**Day 3 API INTERATION AND DATA MIGRATION TO E-Commerce**

**Name: Narjis Fatima.**

Create my own API and migrated data to Sanity.

**Introduction**

This report details the process of integrating APIs, adjusting schemas, and migrating data to the Sanity CMS for the Fashion and Other Accessories website Marketplace. The primary objective was to facilitate seamless API interactions, incorporate external product data, and ensure that the CMS is fully populated and operational.

**API Integration Process**

1. **API Integration**

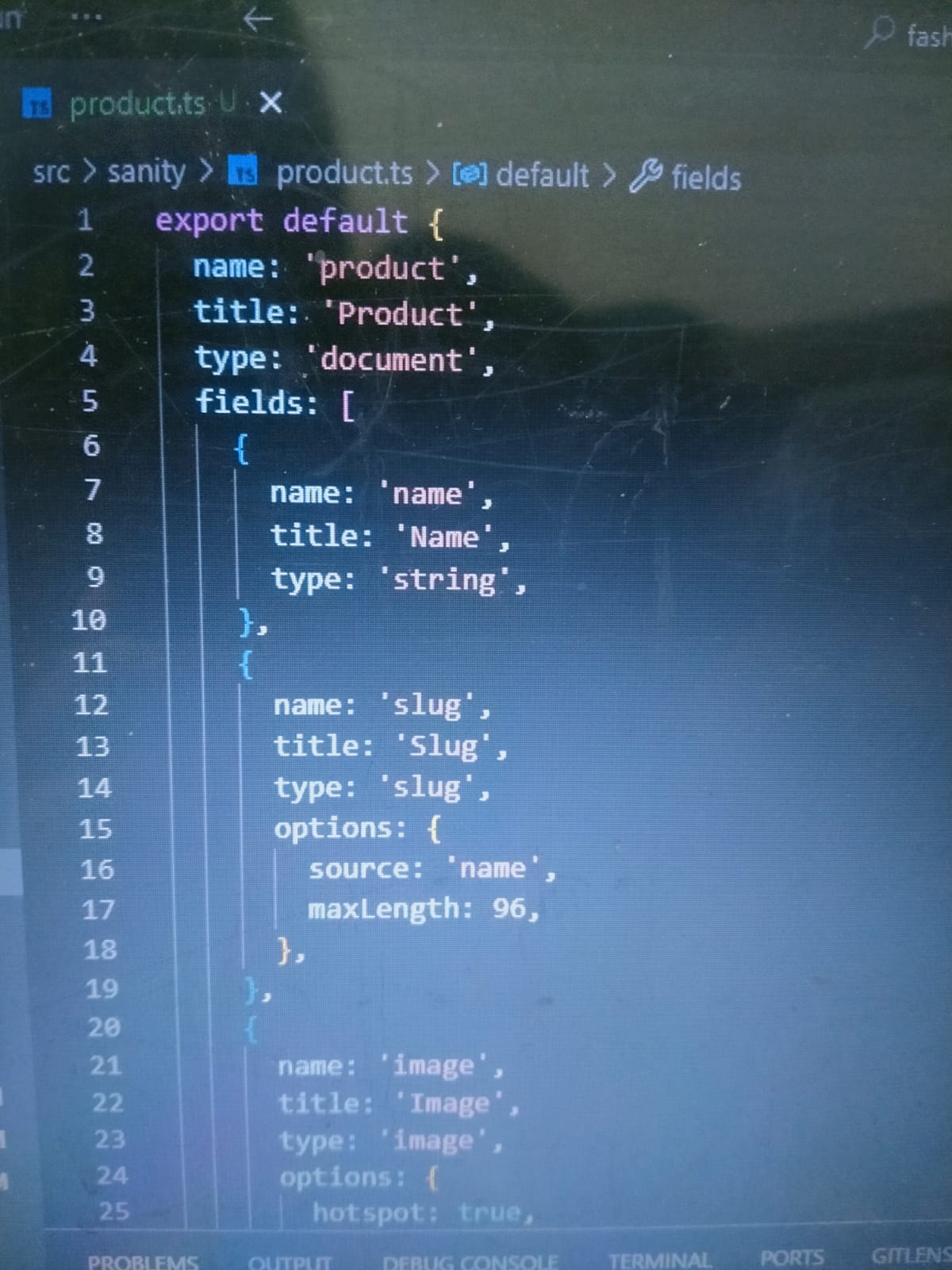
The API integration process involved retrieving product data from an external source and populating it within the Sanity CMS. The following steps were undertaken:

