# Day 3 Hackathon Documentation:

# API Integration with Sanity.io CMS

**Objective:**

Integrate a mock API with Sanity.io CMS to enhance content management functionality. This integration will allow real-time data fetching, updates, and management of content stored in Sanity CMS using a fake API as a source.

**Key Concepts:**

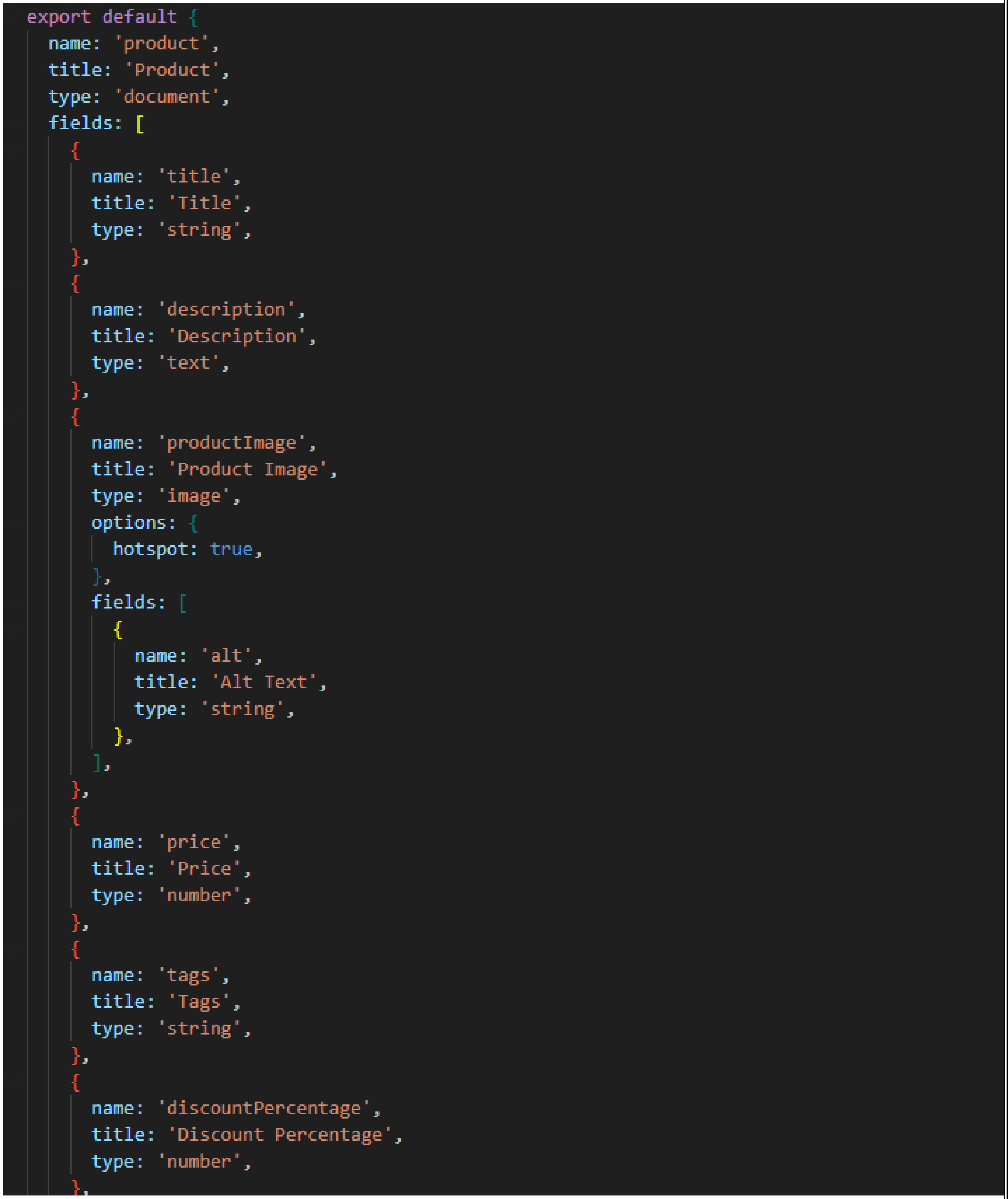
1. **Sanity.io CMS**: A headless CMS (Content Management System) that provides structured content, offering flexibility for managing and distributing content across platforms.
2. **API Integration**: In this case, connecting an external API (fake API) with Sanity to fetch and push data.
3. **Mock api**: A simulated API that provides fake data to mimic real-world APIs. It will be used to simulate the process of API integration with Sanity.io CMS.

