# ***Marketplace Technical Foundation – Customized & Themed Party Essentials Delivery Q-Commerce***

*This section provides the technical breakdown for creating a seamless platform for delivering customized party essentials quickly and efficiently.*

## ***1. Technical Requirements***

### Frontend Requirements

1. User-Friendly Interface:
    - Easy-to-use navigation to browse, filter, and customize party kits.
    - Dynamic search and sorting options (by theme, price, popularity, eco-friendly options).
2. Responsive Design:
    - Mobile-first design for an excellent experience on all devices.
3. Essential Pages:
    - Home: Highlights trending themes, offers, and featured kits.
    - Product Listings: Browse party essentials by theme (birthdays, holidays, corporate events).
    - Product Details: View item descriptions, customization options, and reviews.
    - Cart: Display selected items with live customization previews.
    - Checkout: Simple, secure, and fast.
    - Order Confirmation: Order tracking, estimated delivery time, and payment status.

### Backend Requirements

1. **Sanity CMS:**
    - Acts as the central backend to manage all products, orders, and user data.
2. **Schemas**:
    - Products Schema:
        - Fields: Name, theme, price, stock, customization options, and images.
    - Orders Schema:
        - Fields: Customer details, ordered items, delivery address, and event type.

  - Customers Schema:
    - Fields: Name, email, contact, and order history.

**Third-Party APIs**

1. Payment Gateway: Stripe or PayPal for secure transactions.
2. Shipment Tracking API: Real-time delivery updates.
3. Geolocation API: Optimize delivery zones and delivery time estimates.

## *2. System Architecture*

*Overview*:
The platform's system architecture ensures a smooth connection between frontend, backend (Sanity CMS), and third-party APIs for seamless operations.

## *3. Key Workflows*

### *1. User Registration:*

- Users register → Data stored in Sanity CMS → Confirmation email sent to the user.

### *2. Product Browsing:*

1. User visits the marketplace →
2. Frontend requests product data from Sanity CMS →
3. Products are dynamically displayed with live customization options.

### *3. Order Placement:*

1. User adds items to the cart and customizes them → Proceeds to checkout.
2. Backend saves order details (customer, products, delivery info) in Sanity CMS.
3. Payment processed via Stripe or PayPal → Confirmation sent to user.
4. Shipment tracking data fetched from third-party API → Displayed to the user in real time.

### *4. Shipment Tracking:*

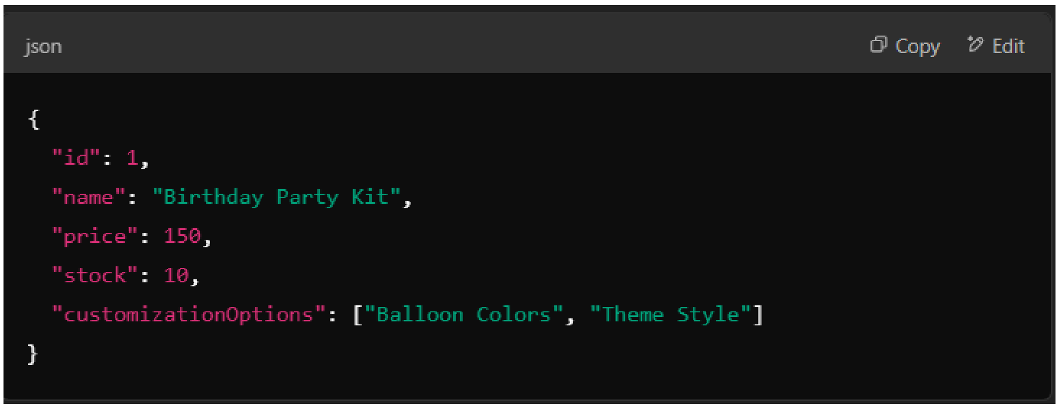
- Users track their order with real-time updates (status and ETA) fetched via API.

