DAY -2:

Technical Foundation Report for "General E-commerce Marketplace "

1. Technical Plan Aligned with Business Goals:

Objective:

The goal is to create a General E-commerce marketplace that supports typical workflows such as product browsing, cart management, and order placement while also ensuring scalability, security, and an excellent user experience.

Features for General E-commerce:

- **Product Browsing:** Customers can browse through categories, filters, and product details.
- **Cart Management:** Users can add, remove, and modify the contents of their cart.
- **Order Placement:** A seamless checkout process, including payment integration, shipping, and order confirmation.
- **Customer Accounts:** Users can create accounts to track orders, manage preferences, and save billing/shipping information.

Marketplace Goals:

- Enhance user experience with responsive and intuitive design.
- Optimize inventory management and real-time updates.
- Enable secure and seamless payment and checkout flows.

2. System Architecture Visualized:

Frontend Requirements:

User-friendly interface for browsing products.

Responsive design (mobile & desktop).

Essential pages: Home, Product Listing, Product Details, Cart, Checkout, Order Confirmation.

Sanity CMS as Backend:

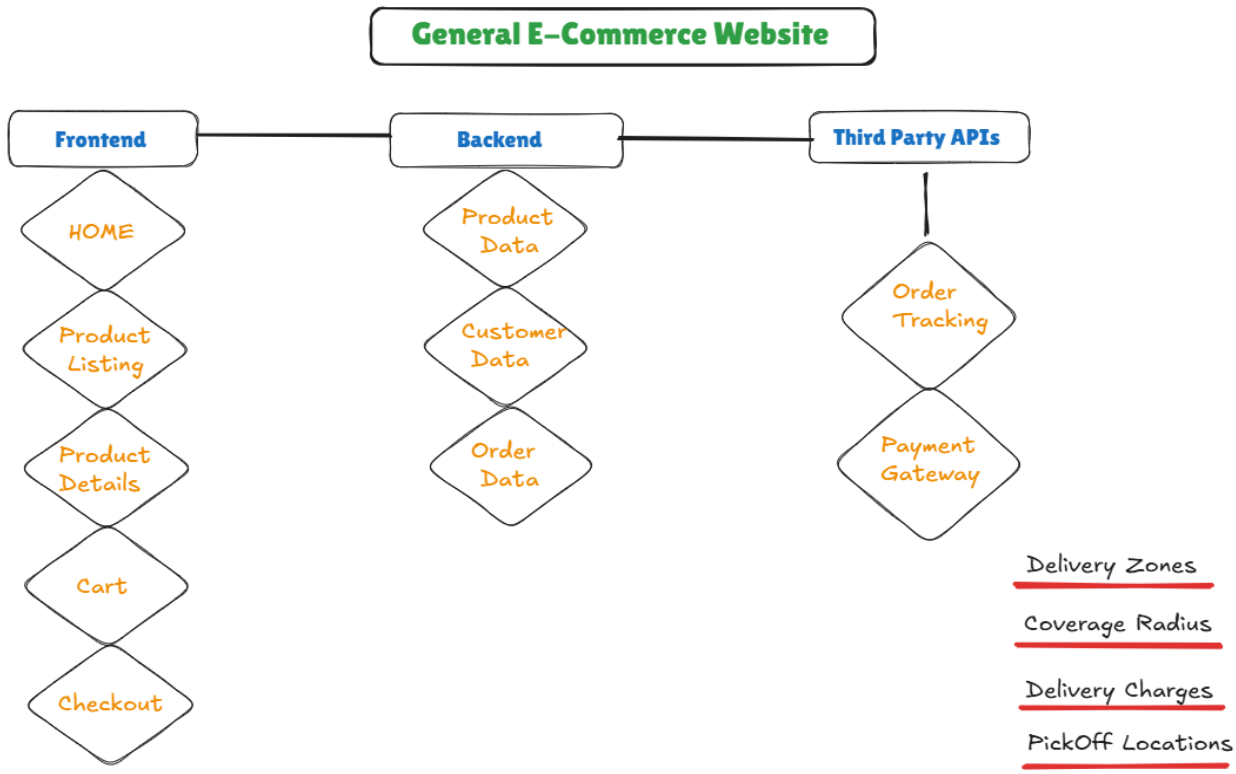
Use Sanity CMS to manage product data, customer details, and orders.

Design schemas in Sanity aligned with business goals.

Third-Party APIs:

Integrate APIs for shipment tracking, payment gateways, and other services.

Ensure APIs provide data needed for frontend functionality.



3.Plan API Requirements:

List of Endpoints and Methods:

Endpoint Name	Method	Description	Payload/Response
`/products`	GET	Fetch all products	**Response:** `{ ID, name, price, stock, image }`
`/orders`	POST	Create a new order	**Payload:** `{ customerId, productIds[], totalAmount, shippingAddress, paymentStatus, orderDate }`
			Response: `{ orderId, status, customerId, productDetails[], totalAmount, shippingAddress }`
`/shipment`	GET	Track order shipment status	**Response:** `{ shipmentId, orderId, status, expectedDeliveryDate, trackingUrl }`

4. Sanity Schemas Drafted:

~Product Schema:

This schema is used to define the structure of a **Product** in the marketplace. It includes essential details about a product such as its name, description, price, category, and images.

Fields:

- **name**: The name of the product (text).
- **price**: The cost of the product (number).
- **description**: A detailed description of the product (text).
- **category**: The category to which the product belongs, such as Electronics, Apparel, etc. (text).
- **image**: An image of the product (image file).

~Order Schema:

This schema tracks **Orders** placed by customers. It stores information like the order ID, the products ordered, the total price, and the order status.

Fields:

- **orderId**: Unique ID for the order (text).
- **status**: The current status of the order, such as 'Pending,' 'Shipped,' or 'Delivered' (text).
- **totalAmount**: Total price for all items in the order (number).
- **items**: List of products included in the order. Each item will include the product and quantity.
- **orderDate**: The date and time when the order was placed (datetime).

~Customer Schema:

The **Customer** schema defines the structure for storing customer data. It helps track customer information for order processing, account management, and personalized experiences.

Fields:

- **firstName**: The customer's first name (text).
- **lastName**: The customer's last name (text).
- **email**: The customer's email address (text).
- **phone**: The customer's phone number (text).
- **shippingAddress**: Address for shipping the order, including street, city, postal code, and country (object with multiple fields).

~Cart Schema:

The **Cart** schema represents the shopping cart where customers store products before checking out. It holds product details and calculates the total cost of the cart.

Fields:

- **customer**: Reference to the customer who owns the cart.
- **items**: A list of products the customer has added to the cart. Each item has the product and quantity.

- **total**: The total cost of all items in the cart (number).

Conclusion:

To build an effective e-commerce platform, we focus on creating a user-friendly frontend with a responsive design that caters to both mobile and desktop users, ensuring smooth navigation through essential pages like product listing, cart, and checkout. The backend is managed through Sanity CMS, which allows us to easily organize and manage key data like products, customers, and orders through well-defined schemas. Additionally, integrating third-party APIs for shipment tracking, payment gateways, and other necessary services will enhance the functionality and user experience. This approach aligns business goals with technical execution, providing a solid foundation for a scalable and efficient e-commerce platform.