DAY 2 PLANNING THE TECHNICAL FOUNDATION FASTFOOD

Technical Requirements:

1. User-Friendly Interface:The website should be easy to navigate so customers can browse food items effortlessly.
2. Responsive Design:It should look great and work well on both mobile and desktop devices.
3. Essential Pages:

- **Home Page**: This will show popular dishes or offers right upfront.
- **Product Listing Page**: All available food items will be displayed here with prices and
- **images.Product Details Page:** Detailed information about each food item, including ingredients, price, and options (e.g., add-ons).
- **Cart Page:** A page to review and manage the selected food items before checkout.
- **Checkout Page**: A place to finalize the order and make the payment.
- **Order Confirmation Page**: A page showing the order status and confirmation details.

Backend Requirements with Sanity CMS

Sanity CMS for Data Management:

- Sanity CMS will act as our database to store and manage everything:
- Food items with their details like price, category, and images.
- Customer information like names, addresses, and order histories.
- Orders, including their statuses (e.g., "Pending," "Delivered").

Schemas in Sanity:

We'll design schemas in Sanity to align perfectly with our business goals, making it easy to add, update, or remove data.

Third-Party APIs

To make the website functional and dynamic, we'll integrate external APIs:

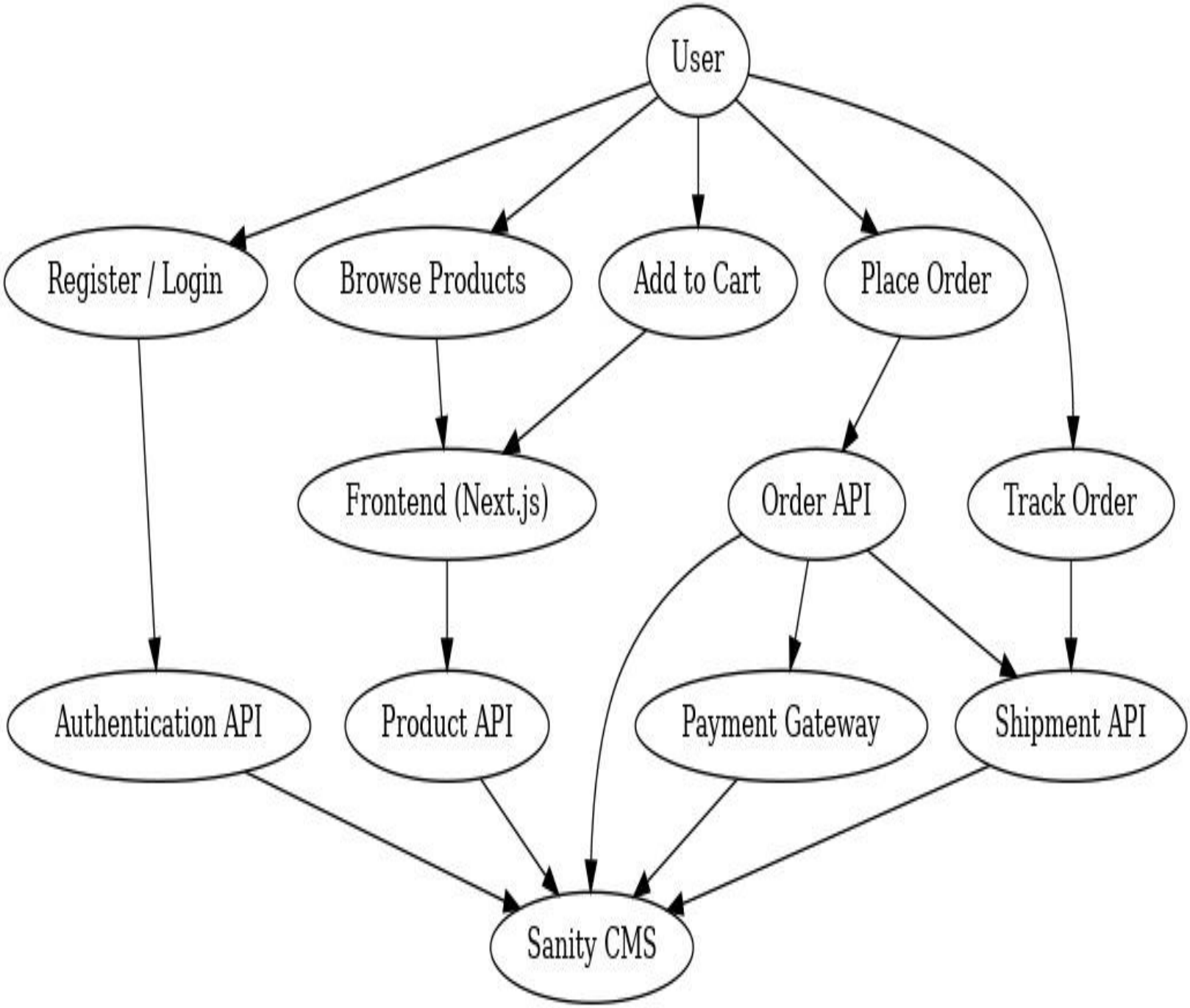
1. Payment Gateways:

