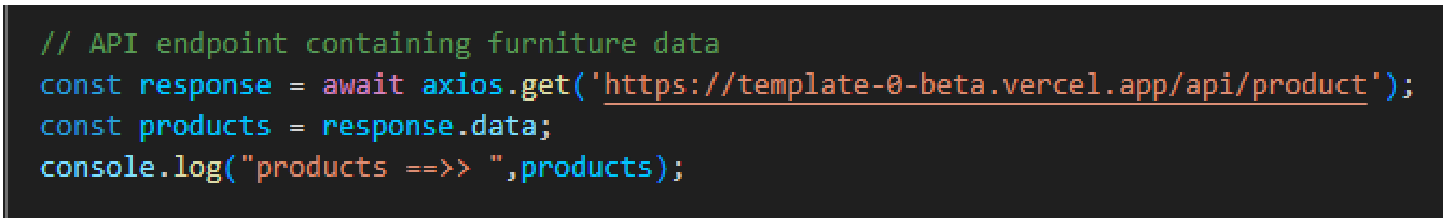
"Day 3 - API Integration Report - [Niche-e-commerce furniture]"

1. API Integration Process

The API integration process involved the following key steps:

1. **Initial Data Retrieval Using MOCK API**:
    - A MOCK API was utilized to fetch furniture-related data during the development phase.
    - The API provided essential product attributes, including names, descriptions, prices, dimensions, and images.



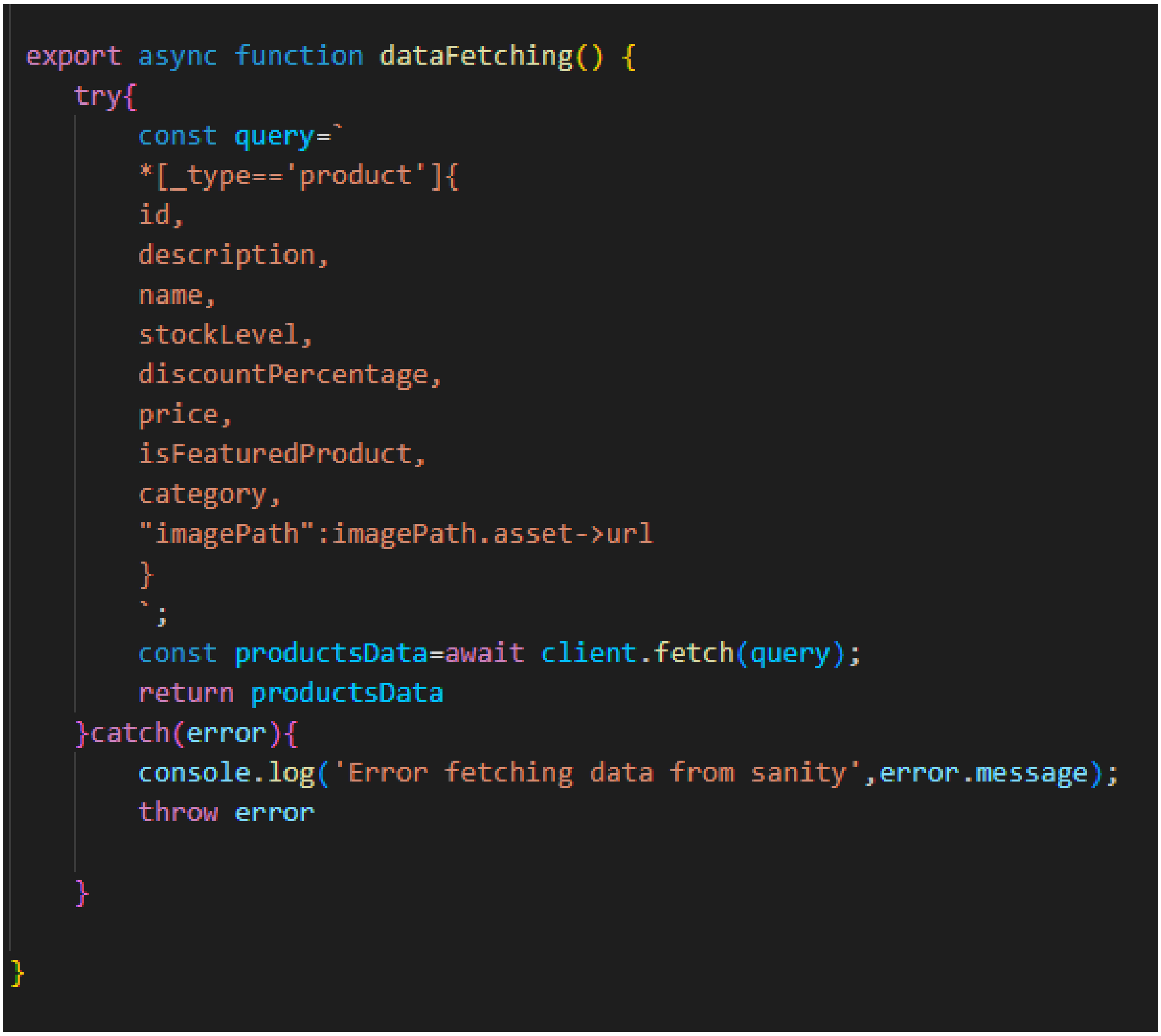
```
// API endpoint containing furniture data
const response = await axios.get('https://template-0-beta.vercel.app/api/product');
const products = response.data;
console.log("products ==>> ",products);
```

2.Migration to Sanity CMS:

- First, we created a new Sanity project and configured it in our Next.js project.
- All sensitive information, such as project ID and dataset, was stored in the .env.local file to ensure security.
- Data retrieved from the MOCK API was then migrated to Sanity CMS through migration script for better management and scalability.

