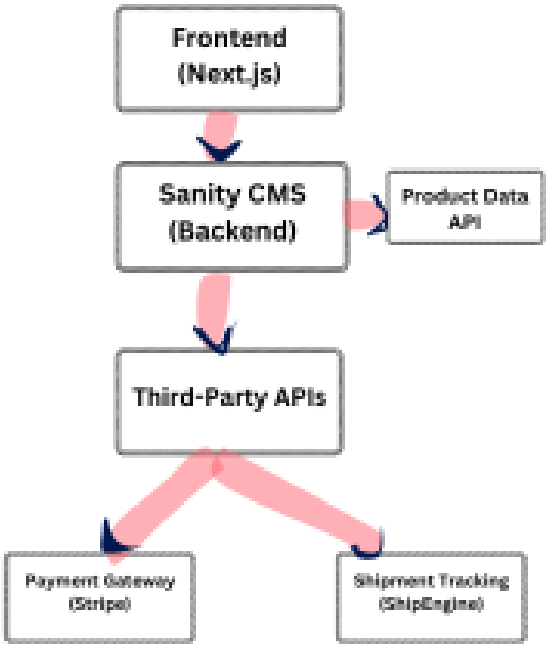
Marketplace Technical Foundation - Fashion E-Commerce Platform

1. System Architecture Overview

System Architecture Diagram



Component Roles

- **Frontend (Next.js):** The user interface where customers browse products, place orders, and make payments.
- **Sanity CMS:** Manages product details, customer information, and order records.
- **Product Data API:** Fetches product details from Sanity CMS and displays them on the frontend.
- **Third-Party API:** Handles external services like shipment tracking and payment processing.
- **Payment Gateway:** Securely processes payments made by users.

2. Core Processes

User Sign-Up

1. A new user registers on the platform.
2. Their profile is stored in the CMS.
3. A verification message is sent to the user.

Exploring Products

1. A user views available items on the interface.
2. The interface retrieves product data from the **Data Retrieval API**.
3. Items are shown dynamically on the site.

Placing an Order

1. A user selects items and proceeds to checkout.
2. The purchase details are sent to the CMS via the **Order API**.
3. The transaction is recorded in the CMS.

Tracking Delivery

1. The interface fetches live updates from the **Delivery Tracking API**.
2. The delivery status is shown to the user.

Completing Payment

1. The user enters payment information (e.g., card number, amount).
2. The interface sends this data to the **Transaction Processor** for handling.
3. Once the payment is successful, a confirmation is sent to the user and logged in the CMS.

2. API Specifications

Endpoint Name	Method	Description	Example Request/Response
/products	GET	Fetch all available products.	Request: GET /products Response: [{ "id": 1, "name": "Nike Air Max", ... }]
/orders	GET	Create a new order.	Request: POST /orders Response: { "orderId": 123, "status": "Confirmed" }
/shipment	GET	Track the status of an order.	Request: GET /shipment?orderId=123 Response: { "status": "In Transit", ... }
/product-details	GET	Fetch detailed information about a product.	Request: GET /product-details?id=1 Response: { "id": 1, "name": "Nike Air Max", ... }
/payments	POST	Process payment for an order.	Request: POST /payments Response: { "paymentId": 789, "status": "Paid" }

4. CMS Schema Examples

Product Schema

```
export default {
```

```
  name: 'item',
  type: 'document',
  fields: [
    { name: 'name', type: 'string', title: 'Item Name' },
    { name: 'brand', type: 'reference', to: [{ type: 'brand' }], title: 'Brand' },
    { name: 'price', type: 'number', title: 'Price' },
    { name: 'quantity', type: 'number', title: 'Quantity' },
    { name: 'details', type: 'text', title: 'Details' },
    { name: 'image', type: 'image', title: 'Item Image' }
  ]
};
```

User Schema

```
export default {
  name: 'user',
  type: 'document',
  fields: [
    { name: 'name', type: 'string', title: 'Full Name' },
    { name: 'email', type: 'string', title: 'Email' },
    { name: 'phone', type: 'string', title: 'Phone Number' },
    { name: 'address', type: 'string', title: 'Address' }
  ]
};
```

Transaction Schema

```
export default {
  name: 'transaction',
  type: 'document',
  fields: [
    { name: 'item', type: 'reference', to: [{ type: 'item' }], title: 'Item' },
    { name: 'user', type: 'reference', to: [{ type: 'user' }], title: 'User' },
    { name: 'quantity', type: 'number', title: 'Quantity' },
    { name: 'totalAmount', type: 'number', title: 'Total Amount' },
    { name: 'status', type: 'string', title: 'Transaction Status' }
  ]
};
```

5. Development Timeline

Phase 1: Interface Development (Weeks 1-2)

- Build the user interface using Next.js.
- Design and implement CMS schemas.

Phase 2: Backend Integration (Weeks 3-4)

- Integrate external services for payment handling and delivery tracking.
- Develop and test API endpoints.

Phase 3: Testing and Launch (Weeks 5-6)

- Conduct comprehensive testing.
- Launch the platform and gather user feedback.