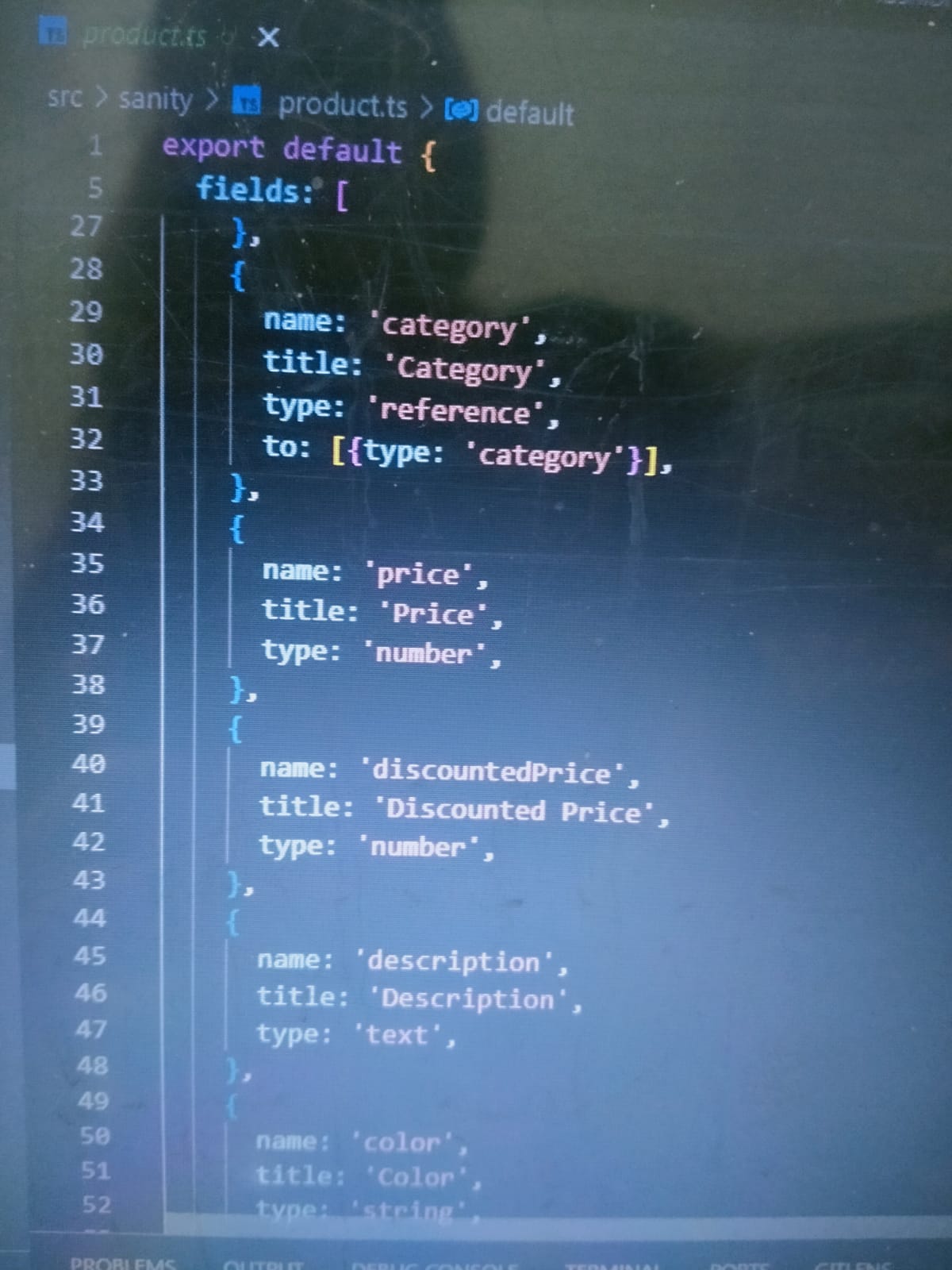
1. **Identifying the API Endpoints:**
   * We identified the external API that supplies product data, including endpoints for product details, images, and categories.
2. **Creating API Fetch Functions:**
   * We developed a fetch function using either the native **fetch()** method or an alternative library like Axios to retrieve product data from the external API.

