Made by Kulsoom Adnan

Hackathon Day 02

Marketplace Technical Foundation

Furniro

Introduction :

Welcome to this presentation on the "Marketplace Technical Foundation." In this session, I will discuss how to plan and establish the technical foundation for our hackathon project. The key focus areas include defining technical requirements, designing system architecture, planning API requirements, and preparing technical documentation. This roadmap is tailored to ensure that we build a scalable and efficient marketplace platform.

1. Technical Requirements

The Technical Requirements are designed to set a strong foundation for the development of your e-commerce marketplace. This phase translates business goals into clear, actionable tasks that ensure both frontend and backend components work harmoniously to deliver a seamless user experience.

Frontend Resources:

1. Next.js

- **Purpose**: A powerful React framework for building fast and scalable web applications.
- **Usage:** Next.js will be used to create the entire frontend of the marketplace, including routing, dynamic rendering, and server-side rendering (SSR).
- **Why Next.js**: It enables fast performance, SEO optimization, and automatic code splitting for better user experience.

2. Tailwind CSS

- **Purpose**: A utility-first CSS framework for designing custom UIs without writing custom CSS.
- **Usage:** Tailwind CSS will be used to style the frontend components of the marketplace, ensuring a responsive and modern design.
- **Why Tailwind CSS**: It allows for rapid styling with a low learning curve and offers complete flexibility to customize designs.

3. ShadCN

- **Purpose:** A UI component library built with Tailwind CSS that offers pre-designed components.
- **Usage**: ShadCN will be used for UI elements like buttons, modals, forms, and input fields, speeding up the design and development process.
- **Why ShadCN**: It provides ready-to-use components that are fully customizable, ensuring consistency in design and speeding up development time.

4. React

- **Purpose**: A JavaScript library for building user interfaces.
- **Usage**: React will be the core library for creating dynamic components and managing the state within your e-commerce marketplace.
- **Why React**: React's component-based architecture makes it easier to create interactive UIs and manage state across the application.

Responsive Design:

The platform is designed with responsiveness in mind, ensuring it looks and functions perfectly across mobile and desktop devices.

- **Mobile First Approach**: The design is optimized for smaller screens, ensuring a smooth user experience on smartphones and tablets.
- **Flexible Layout:** The layout adjusts dynamically to different screen sizes, maintaining a consistent structure without compromising content.
- **Adaptive Navigation:** The navigation bar and menus adapt to different screen sizes, with mobile-friendly collapsible options for easy browsing.

Essential Pages:

- **Home Page:** Engaging hero section, key features, promotions, and categories.
- **Product Listing Page:** Display of product categories, filtering, and sorting options for easy browsing.
- **Product Details Page**: Rich product descriptions, high-quality images, specifications, and pricing details.
- **Cart Page:** Allows users to review, modify, and proceed to checkout with their selected items.
- **Checkout Page**: A streamlined, user-friendly process for entering payment and shipping details.
- **Order Confirmation Page**: A reassuring confirmation message with order summary and tracking information.

Backend (Sanity CMS):

Sanity CMS will act as the **central hub** for all content, enabling you to manage product information, customer data, and order records efficiently.

