Day 3 - API Integration Report - Furniro

Introduction:

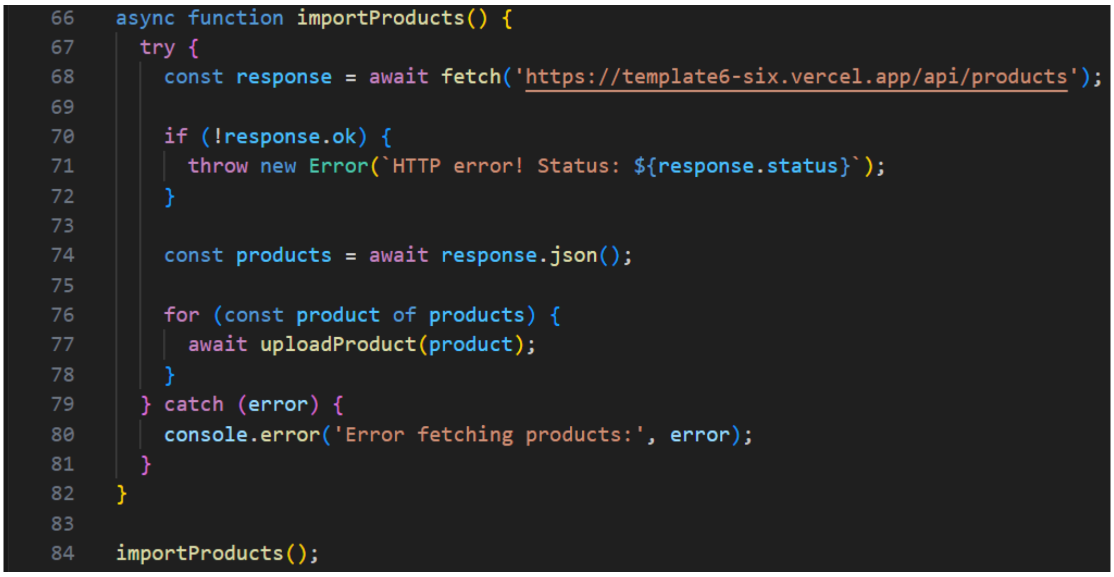
The purpose of this report is to enhance our skills in API integration and data migration by working on a functional marketplace backend. This exercise involved populating Sanity CMS with data from a provided API and integrating it into a Next.js frontend. Through this, we aimed to replicate real-world practices and prepare for handling diverse client requirements.

API Integration Process:

1. Overview:

We used the provided API for Template 6:
API URL: https://template6-six.vercel.app/api/products

The API provided product details such as titles, images, prices, and descriptions. These details were migrated to Sanity CMS and then fetched to display on the frontend.



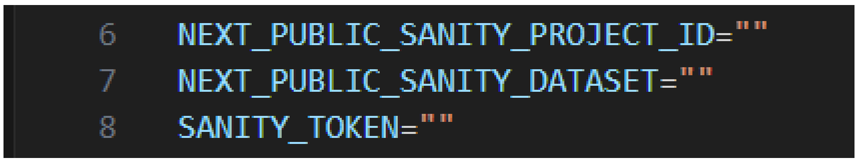
```
async function importProducts() {
  try {
    const response = await fetch('https://template6-six.vercel.app/api/products');

    if (!response.ok) {
      throw new Error(`HTTP error! Status: ${response.status}`);
    }

    const products = await response.json();

    for (const product of products) {
      await uploadProduct(product);
    }
  } catch (error) {
    console.error('Error fetching products:', error);
  }
}

importProducts();
```

2. Steps Taken:

- **Environment variables:**

· We securely stored sensitive data in .env.local to avoid hardcoding values.

· The following variables were used:



```
NEXT_PUBLIC_SANITY_PROJECT_ID=""
NEXT_PUBLIC_SANITY_DATASET=""
SANITY_TOKEN=""
```

- **Sanity Client Creation:**
  - Configured the Sanity client using the project ID and dataset in the Next.js project.
  - Ensured secure handling of sensitive data using .env files.