**3. Integrating API Calls in the Project:**

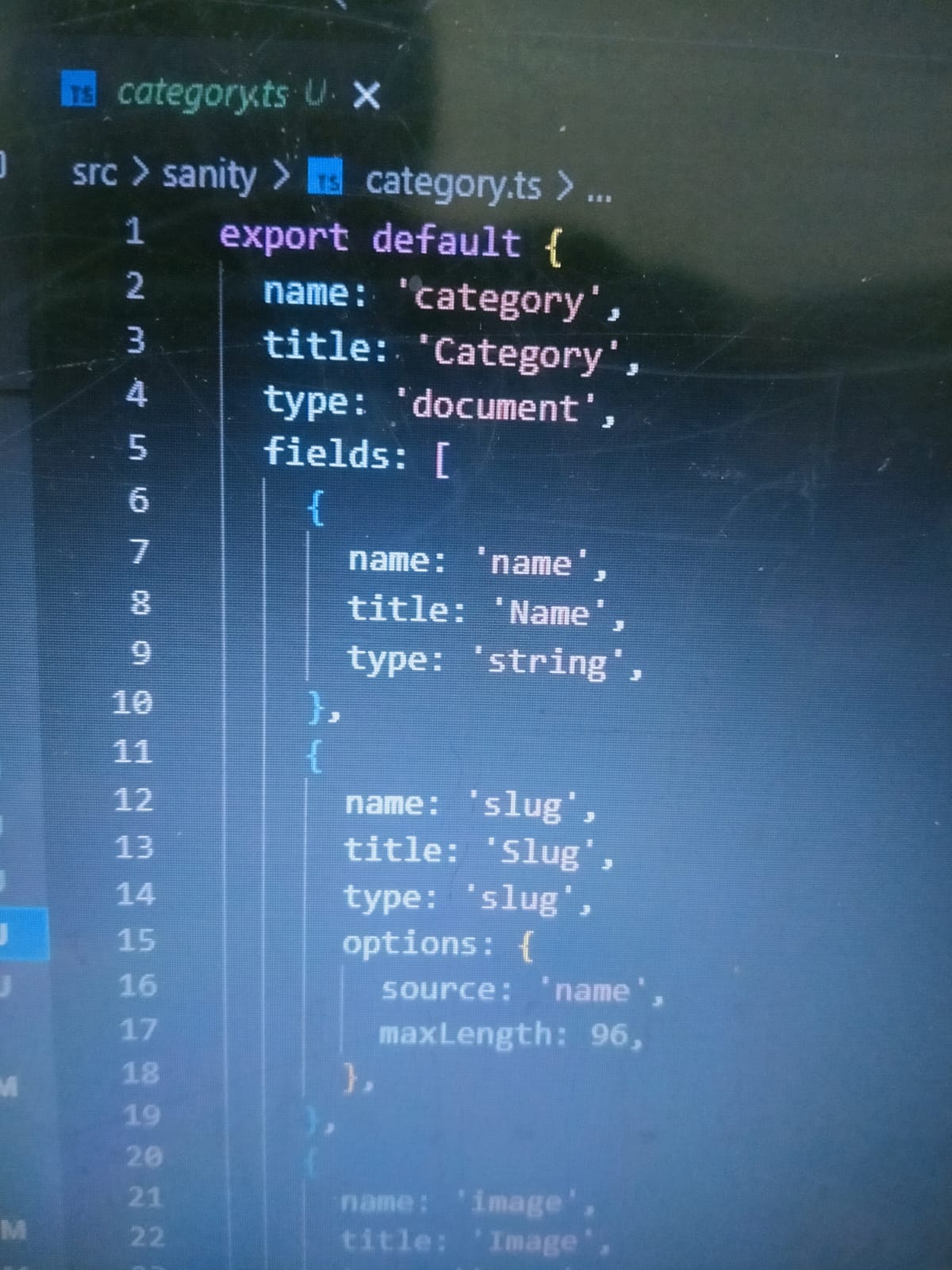
* + **API Calls Implementation:**
  + API calls were executed during the data fetching process within the Next.js framework. The API was seamlessly integrated with the frontend and utilized in various pages, including the Product Detail page.
  + **Frontend Display of API Data:**
  + Once the data was successfully fetched, it was rendered on the frontend. For instance, on the Product Detail page, we showcased essential product information such as the title, price, description, and inventory status, all retrieved from the API.
  + **Handling Data:**
  + Upon successful API calls, the product data was parsed and displayed on the frontend. The product details were dynamically rendered on the Product Detail page, ensuring a smooth user experience.
  + **Error Handling:**
  + A robust fallback mechanism was implemented to manage any errors that might occur during the API fetch process, such as scenarios where a product could not be found.

