DAY 2

Technology Stack Overview

- **Next.js**: A React framework for server-side rendering (SSR), static site generation (SSG), and dynamic routing. It improves performance and SEO.
- **HTML/CSS**: HTML structures the content, while CSS (with Tailwind CSS) handles styling and responsiveness for a clean, mobile-first design.
- **JavaScript/TypeScript**: JavaScript adds interactivity, and TypeScript provides type safety, improving code maintainability.
- **Tailwind CSS**: A utility-first CSS framework for rapid custom styling without writing extensive custom CSS.
- **Sanity CMS**: A headless CMS used for managing product, customer, and order data, allowing for easy content updates.
- **Payment APIs (Stripe)**: Used for secure payment processing during checkout.
- **Shipment Tracking APIs**: External APIs (like UPS or FedEx) to track and display shipment status for orders.

Technical Requirements:

---

1. Frontend:

- **User-Friendly Interface**: The website displays products in a grid layout with essential details like name, price, and image. Features include hover effects, product zoom, and smooth interactions.
- **Responsive Design**: Fully optimized for mobile, tablet, and desktop devices using a mobile-first approach and Tailwind CSS.
- **Key Pages**:
  - **Home**: Includes hero section, menu, carousel, customer reviews, and about section.
  - **Shop**: Displays products with search, filtering (by category, price), and sorting (e.g., best sellers).
  - **Product Details**: Features product description, quantity control, add-to-cart functionality, and customer reviews.
  - **Menu**: Showcases the full menu of available items. Users can browse food options, view descriptions, and explore categories.
  - **Blog:** Contains all the food and healthy-diet related blogs
  - **Cart**: Displays selected items and links to checkout.
  - **Checkout**: Handles user payment and order details.

  - **Order Confirmation**: Shows order summary and shipping info.
  - **Sign up**: Allows new users to create an account by entering basic details like name, email, and password for easier checkout and order management.

---

2. Backend:

- **Sanity Schemas:**

1. Product Schema (Food Schema)

- **Food Name**: Stores the name of the food item.
- **Slug**: A URL-friendly version of the food's name.
- **Category**: Specifies the category of the food item (e.g., Burger, Sandwich, Drink).
- **Current Price**: The price of the food item.
- **Original Price**: The original price of the food item before any discounts.
- **Food Image**: An image of the food item.
- **Description**: A brief description of the food item.
- **Available**: Boolean field indicating if the food item is available for purchase.

2. Chef Schema

- **Chef Name**: Stores the full name of the chef.
- **Slug**: A URL-friendly version of the chef's name for easier linking.
- **Experience**: Number of years the chef has been working in the culinary field.
- **Expertise**: The chef's area of culinary expertise (e.g., Italian cuisine, Pastry, etc.).
- **Chef Image**: An image of the chef (typically a professional photo).
- **Bio**: A short biography of the chef, including their background, career highlights, and achievements.

3. Customer Schema

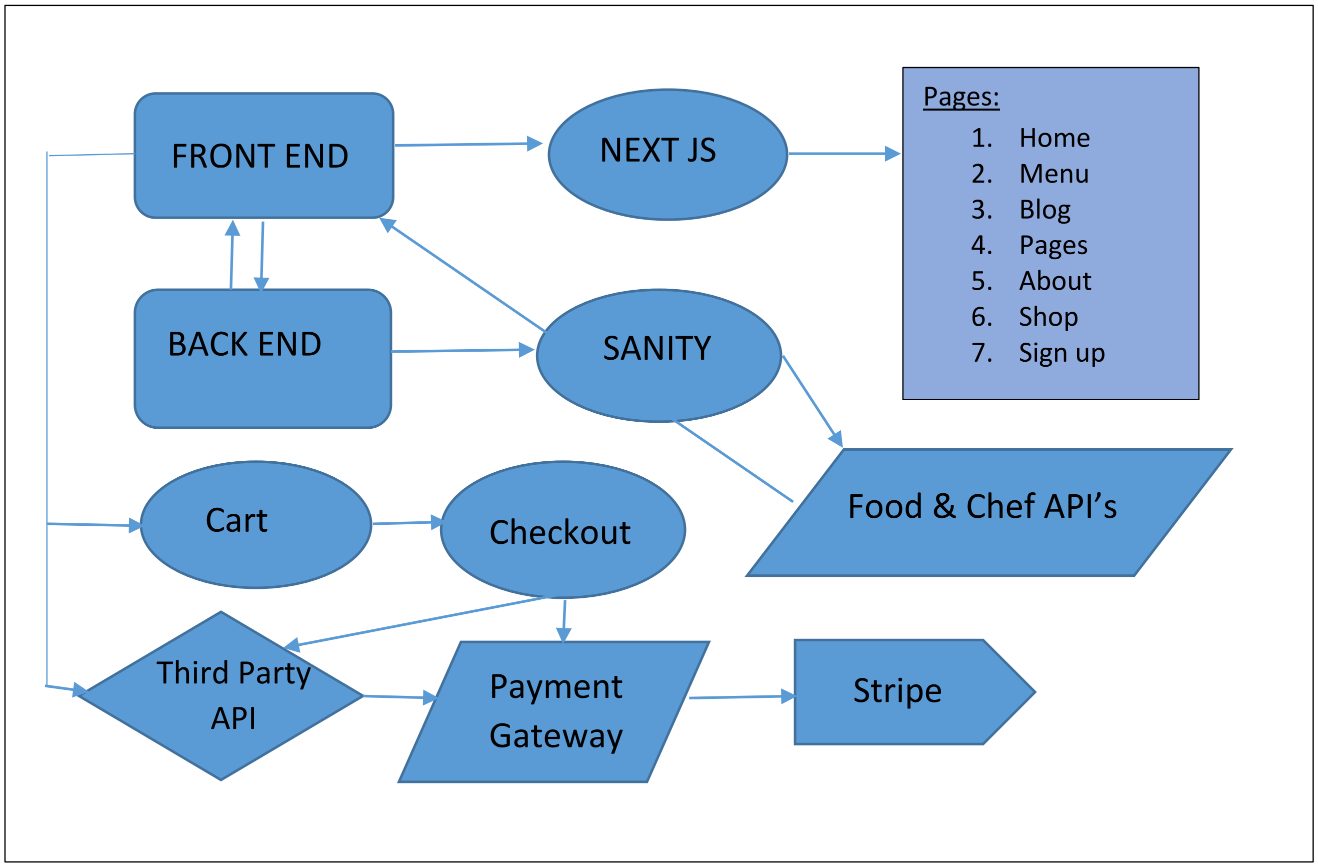
- **First Name**: Stores the customer's first name.
- **Last Name**: Stores the customer's last name.
- **Email**: Stores the customer's email address for communication.
- **Phone Number**: Stores the customer's phone number.
- **Shipping Address**: Stores the customer's shipping address (including street, city, state, zip code).
- **Billing Address**: Stores the customer's billing address (similar to shipping address).
- **Order History**: Stores a list of previous orders made by the customer, including order details and statuses.