- **Data Fetching:**
  - Used GROQ queries to fetch products from Sanity CMS.
  - Queried fields like _id, title, productImage, price, originalPrice, discountPercentage, and description.

- **Data Processing:**
  - Processed the fetched data to align with the frontend requirements.
  - Used the urlFor function to generate image URLs dynamically.

- **Sanity Documentation Creation:**
  - Created a schema in Sanity CMS to align with the API structure.
  - Fields included title, price, originalPrice, discountPercentage, isNew, tags, and description.

- **Error Handling:**
  - Added error handling during API calls and data fetching.
  - Logged errors for debugging and displayed user-friendly messages in the frontend.

3. Migration Steps and Tools Used:

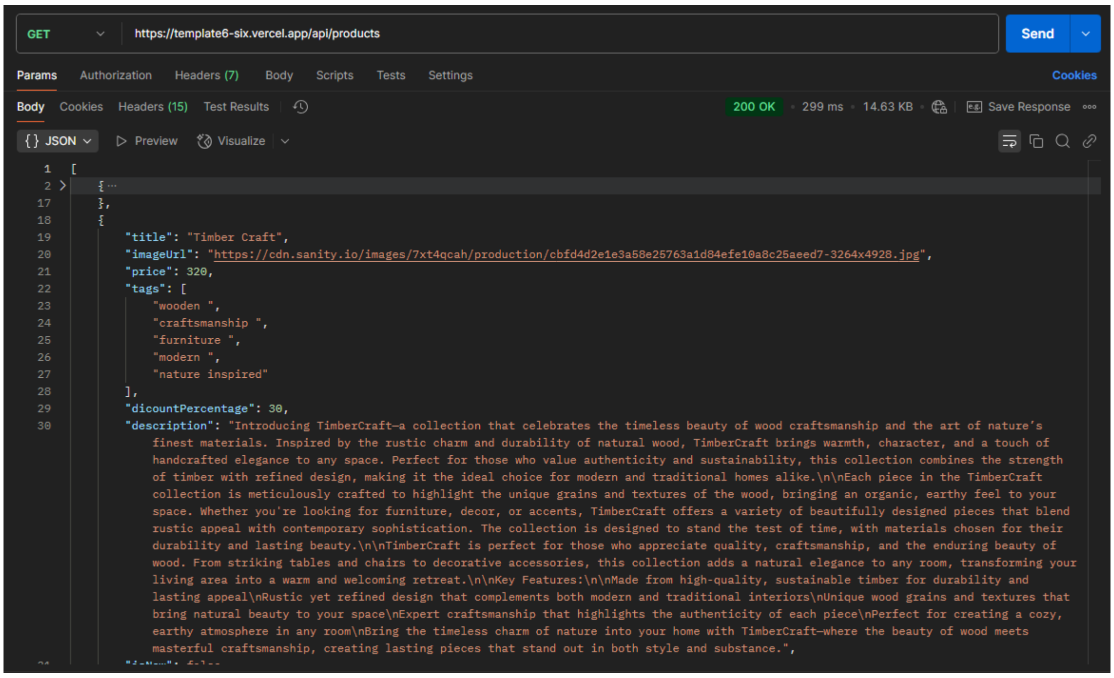
- **Migration Script:**
  - Used a script to fetch data from the API and populate Sanity CMS programmatically.
  - Ensured data validation during the migration process.

- **Sanity Schema:**
  - The schema was adjusted to match the API fields to ensure seamless migration and integration.

- **Frontend Integration:**
  - Integrated the data into the Next.js frontend using dynamic rendering.
  - Implemented a loading state for better user experience.