1. **Adjustments Made to Schemas:**
   1. **2. Schema Adjustments:** To effectively integrate and store data, several modifications were made to the Sanity CMS schemas:
   2. **Product Schema:**
   3. New fields were incorporated into the product schema to accommodate API data, including attributes like **priceWithDiscount**, **tags**, and **rating**.



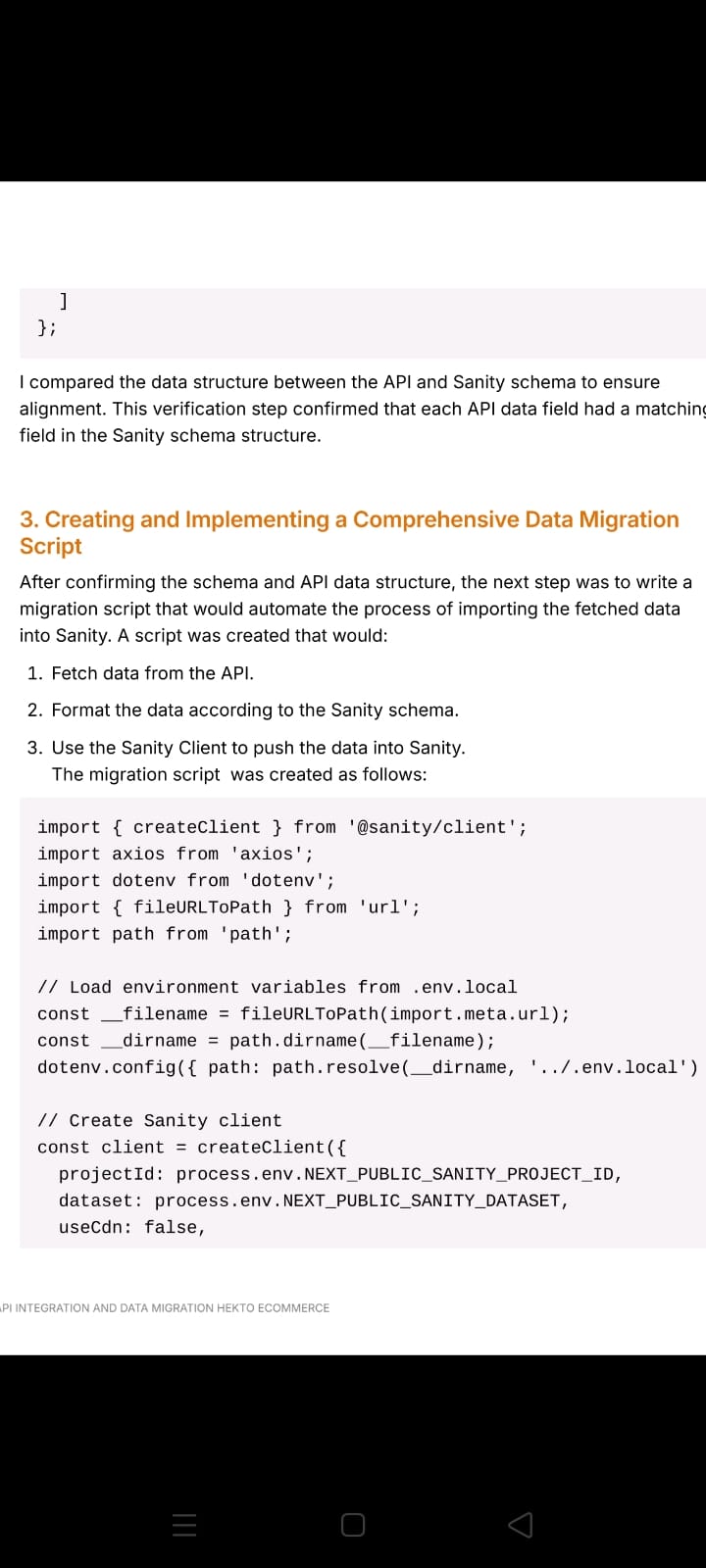