## *4. API Requirements*

Endpoints:

▢ /products

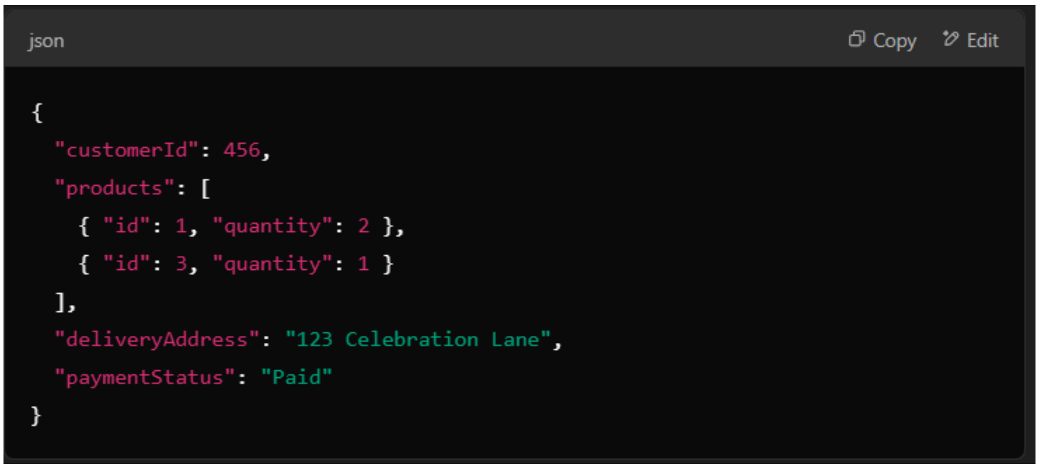
- **Method**: GET
- **Description**: Fetch product listings with customization options.
- **Response**:



```json
{
  "id": 1,
  "name": "Birthday Party Kit",
  "price": 150,
  "stock": 10,
  "customizationOptions": ["Balloon Colors", "Theme Style"]
}
```

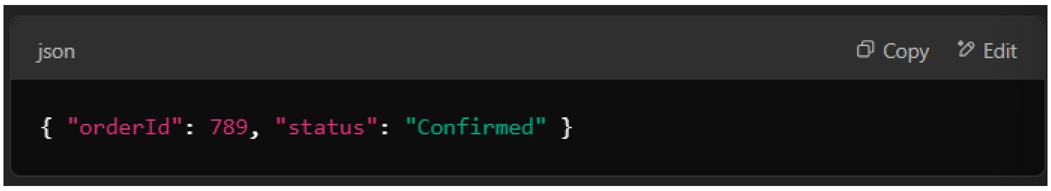
▢ /orders

- **Method**: POST
- **Description**: Save a new order with details of items and customer information.
- **Payload**:



```json
{
  "customerId": 456,
  "products": [
    { "id": 1, "quantity": 2 },
    { "id": 3, "quantity": 1 }
  ],
  "deliveryAddress": "123 Celebration Lane",
  "paymentStatus": "Paid"
}
```

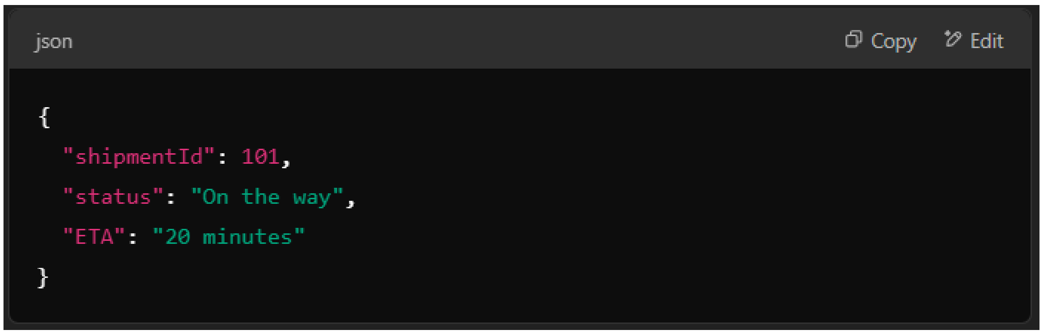
- **Response**:



```json
{ "orderId": 789, "status": "Confirmed" }
```

▢ /shipment

- **Method**: GET
- **Description**: Fetch real-time delivery status.
- **Response**:



```json
{
  "shipmentId": 101,
  "status": "On the way",
  "ETA": "20 minutes"
}
```

## *5. Sanity Schema Examples*

### Products Schema:

- Fields: Name, price, theme, stock, customization options, images.