4. Screenshots:

API Calls:

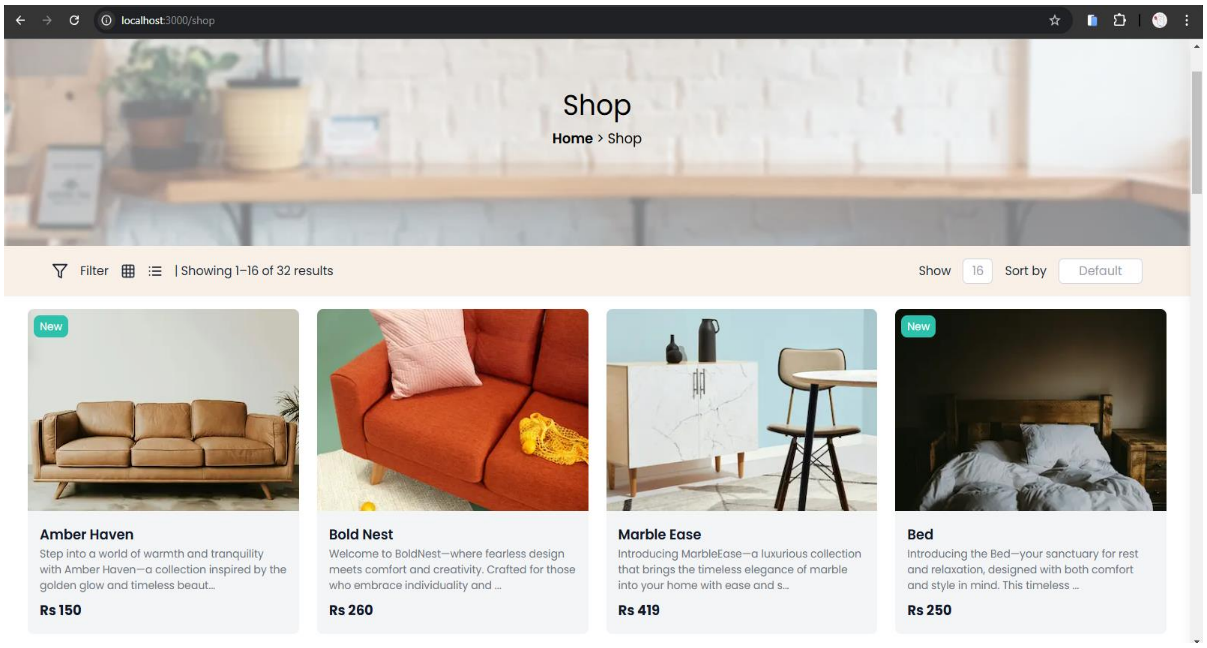


GET https://template6-six.vercel.app/api/products

Send

```
[
    {…
    },
    {
        "title": "Timber Craft",
        "imageUrl": "https://cdn.sanity.io/images/7xt4qcah/production/cbfd4d2e1e3a58e25763a1d84efe10a8c25aeed7-3264x4928.jpg",
        "price": 320,
        "tags": [
            "wooden ",
            "craftsmanship ",
            "furniture ",
            "modern ",
            "nature inspired"
        ],
        "dicountPercentage": 30,
        "description": "Introducing TimberCraft—a collection that celebrates the timeless beauty of wood craftsmanship and the art of nature's
            finest materials. Inspired by the rustic charm and durability of natural wood, TimberCraft brings warmth, character, and a touch of
            handcrafted elegance to any space. Perfect for those who value authenticity and sustainability, this collection combines the strength
            of timber with refined design, making it the ideal choice for modern and traditional homes alike.\n\nEach piece in the TimberCraft
            collection is meticulously crafted to highlight the unique grains and textures of the wood, bringing an organic, earthy feel to your
            space. Whether you're looking for furniture, decor, or accents, TimberCraft offers a variety of beautifully designed pieces that blend
            rustic appeal with contemporary sophistication. The collection is designed to stand the test of time, with materials chosen for their
            durability and lasting beauty.\n\nTimberCraft is perfect for those who appreciate quality, craftsmanship, and the enduring beauty of
            wood. From striking tables and chairs to decorative accessories, this collection adds a natural elegance to any room, transforming your
            living area into a warm and welcoming retreat.\n\nKey Features:\n\nMade from high-quality, sustainable timber for durability and
            lasting appeal\nRustic yet refined design that complements both modern and traditional interiors\nUnique wood grains and textures that
            bring natural beauty to your space\nExpert craftsmanship that highlights the authenticity of each piece\nPerfect for creating a cozy,
            earthy atmosphere in any room\nBring the timeless charm of nature into your home with TimberCraft—where the beauty of wood meets
            masterful craftsmanship, creating lasting pieces that stand out in both style and substance.",
        "isNew": false
```