**Integration Overview:**

The goal is to pull data from the mock api and automatically update or create content in Sanity.io CMS. This integration will be done using a server-side approach (Node.js or Next.js API routes).

**Steps for API Integration:**

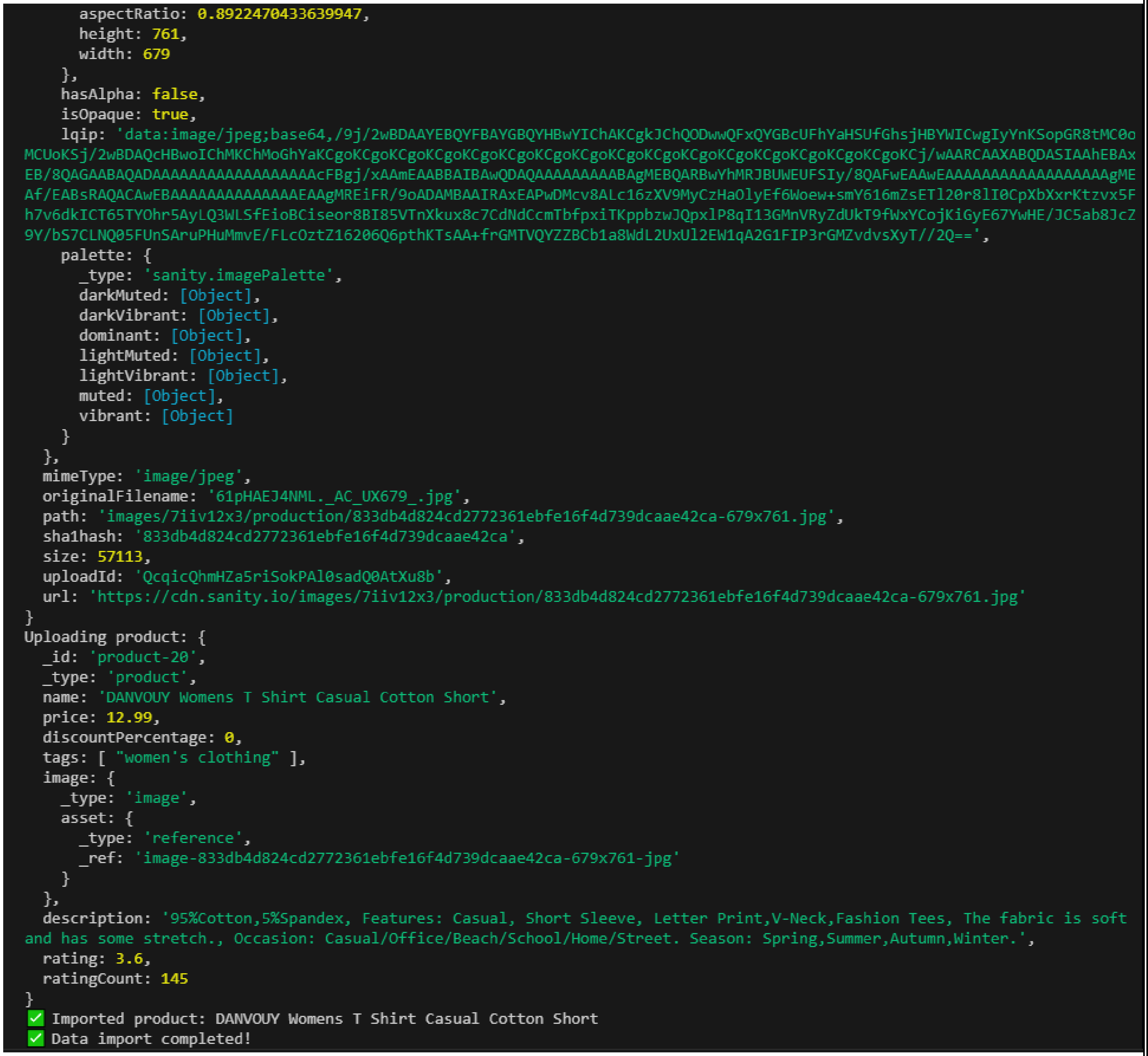
1. **Setup Sanity.io CMS**:
   - If not already done, create a Sanity.io project and a dataset where the content will be stored.
   - Define the schema in Sanity Studio to structure the content you want to receive from the API.



```
export default {
  name: 'product',
  title: 'Product',
  type: 'document',
  fields: [
    {
      name: 'title',
      title: 'Title',
      type: 'string',
    },
    {
      name: 'description',
      title: 'Description',
      type: 'text',
    },
    {
      name: 'productImage',
      title: 'Product Image',
      type: 'image',
      options: {
        hotspot: true,
      },
      fields: [
        {
          name: 'alt',
          title: 'Alt Text',
          type: 'string',
        },
      ],
    },
    {
      name: 'price',
      title: 'Price',
      type: 'number',
    },
    {
      name: 'tags',
      title: 'Tags',
      type: 'string',
    },
    {
      name: 'discountPercentage',
      title: 'Discount Percentage',
      type: 'number',
    },
```