### Orders Schema:

- Fields: Customer ID, items ordered, delivery address, payment status.

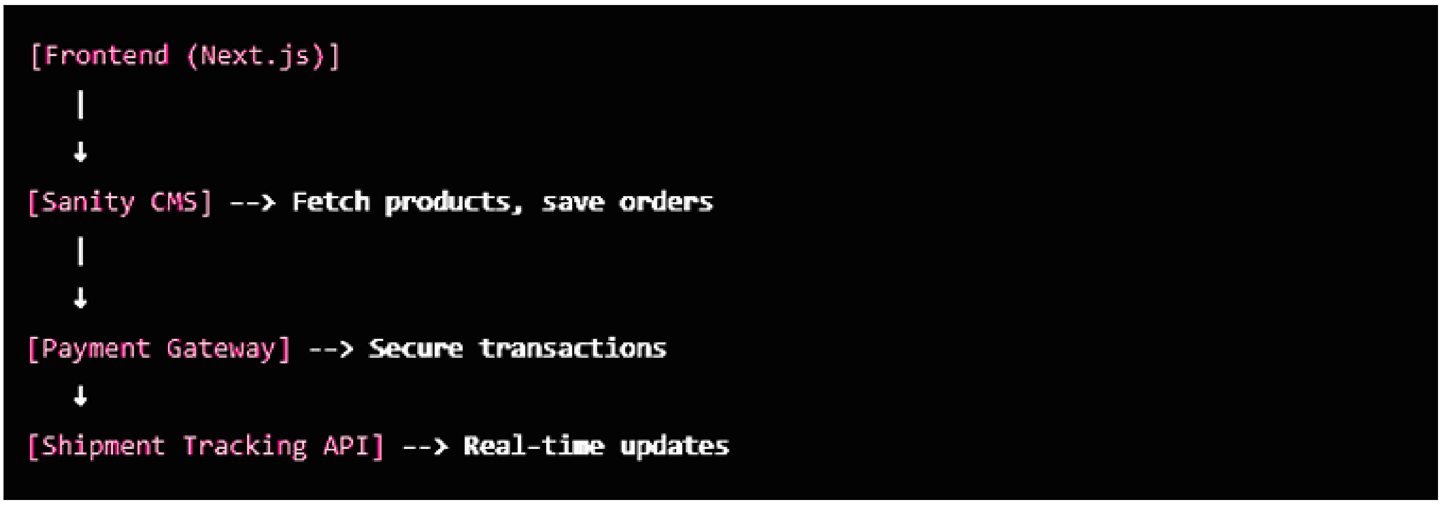
### Customers Schema:

- Fields: Name, email, contact info, and order history.

## *6. System Diagram*

**Components**:

1. Frontend: React.js or Next.js for building a dynamic and responsive user interface.
2. Backend: Node.js server integrated with Sanity CMS for API and database management.
3. Third-Party APIs: Stripe (payments), Mapbox (geolocation), and a shipment tracking service.



```
[Frontend (Next.js)]
   |
   ↓
[Sanity CMS] --> Fetch products, save orders
   |
   ↓
[Payment Gateway] --> Secure transactions
   ↓
[Shipment Tracking API] --> Real-time updates
```

## *7. Technical Roadmap*

**Milestones and Deliverables:**

1. UI/UX Design:
   - Design wireframes for all key pages.
   - Ensure a consistent and mobile-friendly interface.
2. Frontend Development:
   - Develop pages (Home, Listings, Details, Cart, Checkout) using React or Next.js.
   - Integrate with Sanity CMS to fetch and display data dynamically.
3. Backend Development:
   - Set up Sanity CMS schemas for managing product, order, and user data.
   - Create APIs for product browsing, order management, and tracking.
4. API Integration and Testing:
   - Connect Stripe for payment processing.
   - Integrate shipment tracking and geolocation APIs.
   - Test workflows from user registration to order delivery.
5. Deployment and Optimization:
   - Deploy the platform using AWS or Google Cloud.
   - Optimize for high performance, security, and scalability.

This approach mirrors the provided structure while tailoring it to your Customized & Themed Party Essentials Delivery idea, ensuring a robust technical foundation!