Frontend Display:



localhost:3000/shop

| Showing 1–16 of 32 results

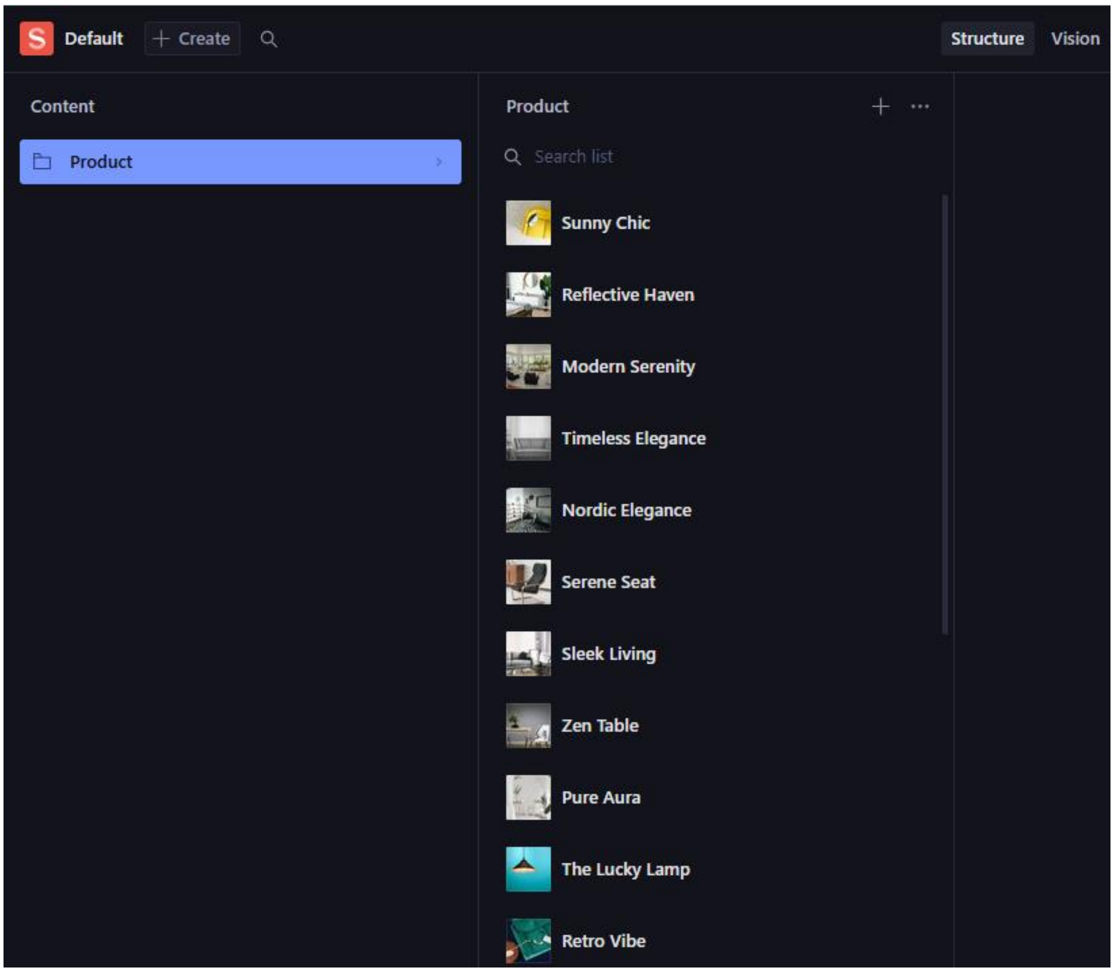
Amber Haven
Step into a world of warmth and tranquility with Amber Haven—a collection inspired by the golden glow and timeless beaut...
Rs 150