2. **Fetch Data from mock.api**:

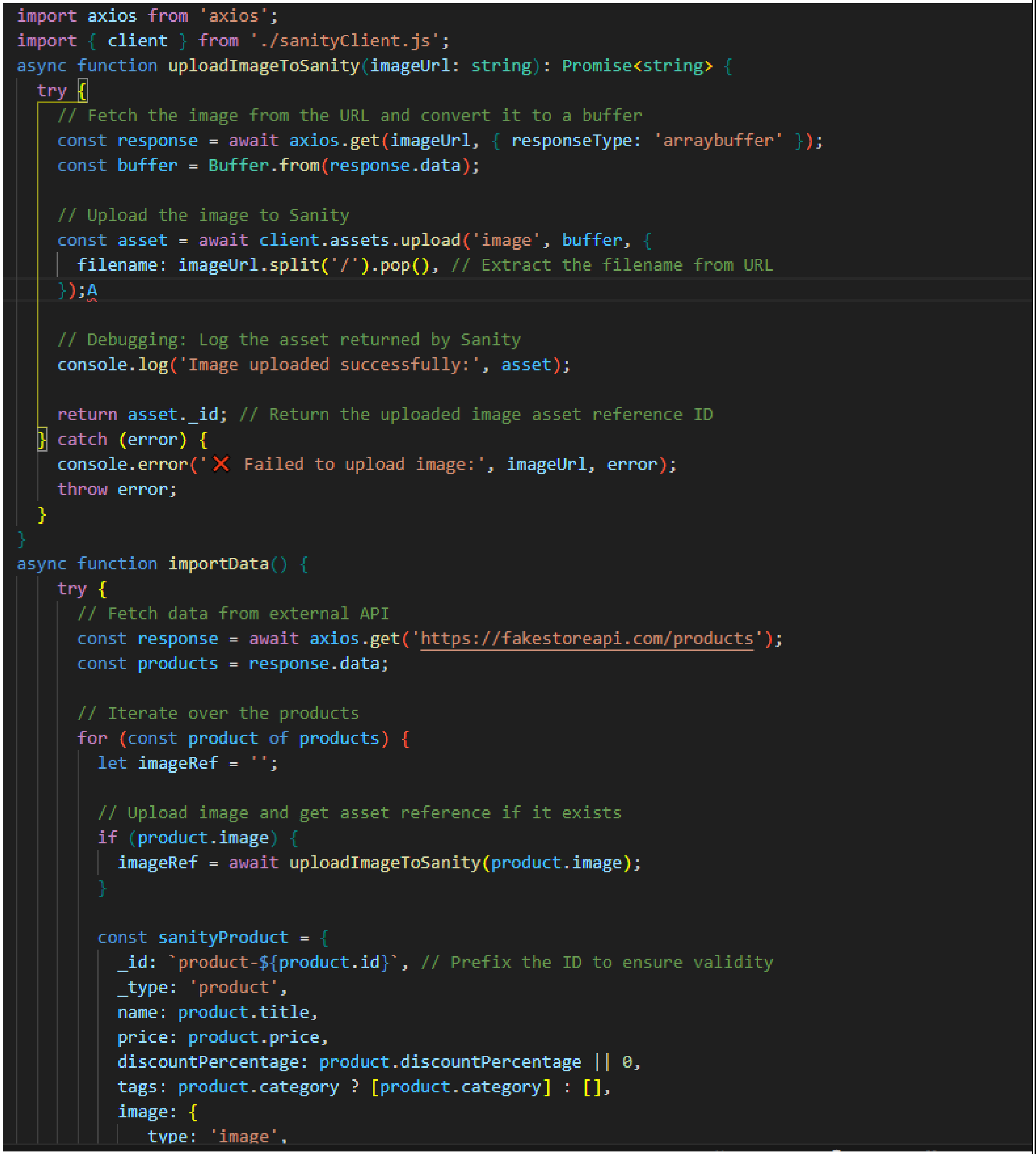
   - Use Node.js or Next.js API routes to make requests to the mock.api endpoint.
   - Parse the response to extract the required content.



```
      aspectRatio: 0.8922470433639947,
      height: 761,
      width: 679
    },
    hasAlpha: false,
    isOpaque: true,
    lqip: 'data:image/jpeg;base64,/9j/2wBDAAYEBQYFBAYGBQYHBwYIChAKCgkJChQODwwQFxQYGBcUFhYaHSUfGhsjHBYWICwgIyYnKSopGR8tMC0o
MCUoKSj/2wBDAQcHBwoIChMKChMoGhYaKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCj/wAARCAAXABQDASIAAhEBAx
EB/8QAGAABAQADAAAAAAAAAAAAAAAAAAAcFBgj/xAAmEAABBAIBAwQDAQAAAAAAAAABAgMEBQARBwYhMRJBUWEUFSIy/8QAFwEAAwEAAAAAAAAAAAAAAAAAAgME
Af/EABsRAQACAwEBAAAAAAAAAAAAAAAEAAgMREiFR/9oADAMBAAIRAxEAPwDMcv8ALc16zXV9MyCzHaOlyEf6Woew+smY616mZsETl20r8lI0CpXbXxrKtzvx5F
h7v6dkICT65TYOhr5AyLQ3WLSfEioBCiseor8BI85VTnXkux8c7CdNdCcmTbfpxiTKppbzwJQpxlP8qI13GMnVRyZdUkT9fWxYCojKiGyE67YwHE/JC5ab8JcZ
9Y/bS7CLNQ05FUnSAruPHuMmvE/FLcOztZ16206Q6pthKTsAA+frGMTVQYZZBCb1a8WdL2UxUl2EW1qA2G1FIP3rGMZvdvsXyT//2Q==',
    palette: {
      _type: 'sanity.imagePalette',
      darkMuted: [Object],
      darkVibrant: [Object],
      dominant: [Object],
      lightMuted: [Object],