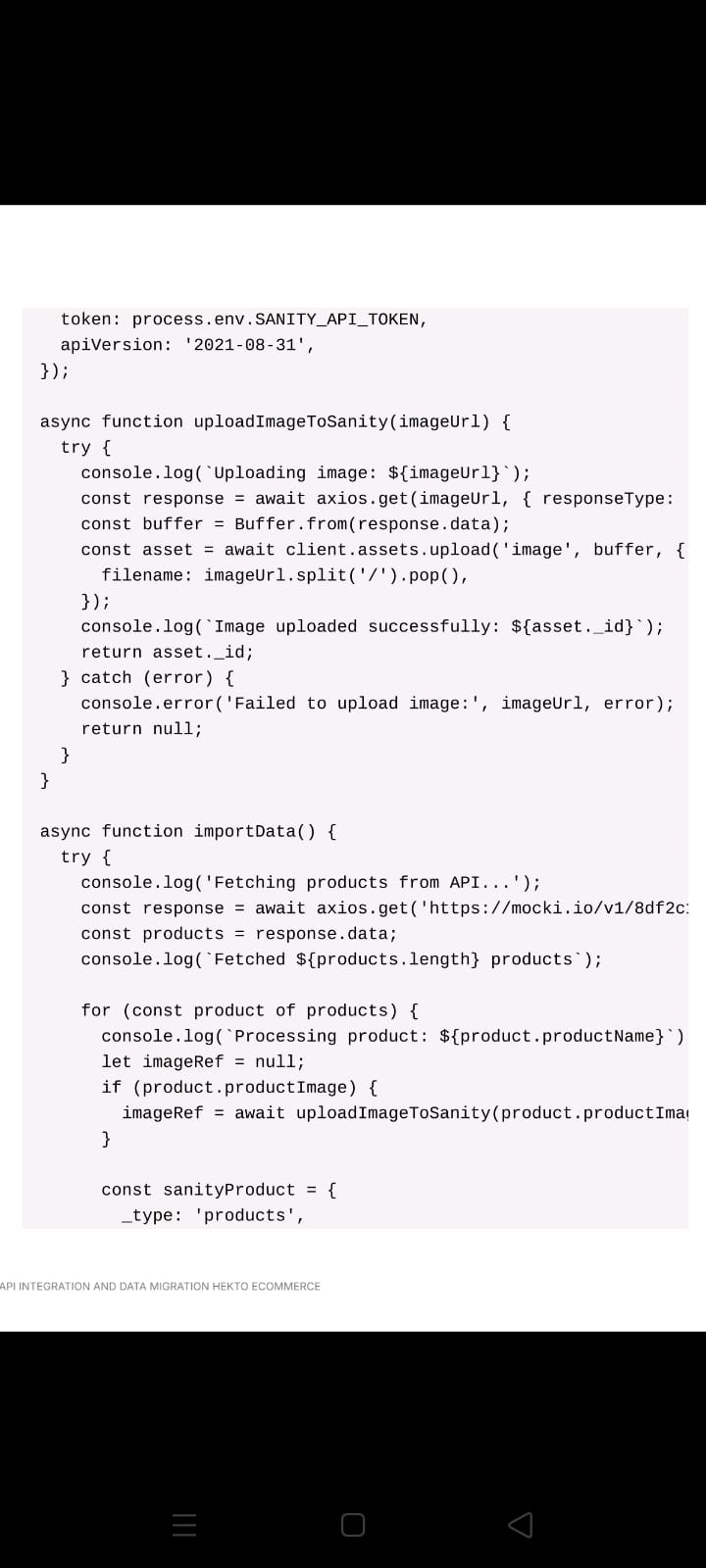
* 1. **Category Schema:**
  2. A reference to categories was added within the product schema to link each product to its respective category, enhancing organization and navigation.



* 1. **Image Handling:**
  2. The product images retrieved via the API were connected to Sanity’s image asset reference field, ensuring proper management and display of image data.

