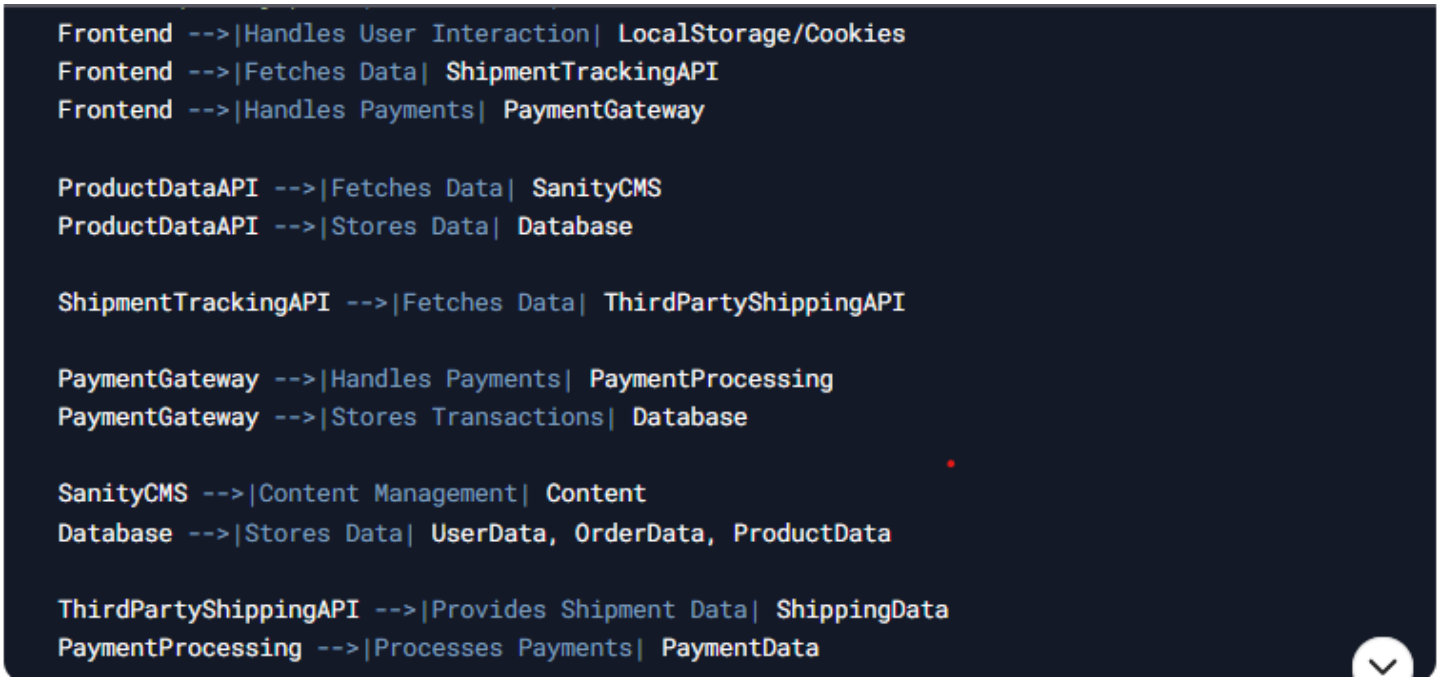
Hackathon:Day-2

System Architecture



1. Frontend (Next.js)

- **Responsibilities:**
  - Render the user interface.
  - Handle user interactions (e.g., adding products to cart, navigating between pages).
  - Fetch data from APIs (e.g., product data, shipment tracking, payment status).
  - Manage local state (e.g., cart, authentication tokens) using local storage or cookies.

- **Components:**
  - Pages (e.g., Home, Product Details, Cart, Checkout, Login, Contact Us).
  - Components (e.g., Header, Footer, Product Cards, Cart Summary).
  - API Routes (e.g., Server-Side Rendering, Data Fetching).

2. Sanity CMS

- **Responsibilities:**
  - Content Management System (CMS) for managing product data, blog posts, and other content.
  - Provides a rich text editor and media library for non-technical users to manage content.

- **Integration:**
  - Exposes a GraphQL API for fetching content.
  - Can be used to manage product catalogs, categories, descriptions, images, etc.

3. Product Data API

- **Responsibilities:**
  - Exposes RESTful or GraphQL APIs for the frontend to fetch product data.
  - Handles CRUD operations for products, categories, and other related data.
  - Caches data for faster retrieval.
- **Integration:**
  - Fetches data from Sanity CMS via its GraphQL API.
  - Stores and retrieves data from the database (if needed).
  - Can be hosted on a backend service like Vercel, Netlify, or AWS Lambda.

4. Third-Party API (Shipment Tracking)

- **Responsibilities:**
  - Provides shipment tracking information for orders.
  - Integrates with shipping carriers (e.g., FedEx, UPS, DHL) to fetch real-time tracking data.
- **Integration:**
  - The frontend can call this API to get tracking information for an order.
  - The Shipment Tracking API can be a custom service that aggregates data from multiple shipping providers.

5. Payment Gateway

- **Responsibilities:**
  - Handles payment processing (e.g., credit card payments, PayPal, Stripe).
  - Provides secure payment forms and APIs.
  - Manages payment transactions, refunds, and disputes.
- **Integration:**
  - The frontend can redirect users to the payment gateway for checkout.
  - The Payment API can be a custom service that interacts with the payment gateway's API to process payments.

6. Database

- **Responsibilities:**
  - Stores user data (e.g., profiles, addresses, payment methods).
  - Stores order data (e.g., order history, shipment details).
  - Stores product data (if not managed entirely by Sanity CMS).

- **Types:**
  - Relational Database (e.g., PostgreSQL, MySQL) for structured data.
  - NoSQL Database (e.g., MongoDB) for unstructured data.
  - Cloud-based databases (e.g., Firebase, AWS DynamoDB) for scalability.

7. Other Components

- **LocalStorage/Cookies:**
  - Stores session data, cart items, and other client-side data.
- **Content:**
  - Managed by Sanity CMS (e.g., product descriptions, blog posts, FAQs).
- **ShippingData:**
  - Provided by third-party shipping APIs (e.g., FedEx, UPS).
- **PaymentData:**
  - Managed by the payment gateway (e.g., Stripe, PayPal).

Summary

- **Frontend (Next.js):** Handles user interface and interactions.
- **Sanity CMS:** Manages content (products, blog posts, etc.).
- **Product Data API:** Fetches and manages product data from Sanity CMS.
- **Third-Party API (Shipment Tracking):** Provides shipment tracking information.
- **Payment Gateway:** Handles payment processing and transactions.
- **Database:** Stores user, order, and product data.