---

3. Third-Party Integrations

- **Payment APIs (Stripe)**: Stripe is used to process payments securely during the checkout process. This APIs handle credit card processing, and transaction details.

System Architecture:



API Endpoints

---

1. Endpoint: /products

- **Method**: GET
- **Description**: Fetch all available products from the Sanity CMS.
- **Response**: Returns detailed information about a specific product, including name, price, description, image, and availability.

Response Example:

```
{
  "id": "123",
  "name": "Country Burger",
  "price": 9.99,
  "originalPrice": 12.99,
  "category": "Burger",
  "image": "https://example.com/images/cheese-burger.jpg",
  "description": "A delicious cheesy burger with fresh veggies.",
  "available": true
}
```

2. Endpoint: /orders

- **Method**: POST
- **Description**: Create a new order in Sanity when a customer completes the checkout.
- **Payload**: Customer information, product details, payment status, and delivery address.

Response Example:

{ "orderId": "5678",
 "status": "Order Created",
 "message": "Your order has been created successfully." }

3. Endpoint: /shipment/:

- **Method**: GET
- **Description**: Track the status of an order via a third-party API (e.g., shipment tracking).

Response Example:

```
{
  "orderId": "5678",
  "shipmentId": "12345",
  "status": "Shipped",
  "eta": "2025-02-10T15:00:00Z",
  "currentLocation": "Chicago, IL"
}
```

4. Endpoint: /payment

- **Method**: POST
- **Description**: Process the payment for an order.
- **Payload**: Payment details (e.g., payment method, transaction ID).

Response Example:

```
{
  "orderId": "5678",
  "paymentMethod": "Credit Card",
  "transactionId": "abc123xyz",
  "amount": 19.98
}
```

5. Endpoint: /chefs

- **Method**: GET
- **Description**: Fetch all chefs' details from Sanity.
- **Response**: A list of chefs with details such as name, image, experience, expertise.

  Response Example:

  [{ "id": "1",
  "name": "Chef John Doe",
  "experience": "10 years",
  "expertise": "Italian Cuisine",
  "image": "https://example.com/images/chef-john.jpg" },{}…]

SANITY SCHEMA / EXAMPLE:

```
// src/sanity/schemaTypes/foods.ts

const foodSchema = {
  name: 'food',
  type: 'document',
  title: 'Food',
  fields: [
    {
      name: 'name',
      type: 'string',
      title: 'Food Name',
    },

    {
      name: 'slug',
      type: 'slug',
      title: 'Slug',
      options: {
        source: 'name',
        maxLength: 96,
      },
    },

    {
      name: 'category',
      type: 'string',
      title: 'Category',
      description:
        'Category of the food item (e.g., Burger, Sandwich, Drink, etc.)',
    },
    {
      name: 'price',
      type: 'number',
      title: 'Current Price',
    },
    {
      name: 'originalPrice',
      type: 'number',
      title: 'Original Price',
      description: 'Price before discount (if any)',
    },
    {
      name: 'image',
      type: 'image',
      title: 'Food Image',
      options: {
        hotspot: true,
      },
    },
    {
      name: 'description',
      type: 'text',
      title: 'Description',
      description: 'Short description of the food item',
    },
    {
      name: 'available',
      type: 'boolean',
      title: 'Available',
      description: 'Availability status of the food item',
    },
  ],
};

export default foodSchema;
```