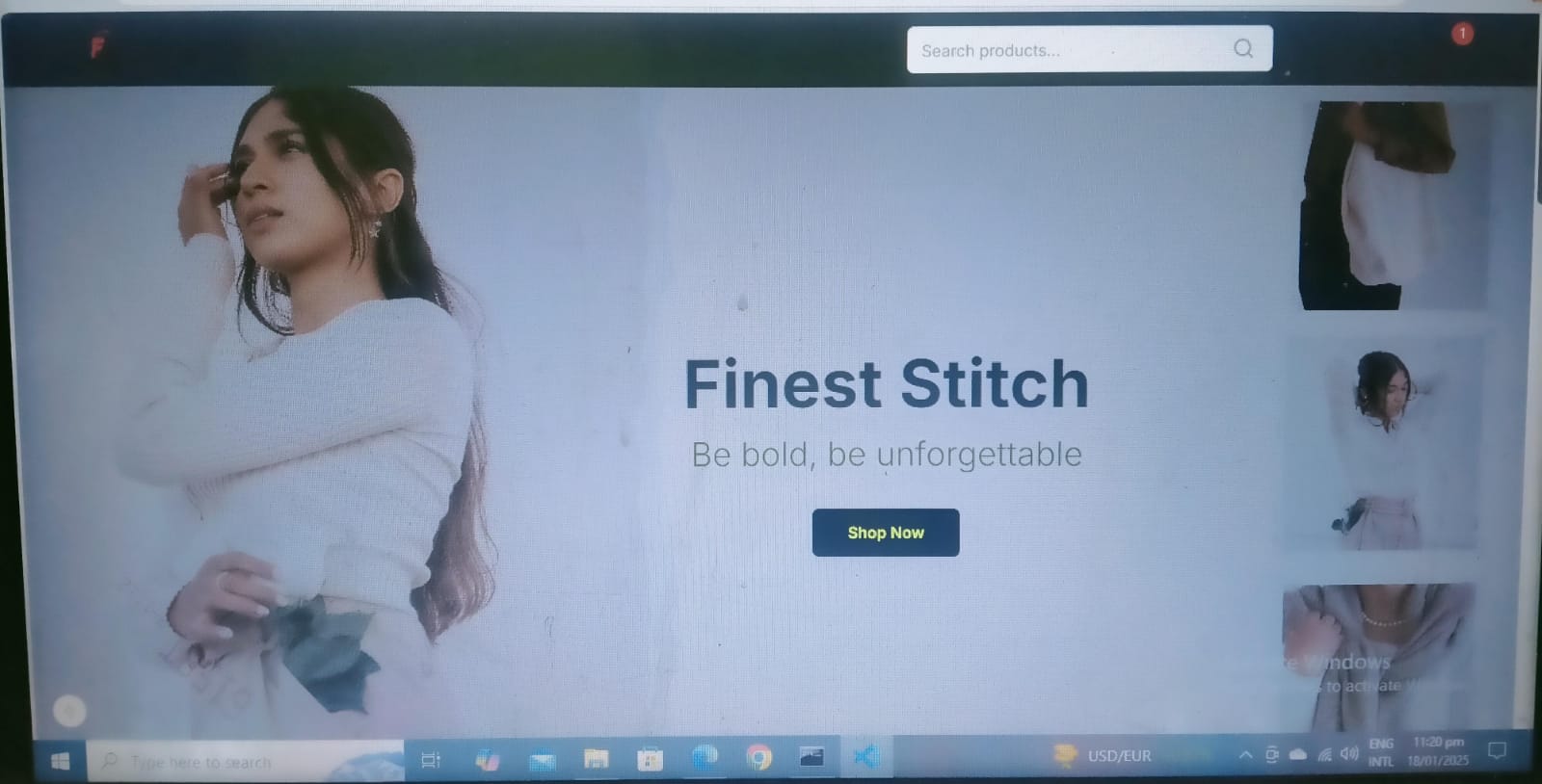
1. **Migration Steps and Tools Used:**
   1. **3. Data Migration:** The migration of product data from the external API to the Sanity CMS was executed through the following steps:
   2. **Preparing the Migration Script:**
   3. A Node.js script was developed to automate the migration process. This script utilized Sanity’s client to create documents in the CMS using data sourced from the external API.
   4. **Script to Migrate Product Data:**
   5. The script iterated through the product data and pushed it to Sanity’s CMS, ensuring that all relevant information was accurately transferred.

