Day 2 - Marketplace Technical Foundation

Introduction:

This marketplace is envisioned as a cutting-edge e-commerce platform offering a broad selection of high-quality, affordable products. My goal is to provide users with an extensive product range, similar to large-scale platforms like Daraz and Amazon, while ensuring quick delivery and a unique shopping experience. Below is a breakdown of the technical foundation for building this marketplace.

1. Technical Requirements:

Frontend Requirements:

- **User-Friendly Interface:**
  - A clean, intuitive design with easy navigation for browsing products.
  - A search and filtering system to help users quickly find what they need.
- **Responsive Design:**
  - Fully optimized for both mobile and desktop devices, ensuring a smooth experience across platforms.
- **Key Pages:**
  - **Home Page:** Featuring popular categories, product highlights, and special offers.
  - **Product Listings:** Allowing users to explore products by category, with filtering and sorting options.
  - **Product Details:** Detailed information, including images, pricing, and availability.
  - **Cart & Checkout:** Simplified cart management and checkout process with multiple payment methods.
  - **Order Confirmation:** Real-time order status and confirmation for users.

Backend with Sanity CMS:

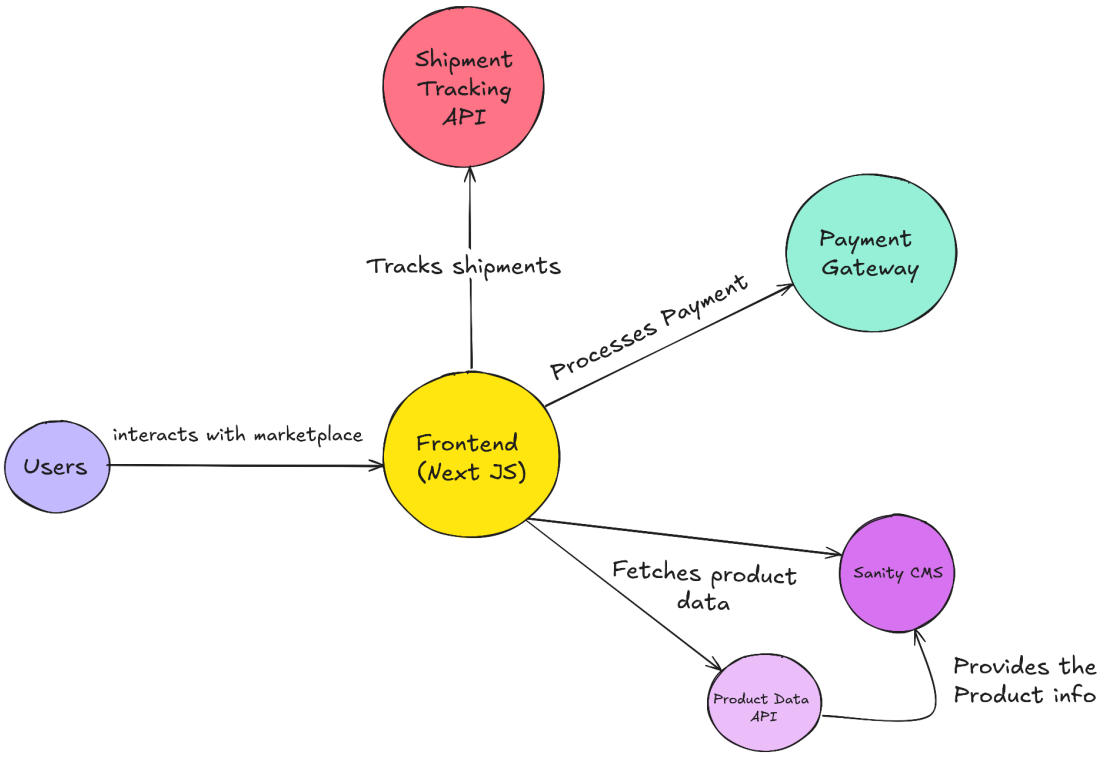
- **Product Data Management:**
  - I will use **Sanity CMS** to manage and update product data dynamically. The CMS will house product details such as names, prices, stock levels, images, and descriptions.
- **Customer Data Management:**
  - **Sanity CMS** will store customer information, including order history, shipping addresses, and preferences.
- **Order Management:**
  - Orders will be tracked and managed using **Sanity CMS**, ensuring that stock levels are automatically updated when orders are placed.

Third-Party APIs:

- **Shipment Tracking:**
  - Integrate APIs to allow customers to track their shipments in real-time.
- **Payment Gateways:**
  - I'll use secure payment APIs like Stripe to handle transactions, offering various payment options (credit/debit cards, wallets, etc.).
- **Email Notifications:**
  - Use third-party APIs to send order confirmations, shipping updates, and promotional emails.

2. System Architecture

I have developed a system architecture that outlines how all the components interact to provide an optimal user experience. Below is a high-level view of the architecture:



Data Flow:

1. **Product Browsing:**
   - Users browse the homepage and product listings, with product details being fetched dynamically from **Sanity CMS** via the **Product Data API**.
2. **Order Placement:**

    - Once users add products to their cart and proceed to checkout, the order is processed and stored in **Sanity CMS**.
3. **Shipment Tracking:**
    - After the order is placed, users can track the shipment status via a third-party API.
4. **Payment Processing:**
    - Payments are securely processed through the **Payment Gateway**, and order statuses are updated accordingly in **Sanity CMS**.

3. API Requirements:

To implement this architecture, I will define API endpoints for product retrieval, order management, and payment processing.

Endpoint Name	Method	Description	Response Example
/products	GET	Fetch all available product details.	{ "id": 1, "name": "Product A", "price": 100, "stock": 50 }
/orders	POST	Create a new order.	{ "id": 1, "name": "Product A", "price": 100, "stock": 50 } order. { "orderId": 12345, "status": "Pending" }
/shipment	GET	Retrieve shipment tracking details.	{ "paymentId": 6789, "status": "Success" }{ "orderId": 12345, "status": "In Transit", "ETA": "2 days" }
/payment	POST	Process payment and confirm.	{ "paymentId": 6789, "statusE": "Success" }

4. Key Workflows:

To ensure a smooth experience for users and efficient operations, the following key workflows have been identified:

1. **User Registration:**
    - A user signs up, and their details are stored in **Sanity CMS**.
    - A confirmation email is sent automatically upon registration.
2. **Product Browsing:**
    - Users explore the homepage and product listings.
    - The frontend requests product data from **Sanity CMS** via the **Product Data API**.

Prepared By Mariyam Asif

3. **Order Placement:**
    - Users add items to their cart and proceed to checkout.
    - Order details are stored in **Sanity CMS**, and payment is processed.
4. **Shipment Tracking:**
    - Once the order is placed, users can track the delivery status via the **Shipment Tracking API**.
    - Shipment updates are reflected in real-time.

Conclusion:

In conclusion, this technical foundation outlines the essential elements required to build a user-centric e-commerce marketplace. By leveraging a combination of frontend design, backend data management with Sanity CMS, and seamless third-party API integrations, the platform will offer an efficient and secure shopping experience. The proposed architecture and workflows ensure smooth product browsing, order processing, and shipment tracking, positioning the marketplace for success in providing users with high-quality products and reliable services.