Syeda Nausheen Raza Zaidi (GIAIC Q2 Student Saturday 9-12)

Roll No: 00284652

# Marketplace Technical Foundation – Fork & Feast

## Marketplace Type: QCommerce

## 1) System Architecture Overview:

### Diagram:

### Components:

- **Frontend (Next.js):**
  - The user interface where customers can browse food items, add them to the cart, and place orders.
  - Pages: Home, Menu, Product Details, Cart, Checkout, Order Confirmation.
  - Responsive design for mobile and desktop users.
- **Backend (Sanity CMS):**
  - **Product Management:** Store and manage food items with categories, pricing, and availability.
  - **Order Management:** Store customer orders, order status, and payment details.
  - **Customer Information:** Although, clerk handles the authentication but still store customer information like his name, phone number, email in sanity.
- **Authentication (Clerk):**
  - Handle user authentication (sign-up, login, password reset).
  - Secure user data and provide session management.
- **Third-Party APIs:**
  - **ShipEngine API:** For real-time shipment tracking and delivery updates.
  - **Payment Gateway API:** For processing payments (e.g., Stripe, PayPal).

- **Other APIs:** For additional services like SMS notifications or email confirmations.

## Workflow:

1. User logs in or registers (via Clerk Auth).
2. User browses food items (data fetched from Sanity CMS using GROQ or GET API).
3. User places an order, stored in Sanity CMS (POST API).
4. Real-time order tracking is powered by ShipEngine (GET API).
5. Payment is processed via a secure payment gateway e.g: Stripe (POST API).

# 2) Key Workflows:

- **User Registration & Authentication:**
  I. User signs up using Clerk → User data is stored in Clerk → Confirmation sent to the user.
  II. User logs in → Clerk manages session → User accesses their dashboard.
- **Product Browsing:**
  I. User visits the website → Frontend fetches product data from Sanity CMS → Products are displayed dynamically.
  II. User can filter by category (e.g., vegetarian, non-vegetarian) or search for specific items.
- **Order Placement:**
  I. User adds items to the cart → Proceeds to checkout → Order details are sent to Sanity CMS.
  II. Payment is processed via a payment gateway API → Payment status is recorded in Sanity.
- **Shipment Tracking:**
  I. After order confirmation, ShipEngine API is called to generate a shipment tracking ID.
  II. Real-time tracking updates are fetched from ShipEngine and displayed to the user.

- **Order Management:**
  I. Admin can view and manage orders in Sanity CMS.
  II. Admin updates order status (e.g., preparing, out for delivery, delivered).

## 3) Category-Specific Instructions:

### Q-Commerce:

- **Real-Time Inventory Updates:**
  I. Ensure that product availability is updated in real-time to avoid overselling.
  II. Example: If a product is out of stock, it should be immediately reflected on the frontend.

- **Express Delivery Workflows:**
  I. Integrate ShipEngine API for real-time delivery tracking.
  II. Example endpoint: /express-delivery-status to fetch real-time tracking updates.

- **SLA (Service Level Agreement) Tracking:**
  I. Track delivery times and ensure orders are delivered within the promised time frame.
  II. Example: Display estimated delivery time (e.g., "Delivery in 30 minutes") on the checkout page.

# 4)API Endpoints:

## Food Item Management:

### Get All Food Items:

- **Endpoint:** /products
- **Method:** GET
- **Purpose:** Fetch all available food items from Sanity.
- **Response Example:**


```
[
  {
    "id": 1,
    "name": "Pizza Margherita",
    "price": 10.99,
    "category": "Vegetarian",
    "stock": 50,
    "desc": “text”
  },
 {
    "id": 2,
    "name": "Biryani ",
    "price": 10.99,
    "category": "Desi",
    "stock": 50,
    "desc": “text”
  }
]
```

## Get Food item By Id:

- **Endpoint:** /product
- **Method:** GET
- **Purpose:** Fetch a single food item from Sanity.
- **Response Example:**

```
[
 {
  "id": 1,
  "name": "Pizza Margherita",
  "price": 10.99,
  "category": "Vegetarian",
  "stock": 50,
  "desc": “text”
 }
]
```

## Order Management:

### Post a new Order:

- **Endpoint:** /order
- **Method:** POST
- **Purpose:** Create a new order in Sanity.
- **Payload Example:**

```
{
    "customerId":  "123",
    "items":  [
        {
            "productId":  1,
            "quantity":  2
        },
       {
            "productId":  2,
            "quantity":  1
        }
    ],
    "totalAmount":  41.98,
    "paymentStatus":  "Paid"
```

```
}
```