Bold Nest
Welcome to BoldNest—where fearless design meets comfort and creativity. Crafted for those who embrace individuality and ...
Rs 260

Marble Ease
Introducing MarbleEase—a luxurious collection that brings the timeless elegance of marble into your home with ease and s...
Rs 419

Bed
Introducing the Bed—your sanctuary for rest and relaxation, designed with both comfort and style in mind. This timeless ...
Rs 250

Sanity Fields:

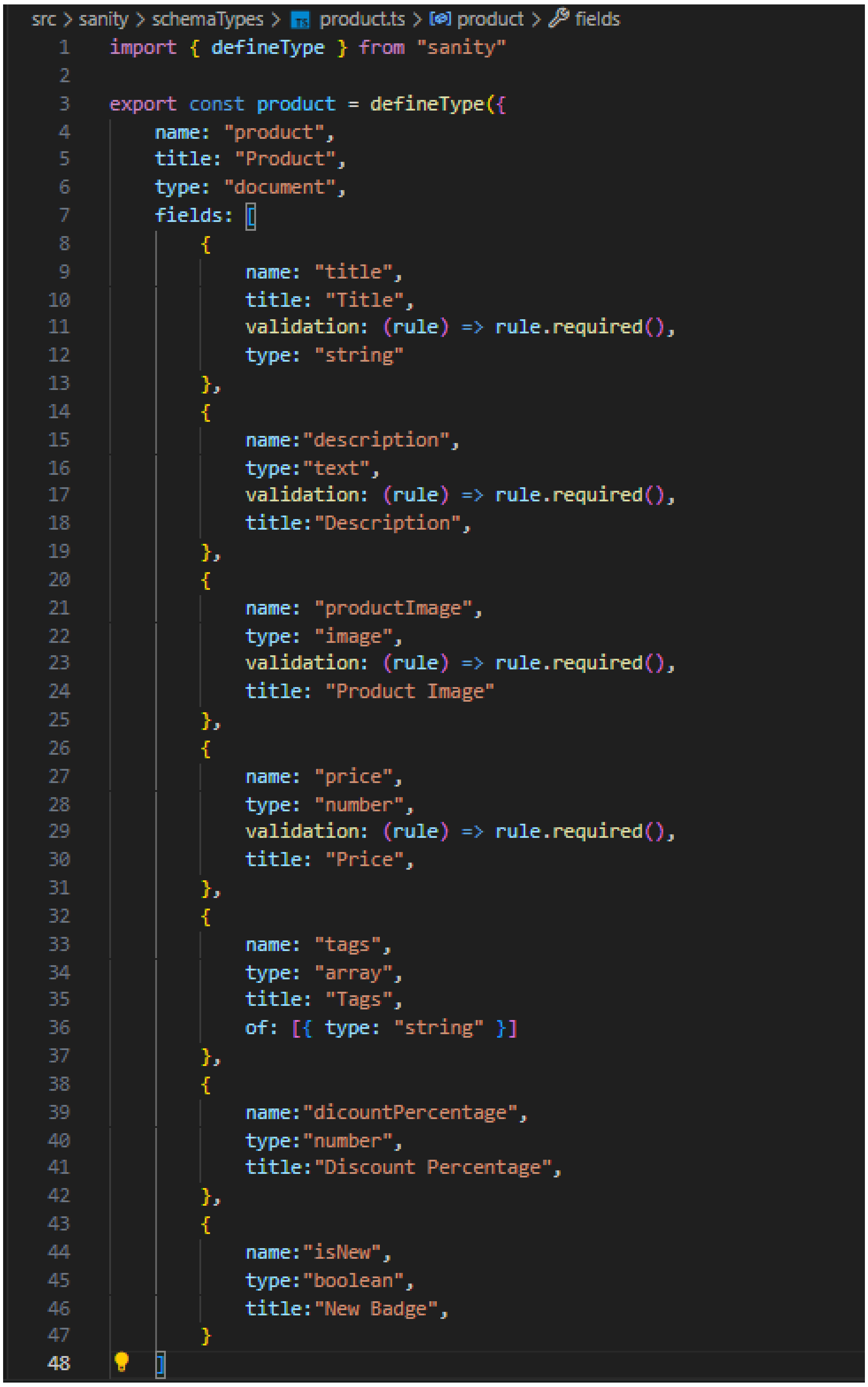


Sunny Chic
Reflective Haven
Modern Serenity
Timeless Elegance
Nordic Elegance
Serene Seat
Sleek Living
Zen Table
Pure Aura
The Lucky Lamp
Retro Vibe

Code Snippets:

1. Sanity Schema

The schema used in Sanity CMS for storing product data:

