Hackathon Day 3

Integrating APIs & Migrating Data for the Clothing Platform

API Integration

To ensure smooth operations, various APIs will be incorporated to connect the frontend, backend, and external services. Below is an outline of their functionalities:

1.User Authentication API:

- Enables secure user registration, login, and account management.
- Uses JWT for secure and efficient session handling.

2. Sanity CMS API:

- Fetches dynamic content like product details and categories for the frontend.
- Supports backend operations to manage products and content updates.

3. Cart and Order Management API:

- Facilitates cart operations, such as adding/removing products and calculating totals.
- Sends order details, including user and payment information, to the backend for processing.

4. Payment Gateway API (e.g., Stripe):

- Processes payments securely during the checkout process.
- Confirms payments and updates order statuses upon completion.

5. Shipment Tracking API:

- Provides real-time updates on order shipping status.
- Integrates with delivery services for accurate tracking and delivery management.

Data Migration

To build a robust backend, existing data will be transferred to Sanity CMS in a structured and systematic way.

1. Preparation:

- Format product, user, and order data into CSV/JSON files compatible with Sanity CMS.
- Define schemas in Sanity for key entities, including:
 - **Products:** Attributes such as name, price, and stock.

- Users: Information like profiles and addresses.
- Orders: Details on items, costs, and status.

2. Migration Steps:

- Use tools like Sanity CLI or custom scripts to upload prepared data.
- Validate imported data to ensure accuracy and consistency.

3. Testing:

- Retrieve migrated data via the Sanity API to verify proper functionality.
- Test relationships, such as product-category links and user-order associations.

4. Ongoing Updates:

 Set up webhooks to sync new entries for products, users, or orders with the database.

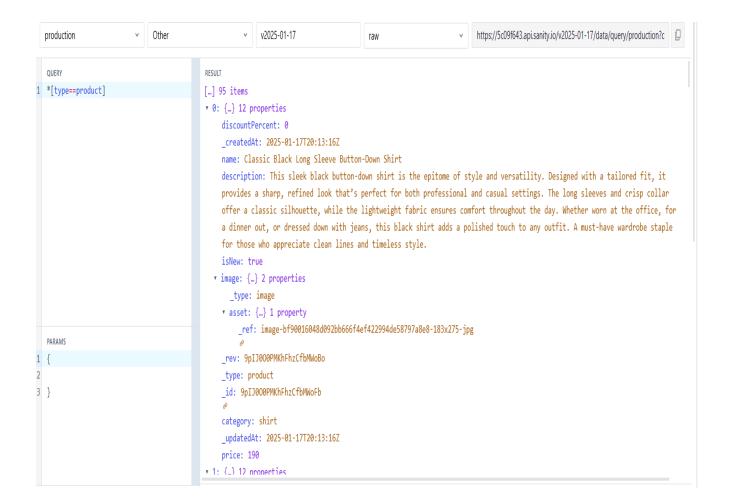
By combining efficient API integration and data migration techniques, the platform will deliver a seamless, feature-rich experience for users.

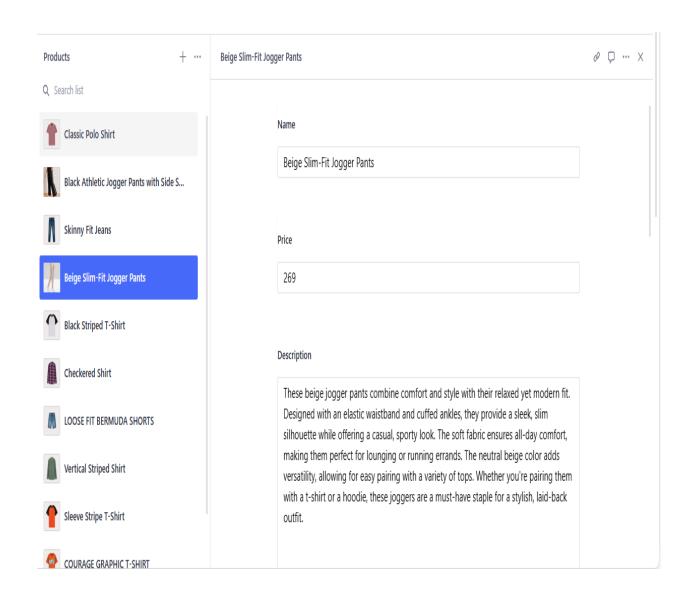
```
TS products.ts U X
     import { defineType } from "sanity"
 2
     export default defineType({
 3
 4
         name: 'products',
         title: 'Products',
 5
         type: 'document',
 6
 7
         fields: [
 8
             {
             name: 'name',
 9
             title: 'Name',
10
             type: 'string',
11
12
             },
13
             {
             name: 'price',
14
             title: 'Price',
15
             type: 'number',
16
17
             },
             {
18
             name: 'description',
19
20
             title: 'Description',
21
             type: 'text',
22
             },
23
             name: 'image',
24
25
             title: 'Image',
             type: 'image',
26
27
             },
28
29
                 name: "category",
30
                  title: "Category",
                  type: 'string',
31
                  options:{
32
```

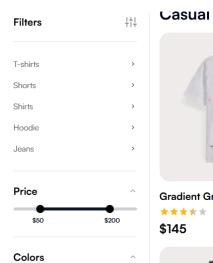
```
  importData.js U 

  ✓

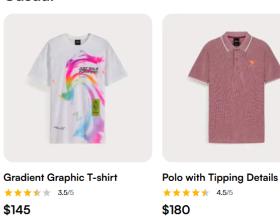
 1
      import { createClient } from '@sanity/client';
 2
     const client = createClient({
projectId: '5c09f643',
 3
 4
 5
        dataset: 'production',
 6
        useCdn: true,
        apiVersion: '2025-01-13',
 7
 8
        token:
        skawVDVATysRQWdyfD5rH5GaZ61qrxWg3ZQ1BX6XrbZ41E4dvLMfytjWYyDAda5DzNeKchqfrt5PKmhCdi7bTZxGooGaMQ9PrfIIeqAoNphGwwu
        qVQqc9WXJTUI8uvYsJcy1Qn5RA0GWlirHG3SVj0PRVI6dGOlho9l1S',
 9
10
      async function uploadImageToSanity(imageUrl) {
11
12
          console.log(`Uploading image: ${imageUrl}`);
13
14
15
          const response = await fetch(imageUrl);
16
17
          if (!response.ok) {
18
          throw new Error(`Failed to fetch image: ${imageUrl}`);
19
20
21
          const buffer = await response.arrayBuffer();
          const bufferImage = Buffer.from(buffer);
22
23
24
          const asset = await client.assets.upload('image', bufferImage, {
          filename: imageUrl.split('/').pop(),
25
26
          });
27
28
          console.log(`Image uploaded successfully: ${asset._id}`);
29
          return asset._id;
30
        } catch (error) {
          console error('Failed to unload image:' imagellal error):
21
```







Siza







Showing I-IO of IOO Products Sort by: Most Popular >

★★★★★ 5.0/5

\$105 \$150 -30%