      lightVibrant: [Object],
      muted: [Object],
      vibrant: [Object]
    }
  },
  mimeType: 'image/jpeg',
  originalFilename: '61pHAEJ4NML._AC_UX679_.jpg',
  path: 'images/7iiv12x3/production/833db4d824cd2772361ebfe16f4d739dcaae42ca-679x761.jpg',
  sha1hash: '833db4d824cd2772361ebfe16f4d739dcaae42ca',
  size: 57113,
  uploadId: 'QcqicQhmHZa5riSokPAl0sadQ0AtXu8b',
  url: 'https://cdn.sanity.io/images/7iiv12x3/production/833db4d824cd2772361ebfe16f4d739dcaae42ca-679x761.jpg'
}
Uploading product: {
  _id: 'product-20',
  _type: 'product',
  name: 'DANVOUY Womens T Shirt Casual Cotton Short',
  price: 12.99,
  discountPercentage: 0,
  tags: [ "women's clothing" ],
  image: {
    _type: 'image',
    asset: {
      _type: 'reference',
      _ref: 'image-833db4d824cd2772361ebfe16f4d739dcaae42ca-679x761-jpg'
    }
  },
  description: '95%Cotton,5%Spandex, Features: Casual, Short Sleeve, Letter Print,V-Neck,Fashion Tees, The fabric is soft
and has some stretch., Occasion: Casual/Office/Beach/School/Home/Street. Season: Spring,Summer,Autumn,Winter.',
  rating: 3.6,
  ratingCount: 145
}
✅ Imported product: DANVOUY Womens T Shirt Casual Cotton Short
✅ Data import completed!
```