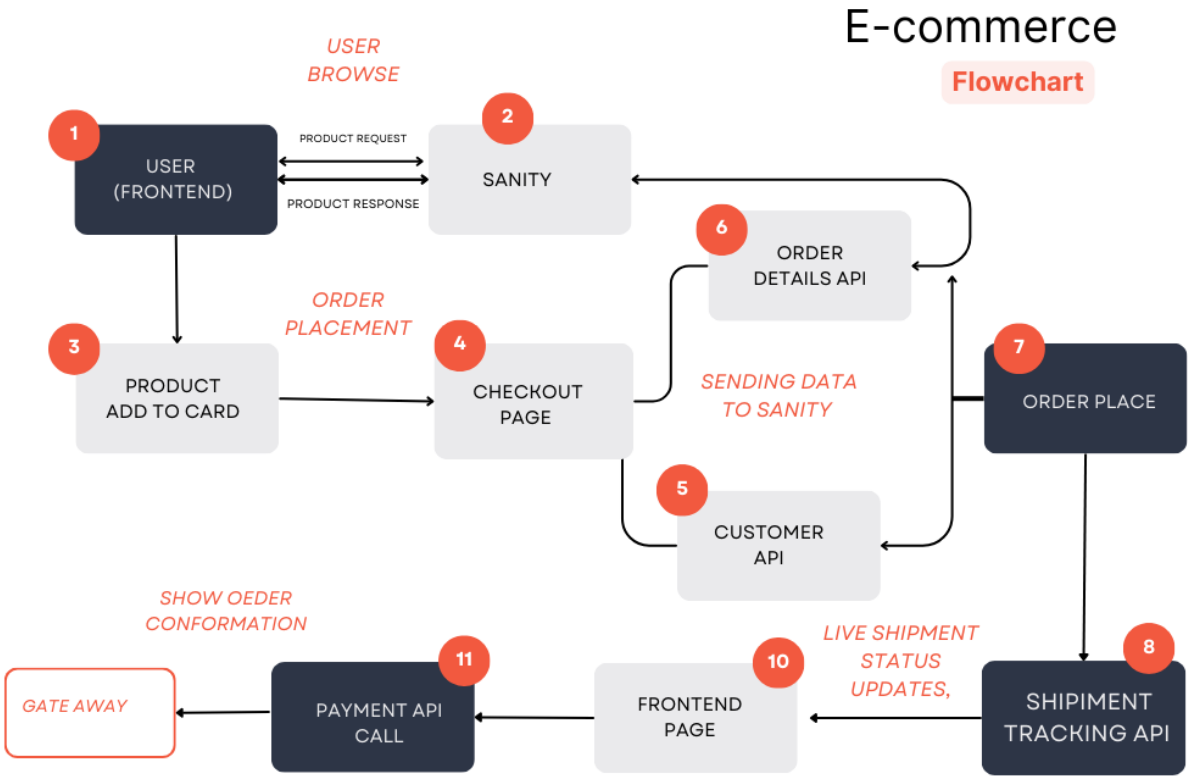
- **Product Data Management**: Store product details such as names, descriptions, prices, and stock information.
- **Customer Data Management**: Maintain user details like names, shipping addresses, and order history.
- **Order Management**: Keep track of every order placed, its status, and relevant customer information.

Usage: Sanity CMS will act as the backend for managing product data, customer details, order records, and more.

Third-Party APIs:

- **Shipment Tracking API**: Integrate a third-party API that provides real-time updates on shipment status and expected delivery times, enhancing customer experience**. ShipEngine** will be used to fetch real-time shipment tracking details, enabling users to track their orders directly from the marketplace.
- **Payment Gateway API:** Secure payment processing via popular gateways like Stripe or PayPal to handle customer transactions efficiently. **stripe** will be used to handle secure payment transactions during the checkout process, ensuring smooth and reliable processing of customer payments.

Design System Architecture



Data Flow Process:

1. **User Browsing**:
    - The user visits the frontend (built with Next.js) and browses the product listings.
    - The frontend requests product data from the **Sanity CMS** via the **Product Data API**, which dynamically displays the information on the site.
2. **Order Placement**:
    - Once the user adds products to their cart and proceeds to checkout, the order details are sent via an API request to **Sanity CMS** for storage and tracking.

3. **Shipment Tracking**:
    - After the order is placed, the **Shipment Tracking API** is called to fetch live shipment status updates, which are displayed to the user.
4. **Payment Processing**:
    - The user provides payment details through the **Payment Gateway API (Stripe)**. Once payment is processed, the user receives a confirmation, and the payment details are securely recorded in **Sanity CMS**.

5. Key Workflows

1. User Registration:

- User signs up and their details are stored in **Sanity CMS**.
- A confirmation message is sent to the user.

2. Product Browsing:

- User browses product categories.
- **Sanity CMS** fetches product data using the **Product Data API**, and the products are displayed dynamically on the frontend.

3. Order Placement:

- User adds products to the cart and proceeds to checkout.
- Order details are recorded in **Sanity CMS**.

4. Shipment Tracking:

- The **Shipment Tracking API** provides order status updates, which are displayed to the user in real-time.

---

6. Conclusion

This architecture provides a clear and scalable way to build a highly functional e-commerce platform using modern technologies like **Next.js**, **Sanity CMS**, and third-party services such as **Stripe** and **ShipEngine**. By integrating these components, we ensure a smooth experience for both the frontend user and the backend system, with seamless data flow and real-time updates.