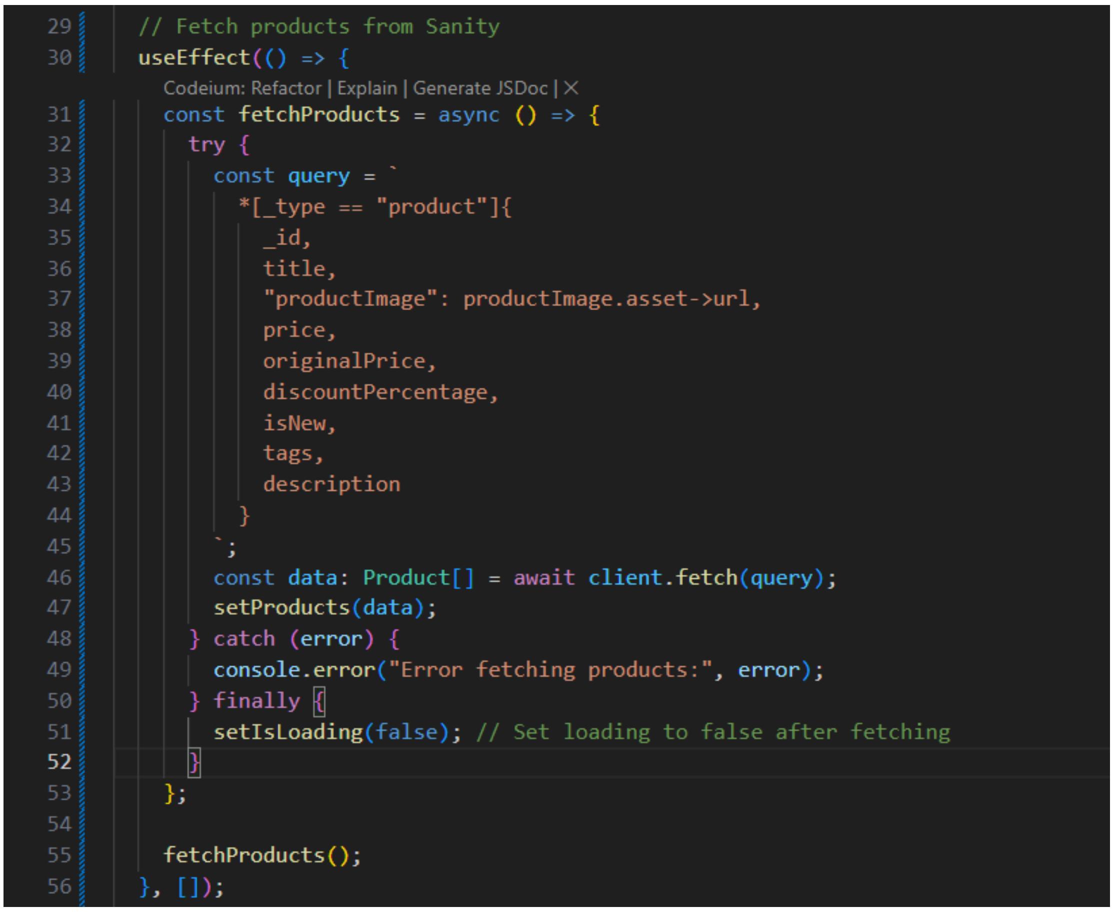


src > sanity > schemaTypes > product.ts > product > fields

```ts
import { defineType } from "sanity"

export const product = defineType({
    name: "product",
    title: "Product",
    type: "document",
    fields: [
        {
            name: "title",
            title: "Title",
            validation: (rule) => rule.required(),
            type: "string"
        },
        {
            name:"description",
            type:"text",
            validation: (rule) => rule.required(),
            title:"Description",
        },
        {
            name: "productImage",
            type: "image",
            validation: (rule) => rule.required(),
            title: "Product Image"
        },
        {
            name: "price",
            type: "number",
            validation: (rule) => rule.required(),
            title: "Price",
        },
        {
            name: "tags",
            type: "array",
            title: "Tags",
            of: [{ type: "string" }]
        },
        {
            name:"dicountPercentage",
            type:"number",
            title:"Discount Percentage",
        },
        {
            name:"isNew",
            type:"boolean",
            title:"New Badge",
        }
    ]
```