3. **Integrate with Sanity.io API**:

   - Use the Sanity client to push the formatted data into the CMS.
   - If new data needs to be created, use the create method. If updating existing content, use the patch method.



```
import axios from 'axios';
import { client } from './sanityClient.js';
async function uploadImageToSanity(imageUrl: string): Promise<string> {
  try {
    // Fetch the image from the URL and convert it to a buffer
    const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
    const buffer = Buffer.from(response.data);

    // Upload the image to Sanity
    const asset = await client.assets.upload('image', buffer, {
      filename: imageUrl.split('/').pop(), // Extract the filename from URL
    });A

    // Debugging: Log the asset returned by Sanity
    console.log('Image uploaded successfully:', asset);

    return asset._id; // Return the uploaded image asset reference ID
  } catch (error) {
    console.error('❌ Failed to upload image:', imageUrl, error);
    throw error;
  }
}
async function importData() {
    try {
      // Fetch data from external API
      const response = await axios.get('https://fakestoreapi.com/products');
      const products = response.data;

      // Iterate over the products
      for (const product of products) {
        let imageRef = '';

        // Upload image and get asset reference if it exists
        if (product.image) {
          imageRef = await uploadImageToSanity(product.image);
        }

        const sanityProduct = {
          _id: `product-${product.id}`, // Prefix the ID to ensure validity
          _type: 'product',
          name: product.title,
          price: product.price,
          discountPercentage: product.discountPercentage || 0,
          tags: product.category ? [product.category] : [],
          image: {
             type: 'image',
```

4. **Testing**:
    - Run the integration and verify the content is being added to Sanity.
    - Ensure the data displayed in Sanity Studio matches the structure and values from the mock API.

**Example Workflow:**

1. The fake API (mock.api) provides product data (e.g., name, price, description).
2. Fetch data from mock.api using a server-side script.
3. Format the data to match the Sanity schema for products.
4. Insert the formatted data into Sanity.io CMS using the Sanity client.
5. Validate the content in Sanity Studio.

**Final Deliverables:**

1. **API Integration Code**: A Node.js or Next.js script that fetches data from the mock API and pushes it into Sanity.io CMS.
2. **Sanity Schema**: A custom schema to represent the data structure in Sanity.
3. **Testing & Validation**: Evidence of successful data insertion and validation within Sanity Studio.

**Challenges Faced:**

- Handling API rate limits from the fake API.
- Ensuring the data format from the mock API matches the expected structure in Sanity.
- Ensuring data integrity during the integration process.

**Conclusion:**

This integration will streamline content management by connecting Sanity.io CMS with a mock API. It allows automated content updates without the need for manual data entry, demonstrating a core principle of headless CMS use—flexibility and automation.

**Self-Validation Checklist** ✅

**✓ API Understanding**
**✓ Schema Validation**
**✓ Data Migration**
**✓ API Integration in Next.js**
**✓ Submission Preparation**