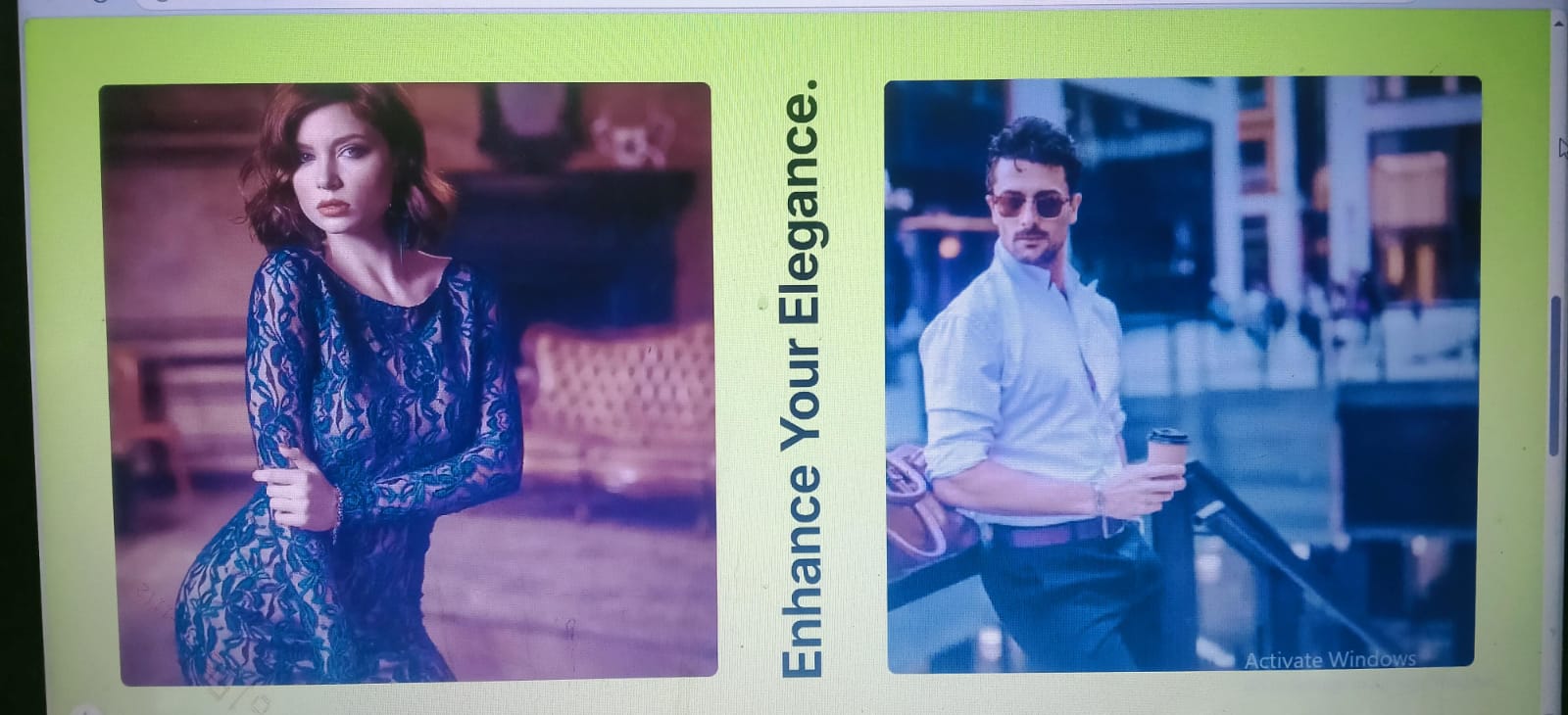






1. **Sanity Studio:**
   1. **Verification of Data Migration:**
   2. Sanity Studio was employed to confirm that the data was correctly migrated and populated. We reviewed the product documents and their respective fields to ensure that all pertinent information was accurately stored.
   3. **Verification Process:**
   4. After the migration, we verified the data within the Sanity Studio interface, ensuring that all product information, including pricing, descriptions, and categories, was correctly integrated.
2. **Screenshots:**
   1. **API Calls:**
   2. Screenshots showcasing successful API calls made to fetch product data.
   3. **Frontend Display:**
   4. A screenshot illustrating the data displayed on the frontend, such as the product details page featuring information sourced from the API.





* 1. **Populated Sanity CMS Fields:**
  2. A screenshot of Sanity Studio displaying the product document with populated fields, including price, description, and tags.
  3. **Conclusion:** The API integration process for the clothing and accessories website was successfully executed, with product data being fetched, stored, and displayed on the frontend. Essential adjustments were made to the Sanity CMS schema to accommodate the new product information. The data migration was automated through a custom script, ensuring a seamless transition from the external API to the CMS.