2. GROQ Query for Frontend Integration:

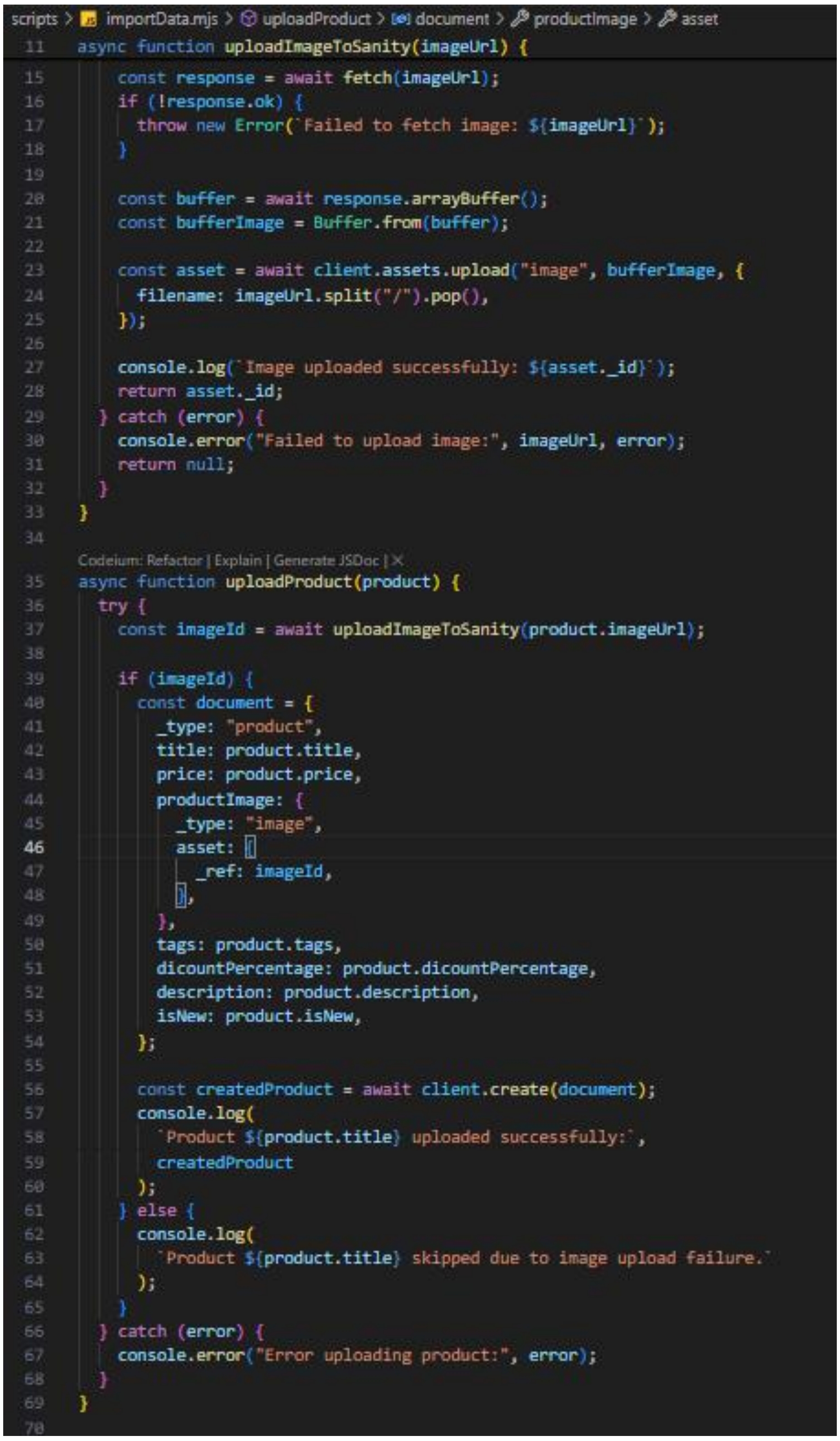
The query used to fetch products from Sanity CMS for rendering in the frontend:



```
// Fetch products from Sanity
useEffect(() => {
  Codeium: Refactor | Explain | Generate JSDoc | ×
  const fetchProducts = async () => {
    try {
      const query = `
        *[_type == "product"]{
          _id,
          title,
          "productImage": productImage.asset->url,
          price,
          originalPrice,
          discountPercentage,
          isNew,
          tags,
          description
        }
      `;
      const data: Product[] = await client.fetch(query);
      setProducts(data);
    } catch (error) {
      console.error("Error fetching products:", error);
    } finally {
      setIsLoading(false); // Set loading to false after fetching
    }
  };

  fetchProducts();
}, []);
```

3. Migration Script

The script used to fetch product data from the API and populate Sanity CMS:


scripts > importData.mjs > uploadProduct > document > productImage > asset
Codeium: Refactor | Explain | Generate JSDoc | ×

```
async function uploadImageToSanity(imageUrl) {
    const response = await fetch(imageUrl);
    if (!response.ok) {
      throw new Error(`Failed to fetch image: ${imageUrl}`);
    }

    const buffer = await response.arrayBuffer();
    const bufferImage = Buffer.from(buffer);

    const asset = await client.assets.upload("image", bufferImage, {
      filename: imageUrl.split("/").pop(),
    });

    console.log(`Image uploaded successfully: ${asset._id}`);
    return asset._id;
  } catch (error) {
    console.error("Failed to upload image:", imageUrl, error);
    return null;
  }
}

async function uploadProduct(product) {
  try {
    const imageId = await uploadImageToSanity(product.imageUrl);

    if (imageId) {
      const document = {
        _type: "product",
        title: product.title,
        price: product.price,
        productImage: {
          _type: "image",
          asset: {
            _ref: imageId,
          },
        },
        tags: product.tags,
        dicountPercentage: product.dicountPercentage,
        description: product.description,
        isNew: product.isNew,
      };

      const createdProduct = await client.create(document);
      console.log(
        `Product ${product.title} uploaded successfully:`,
        createdProduct
      );
    } else {
      console.log(
        `Product ${product.title} skipped due to image upload failure.`
      );
    }
  } catch (error) {
    console.error("Error uploading product:", error);
  }
}
```

Final Checklist:

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