Marketplace Technical Foundation – Customized & International Cuisine Delivery Q-Commerce

Here's the breakdown for Day 2's tasks, tailored for **Customized & International Cuisine Delivery Q-Commerce**, the customized international cuisine delivery platform:

1. Technical Requirements

Frontend Requirements

- **User-Friendly Interface:** Easy navigation to browse and customize menus.
- **Responsive Design:** Seamless experience across mobile and desktop.
- **Essential Pages:**
  - Home
  - Product Listings (Cuisines)
  - Product Details (Dish-specific info, customization options)
  - Cart
  - Checkout
  - Order Confirmation

Sanity CMS as Backend

- **Purpose:** Manage data for cuisines, customer details, and order records.
- **Schemas:**
  - **Products Schema:** Cuisine type, customization options, pricing, stock.
  - **Orders Schema:** Customer details, event type, ordered dishes, delivery info.
  - **Customers Schema:** Name, contact info, order history.

Third-Party APIs

- **Payment Gateway:** Securely process payments (e.g., Stripe, PayPal).
- **Shipment Tracking API:** Display real-time delivery updates.
- **Geolocation API:** Optimize delivery zones and times.

2. System Architecture

Overview: High-level system flow for **Customized & International Cuisine Delivery Q-Commerce**.

Data Flow Workflow:

1. A user visits the marketplace frontend to browse cuisines.
2. The frontend makes a request to the Product Data API (powered by Sanity CMS) to fetch product listings and customization options, which are displayed dynamically on the site.
3. When the user places an order, the order details are sent to Sanity CMS via an API request, where the order is recorded.
4. Shipment tracking information is fetched through a third-party API and displayed to the user in real time.
5. Payment details are securely processed through the Payment Gateway, and a confirmation is sent to the user and recorded in Sanity CMS.

This workflow ensures seamless data flow between frontend interactions, Sanity CMS for content management, and third-party APIs for logistics and transactions.

3. Key Workflows

1. User Registration:

- User signs up → Data stored in Sanity CMS → Confirmation sent to the user.

2. Product Browsing:

- User views product categories → Sanity API fetches product data → Products displayed dynamically on the frontend.

3. Order Placement:

- User adds items to the cart → Proceeds to checkout → Order details saved in Sanity CMS.

4. Shipment Tracking:

- Shipment status updates fetched via the shipment API → Displayed to the user in real-time.

3. API Requirements

Endpoints:

- [ ] **/products**
  - **Method:** GET
  - **Description:** Fetch available cuisines and customization options.
  - **Response:** { "id": 1, "name": "Sushi Platter", "price": 50, "stock": 20 }
- [ ] **/orders**
  - **Method:** POST
  - **Description:** Create a new order with customer and event details.
  - **Payload:**

```
{
  "customerId": 101,
  "eventType": "Wedding",
  "orderItems": [{ "productId": 1, "quantity": 2 }],
  "paymentStatus": "Paid"
}
```

  - **Response:** { "orderId": 123, "status": "Success" }
- [ ] **/shipment**
  - **Method:** GET
  - **Description:** Fetch real-time delivery status.
  - **Response:** { "shipmentId": 456, "status": "On the way", "ETA": "15 mins" }

4. Sanity Schema Examples

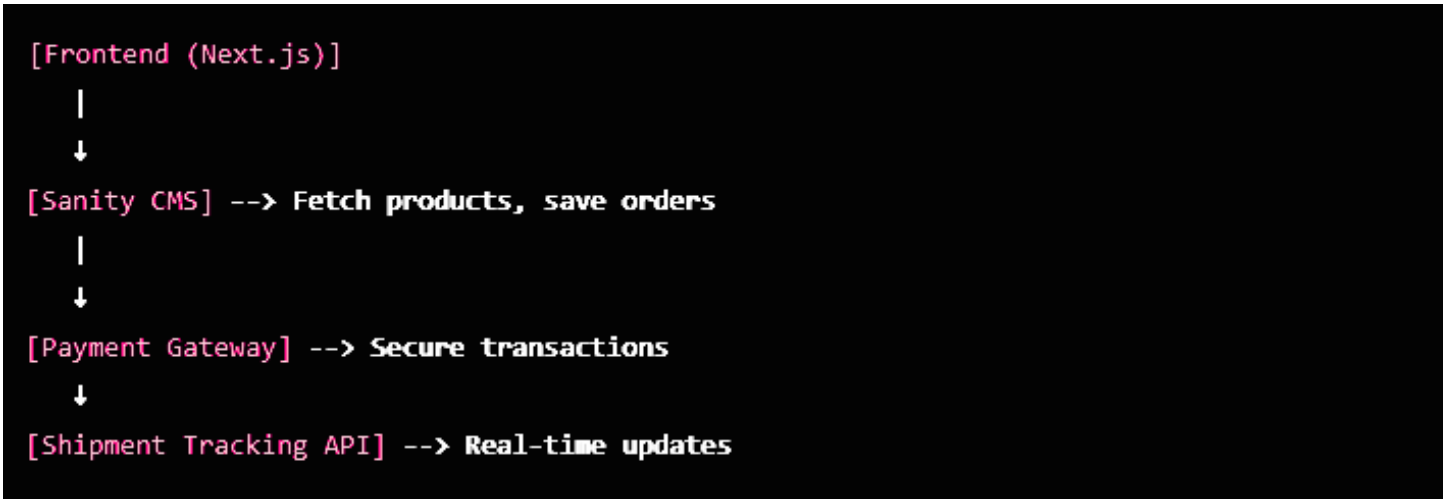
Products Schema

```
export default {
  name: 'product',
  type: 'document',
  fields: [
    { name: 'name', type: 'string', title: 'Dish Name' },
    { name: 'price', type: 'number', title: 'Price' },
    { name: 'category', type: 'string', title: 'Cuisine Type' },
    { name: 'customizationOptions', type: 'array', of: [{ type: 'string' }], title: 'Custom Options' },
    { name: 'stock', type: 'number', title: 'Stock' }
  ]
};
```

Orders Schema

```
export default {
  name: 'order',
  type: 'document',
  fields: [
    { name: 'customerId', type: 'string', title: 'Customer ID' },
    { name: 'eventType', type: 'string', title: 'Event Type' },
    { name: 'orderItems', type: 'array', of: [{ type: 'object', fields: [
        { name: 'productId', type: 'string', title: 'Product ID' },
        { name: 'quantity', type: 'number', title: 'Quantity' }
    ] }] },
    { name: 'paymentStatus', type: 'string', title: 'Payment Status' }
  ]
};
```

5. System Diagram



6. Technical Roadmap

Milestones and Deliverables:

1. **UI/UX Design:**
   - Build wireframes for key pages (Home, Product Listings, Product Details, Cart, Checkout).
   - Finalize responsive design for mobile and desktop.
2. **Frontend Development:**
   - Implement Next.js for frontend.
   - Integrate Sanity API to fetch and display products dynamically.
3. **Backend Development:**
   - Set up Sanity CMS with schemas for products, orders, and customers.
   - Configure third-party APIs for payment processing and shipment tracking.
4. **API Integration and Testing:**
   - Connect APIs for payments, shipment tracking, and geolocation.
   - Test end-to-end functionality, including order placement and tracking.
5. **Final Deployment and Optimization:**
   - Deploy the platform to a cloud environment.
   - Optimize for performance, security, and scalability.