Hackathon-3 Day-2

MarketPlace Technical Foundation Shoes

1 system Architecture Documents

_ 1 overview

The marketplace website for nike shoes is designed at 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 interaction,and performing crud operations .

➢ **CMS:** A CMS TO Manage and store content such as shoe details , category ,promotional offers , and other relevant information.

➢ **Database:** the database is integrated with the CMS to store related to shoes ,customer profiles , and other histories

1.1 FRONTEND

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

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

- **Product detail :** A detail page showing information for each shoe , including images (from various angles) , description ,size available , and review.

- **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 delicate page to Track the real delivery status of placed Orders

2 Sanity CMS

- **Product Management :** Store and manage shoes details such as **Name , Price , Size , Color , And Description .**
- **Order Management :** Track Customer **Order , Payment Status , and Shipping** details.

3 Third-Party-APIs

1 Payment Gateway

Integrate **Strip** , Paypal , or Similar Payment Gateways for secure and reliable payment Processing.

Feature includes :

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 Shoes** Orders.

Features include :

Real-Time Status Updates (e.g. Shipped , Out for delivery , deliverd).

Notifications for order movement to keep customers informed .

integration with multiple courier servises 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.

Feature include :

Simulating product for **Nike Shoes** , including styles , size , 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 backend API
- Backend stores the user data in the database.
- User login/signup and process their personalize 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 :
User add a product to their cart

Cart page :

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

Check out :

User clicks the **place Order** button and provides order details .

Track Order :

User track their order status .

Third-Party Services :

Mock io

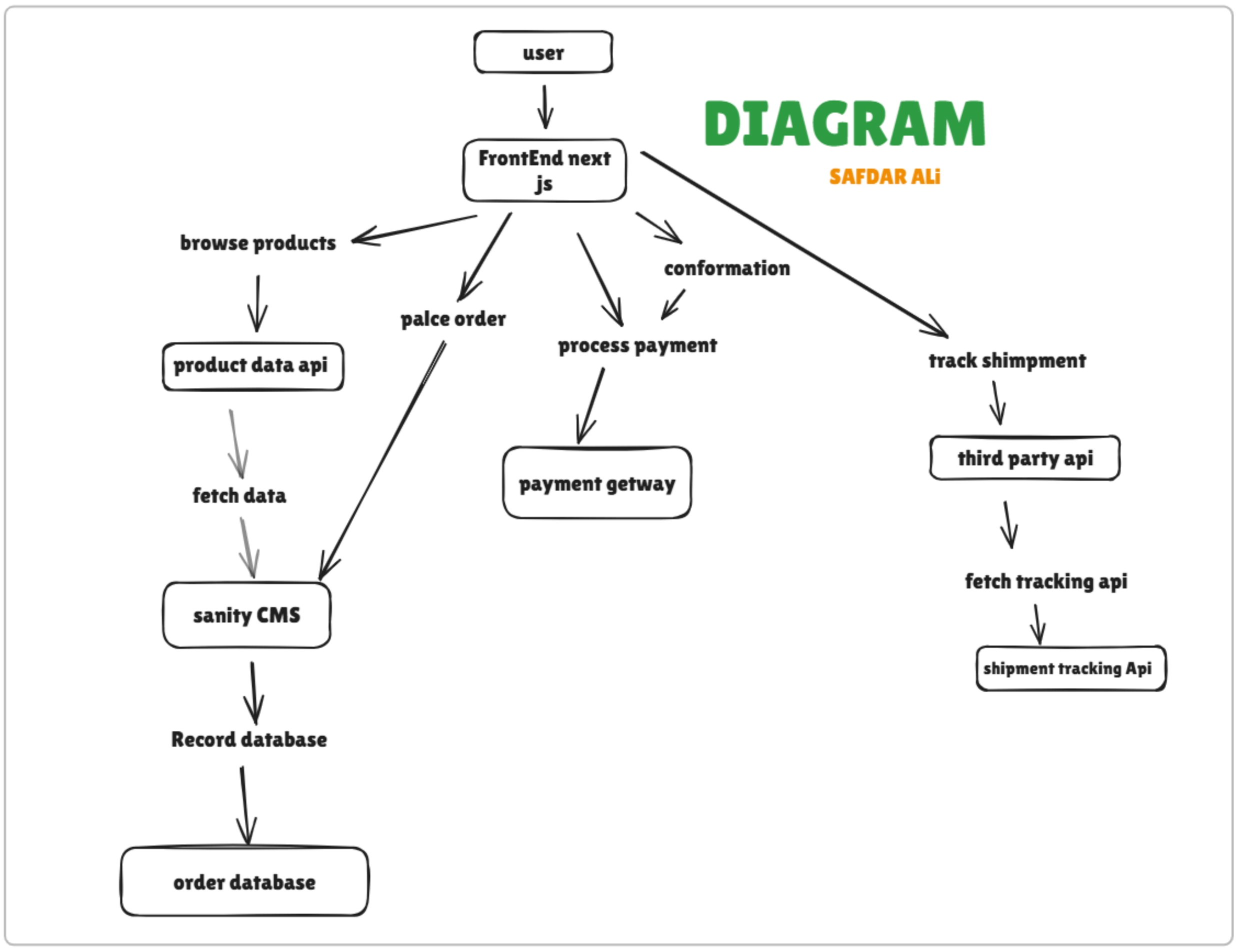
- Provides a mock API product data
- Data fetched by the backend and stored in sanity for future use .

ShipEngine

- Use to track order shipping status .
- Provides tracking information to the backend which is displayed to the user .

Design System Architecture :

Frame



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',
      validation: (Rule) => Rule.required(),
    },
  ],
};
```

```
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 shered understanding of the project architecture ,workflow and APIs to ensure all team are aligned .
2. **Scalability :** Acts as a reference guide for adding new feature or scaling the system without disrupting the exiting architecture .
3. **Onboarding** : simplifies the onboarding process for new developers by giving them clear insights into the system .
4. **Troubleshooting** : helps identify and resolve isusses by offering detailed workflow and data structures .
5. **Consistency :** Ensures uniformity in code standards and workflow across the team .
6. **Client Communication** : Service as a professional document to explain the project's architecture and workflow to stakeholders or Clients

Prepared by : **SAFDAR ALI / SAIFI**
Slot : **Sunday** ***2pm to 5pm***
Teacher : **Ali jawwad**