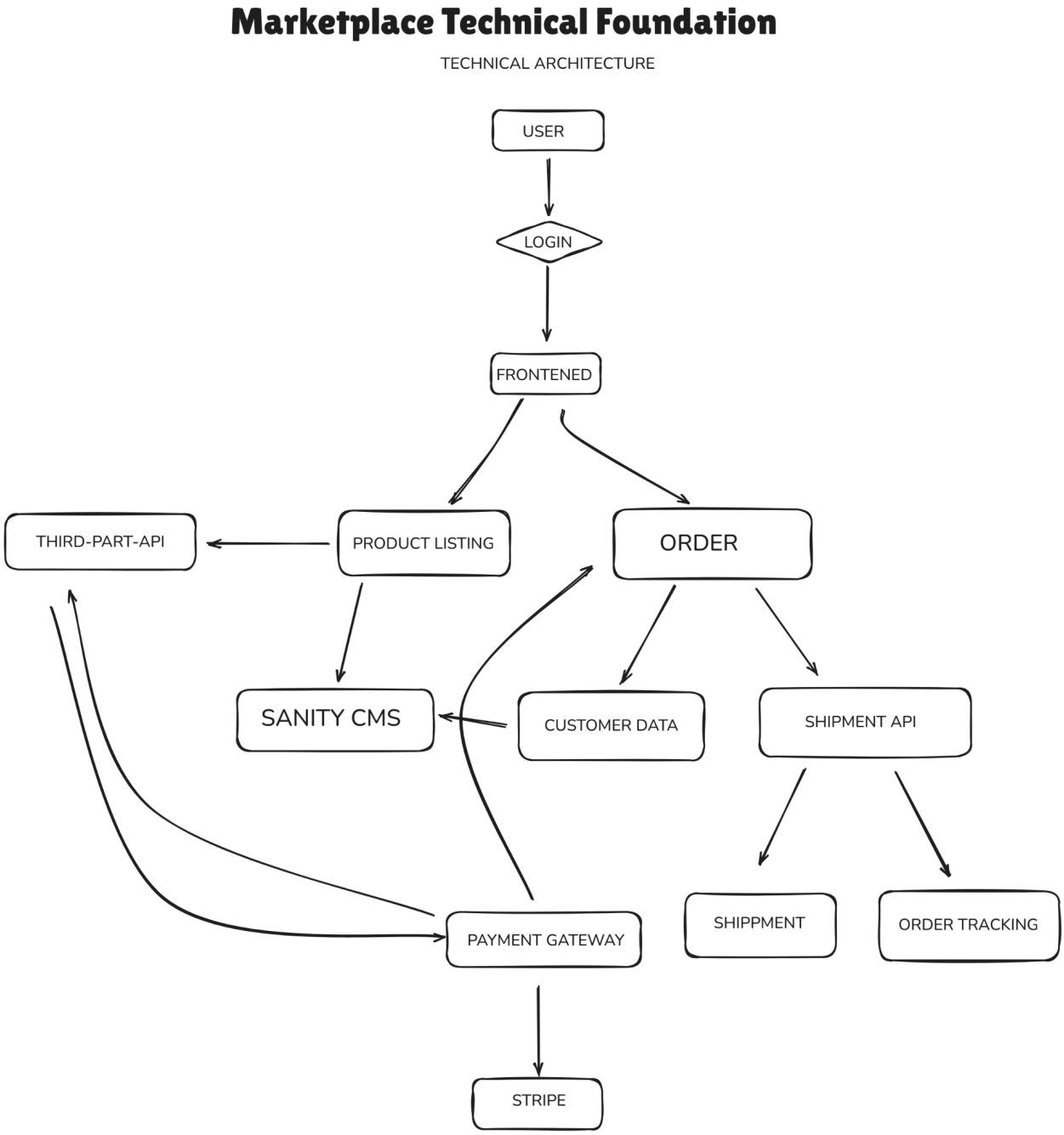
### Mock.io

o Provides a mock API for product data.
o Data fetched by the Backend and stored in Sanity for future use.

### ShipEngine

o Used to track order shipping status.
o Provides tracking information to the Backend, which is displayed to the user.

# Design System Architecture:



**Sanity Schema:-**

**Product :**

```
export default {
name: 'product',
 title: 'Product',
type: 'document',
fields: [
 {
 name: 'name',
 title: 'Product Name',
 type: 'string',
 validation: Rule => Rule.required().min(3).max(50)
},
 {
 name: 'slug',
title: 'Slug',
 type: 'slug',
 options: {
 source: 'name',
 maxLength: 96,
 },
validation: Rule => Rule.required()
 },
{
 name: 'description',
title: 'Description',
type: 'text',
validation: Rule => Rule.required().max(300)
},
{
  name: 'price',
title: 'Price',
type: 'number',
validation: Rule => Rule.required().positive()
},
{
{ name: 'category',
 title: 'Category',
 type: 'string',
```

```
options: {
list: [
{ title: 'Korean Style', value: 'korean-style' },
{ title: 'Western Clothes', value: 'western-clothes' },
 { title: 'Old Money Fashion', value: 'old-money-fashion' },
],
}
validation: Rule => Rule.required()
}
{
{ name: 'images',
title: 'Images',
type: 'array',
of: [{ type: 'image'}],
options: {
hotspot: true,
},
},
{
 name: 'stock',
 title: 'Stock',
 type: 'number',
 validation: Rule => Rule.required().min(0),
},
 {
 name: 'sizes',
title: 'Sizes',
 type: 'array',
 of: [{ type: 'string' }],
options: {
list: [
{ title: 'Small', value: 'S' },
 { title: 'Medium', value: 'M' },
{ title: 'Large', value: 'L' },
 { title: 'Extra Large', value: 'XL' },
],
},
 validation: Rule => Rule.required().min(1)
},
 {
 name: 'createdAt',
 title: 'Created At',
type: 'datetime',
initialValue: () => new Date().toISOString(),
  },
```

```
  ],
  };
```

## Purpose of Documentation

**1. Team Alignment:** Provides a shared understanding of the project architecture, workflows, and APIs to ensure all team members are aligned.

**2. Scalability:** Acts as a reference guide for adding new features or scaling the system without disrupting the existing architecture.

**3. Onboarding:** Simplifies the onboarding process for new developers by giving them clear insights into the system.

**4. Troubleshooting:** Helps identify and resolve issues by offering detailed workflows and data structures.

**5. Consistency:** Ensures uniformity in code standards and workflows across the team.

**6. Client Communication:** Serves as a professional document to explain the project's architecture and workflows to stakeholders or clients.

**Prepared by:** Mohammad Danish
**Slot:** Saturday 2 to 5
**Teacher:** Ali Aftab & Mohammad Bilal