These will handle secure online payments, making transactions quick and reliable.

2. Shipment Tracking APIs:

These will track the delivery progress in real time, ensuring customers know when their food is on its way.

2)Design System Architecture



Data Flow and Workflows

1. User Registration:

- The user signs up on the frontend.
  The data is stored in Sanity CMS, and a confirmation message is displayed to the user.

2. Product Browsing:

- When the user visits the menu or product listing page:
- The frontend requests product data from Sanity CMS via APIs.
- Sanity sends the data, and it's displayed dynamically on the frontend.

3. Order Placement:

- The user adds items to the cart and proceeds to checkout.
- Once the order is placed, details are sent to Sanity CMS and recorded in the database.

4. Shipment Tracking:

- Shipment information is fetched from a Third-Party API and displayed on the user's order status page in real-time.

5. Payment Processing:

- During checkout, the Payment Gateway processes the user's payment securely.
- A confirmation of payment is sent back to the frontend and recorded in Sanity CMS.

6. Adding Products to Cart

- The user selects a product and clicks "Add to Cart."
- The frontend stores the cart data temporarily (e.g., in local storage or state management).
- On proceeding to checkout, the cart details are sent to the backend.

7. Tracking an Order

- The user clicks on "Track Order."
- The frontend calls the /shipment/{orderId} API.
- Shipment details are fetched from the third-party tracking API and displayed to the user.

8. Payment Processing

- User completes the checkout and proceeds to payment.
- Payment details are securely sent to the Payment Gateway.
- Once confirmed, the order status is updated in the backend.

Placing an Order

- User reviews the cart and confirms the order.
- The frontend sends the order details (products, customer info, and payment status) to the /orders API.
- Sanity CMS stores the order, and a confirmation is sent to the user.

3) API ENDPOINTS

ENDPOINT	METHOD	PURPOSE	RESPONSE EXAMPLE
/products	**GET**	**Fetches all available products.**	**[{"id":1,"name":"Burger","price":599,"stock":50}]**
/products/{id}	**GET**	**Fetches details for a specific product.**	**{"id":1,"name":"Burger","price":599,"description":"Tasty burger with fries"}**
/orders	**POST**	**Creates a new order.**	**{"orderId":101,"status":"Order Placed","ETA":"30 mins"}**
/shipment/{orderId}	**GET**	**Fetches real-time order tracking.**	**{"shipmentId":456,"status":"In Transit","expectedDelivery":"15 mins"}**

4) Data Schema Design

Entities

Products

Fields:

id, name, description, price, stock, category, image

Orders

Fields:

orderId, customerName, contactInfo, address, items, totalAmount, paymentStatus

Shipment

Fields:

shipmentId, orderId, status, expectedDelivery

Customers

Fields:

customerId, name, contactInfo, address, orderHistory