## Shipment Tracking:

### Track the order in real time:

- **Endpoint:** /shipment
- **Method:** GET
- **Purpose:** Track order status via ShipEngine API.
- **Response Example:**

```
{
  "orderId": 123,
  "status": "Out for Delivery",
  "ETA": "15 mins"
}
```

## User Authentication (Clerk):

### User Signs up:

- **Endpoint:** /auth/signup
- **Method:** POST
- **Purpose:** Register a new user.
- **Payload Example:**

```
{
  "email": "user@example.com",
  "password": "password123"
}
```

### User Logins:

Endpoint: /auth/login

- **Method:** POST
- **Purpose:** Authenticate an existing user and return a JWT token for session management.
- **Payload Example:**

```
{
  "email": "user@example.com",
  "password": "password123"
}
```

- **Response Example (Success):**

{
 "status": "success",
 "message": "User logged in successfully",
 "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOiIxMjM0NTY3ODkwIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c",
 "user": {
  "id": "1234567890",
  "email": "user@example.com",
  "name": "John Doe"
 }
}

- **Response Example (Failure - Invalid Credentials):**

{
  "status": "error",
  "message": "Invalid email or password"
}

- **Response Example (Failure - User Not Found):**

{
  "status": "error",
  "message": "User not found"
}

# 4) Sanity Data Schemas:

## Food Item Schema:

```
export default {
  name: 'item',
  type: 'document',
  fields: [
    { name: 'id', type: 'string', title: 'Product ID' },
    { name: 'name', type: 'string', title: 'Product Name' },
    { name: 'price', type: 'number', title: 'Price' },
    { name: 'category', type: 'string', title: 'Category' },
    { name: 'stock', type: 'number', title: 'Stock Level' },
    { name: 'mainImage', type: 'image', title: 'Main Product Image' },
    {
    name: 'images',
      type: 'array',
      title: 'Food Item Images',
      of: [{ type: 'image' }] // Array of images for the detail page
    }
  ]
};
```

## Order Schema:

```
export default {
  name: 'order',
  type: 'document',
  fields: [
    { name: 'orderId', type: 'string', title: 'Order ID' },
    { name: 'customerId', type: 'string', title: 'Customer ID' },
    {
      name: 'items',
      type: 'array',
      title: 'Order Items',
      of: [{ type: 'reference', to: [{ type: 'item' }] }]
    },
```

```
    { name: 'totalAmount', type: 'number', title: 'Total Amount' },
    { name: 'paymentStatus', type: 'string', title: 'Payment Status' },
    { name: 'orderStatus', type: 'string', title: 'Order Status' },
    {
      name: 'shipmentDetails',
      type: 'object',
      title: 'Shipment Details',
      fields: [
        { name: 'trackingId', type: 'string', title: 'Tracking ID' },
        { name: 'carrier', type: 'string', title: 'Carrier' },
        { name: 'status', type: 'string', title: 'Shipment Status' },
        { name: 'estimatedDelivery', type: 'datetime', title: 'Estimated Delivery' }
      ]
    },
    {
      name: 'paymentDetails',
      type: 'object',
      title: 'Payment Details',
      fields: [
        { name: 'paymentId', type: 'string', title: 'Payment ID' },
        { name: 'method', type: 'string', title: 'Payment Method' },
        { name: 'status', type: 'string', title: 'Payment Status' },
        { name: 'amount', type: 'number', title: 'Amount Paid' }
      ]
    }
  ]
};
```

## Customer Schema:

```
export default {
  name: 'customer',
  type: 'document',
  title: 'Customer',
  fields: [
    {
      name: 'clerkUserId',
      type: 'string',
      title: 'Clerk User ID',
```

```
    description: 'Unique ID from Clerk to link authentication with customer profile.',
   },
   {
    name: 'name',
    type: 'string',
    title: 'Full Name',
   },
   {
    name: 'email',
    type: 'string',
    title: 'Email',
   },
   {
    name: 'phone',
    type: 'string',
    title: 'Phone Number',
   },
   {
    name: 'address',
    type: 'object',
    title: 'Address',
    fields: [
     { name: 'street', type: 'string', title: 'Street' },
     { name: 'city', type: 'string', title: 'City' },
     { name: 'state', type: 'string', title: 'State' },
     { name: 'zipCode', type: 'string', title: 'Zip Code' },
    ],
   },
   {
    name: 'orderHistory',
    type: 'array',
    title: 'Order History',
    of: [{ type: 'reference', to: [{ type: 'order' }] }],
   },
  ],
};
```

Syeda Nausheen Raza Zaidi (GIAIC Q2 Student Saturday 9-12)

Roll No: 00284652