3.Data Retrieval from Sanity Using GROQ Query:

- A GROQ query was written to fetch data from Sanity CMS to render it dynamically on the frontend.

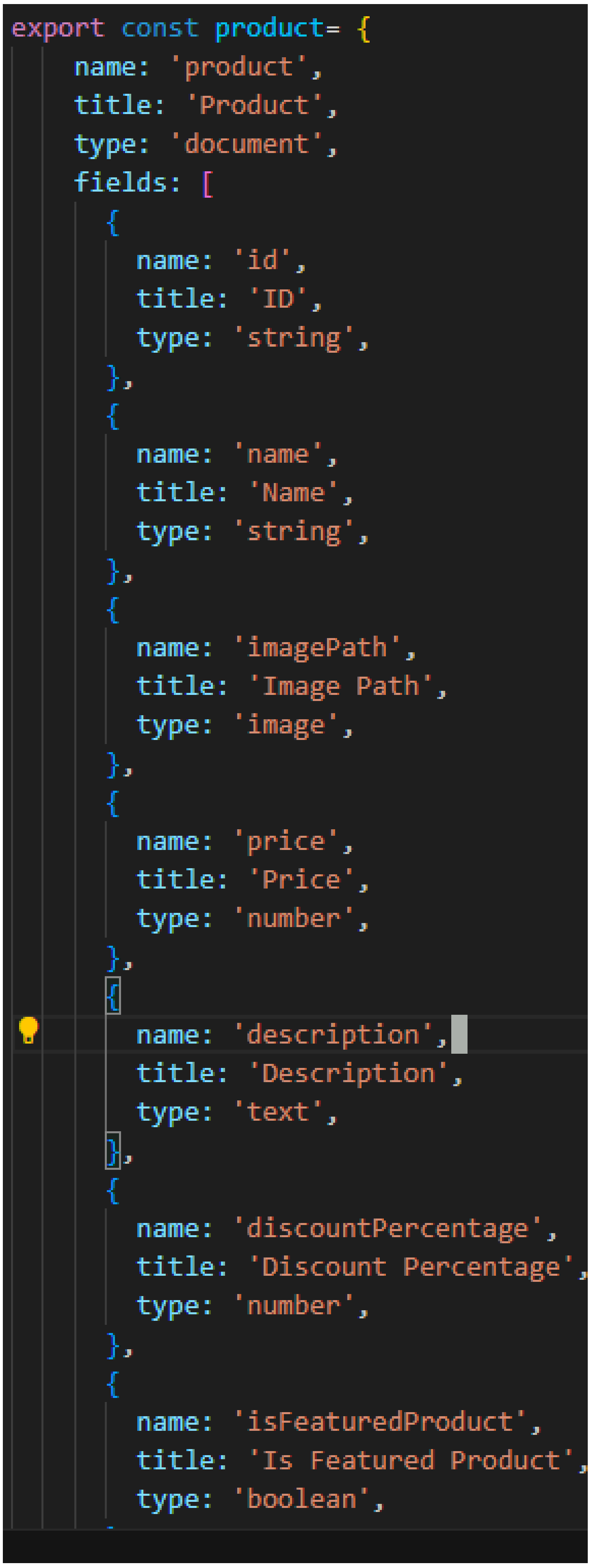


```
export async function dataFetching() {
    try{
        const query=`
        *[_type=='product']{
        id,
        description,
        name,
        stockLevel,
        discountPercentage,
        price,
        isFeaturedProduct,
        category,
        "imagePath":imagePath.asset->url
        }
        `;
        const productsData=await client.fetch(query);
        return productsData
    }catch(error){
        console.log('Error fetching data from sanity',error.message);
        throw error

    }

}
```

This query ensured that all relevant data fields were seamlessly fetched and displayed in the user interface.

2. Adjustments Made to Schemas

We thoroughly matched our Sanity schema to the data provided by the MOCK API to ensure a seamless workflow. The schema was designed to efficiently store and query all necessary furniture-related attributes, including titles, descriptions, prices, dimensions, and images etc.


```
export const product= {
    name: 'product',
    title: 'Product',
    type: 'document',
    fields: [
      {
        name: 'id',
        title: 'ID',
        type: 'string',
      },
      {
        name: 'name',
        title: 'Name',
        type: 'string',
      },
      {
        name: 'imagePath',
        title: 'Image Path',
        type: 'image',
      },
      {
        name: 'price',
        title: 'Price',
        type: 'number',
      },
      {
        name: 'description',
        title: 'Description',
        type: 'text',
      },
      {
        name: 'discountPercentage',
        title: 'Discount Percentage',
        type: 'number',
      },
      {
        name: 'isFeaturedProduct',
        title: 'Is Featured Product',
        type: 'boolean',
```

3. Migration Steps and Tools Used

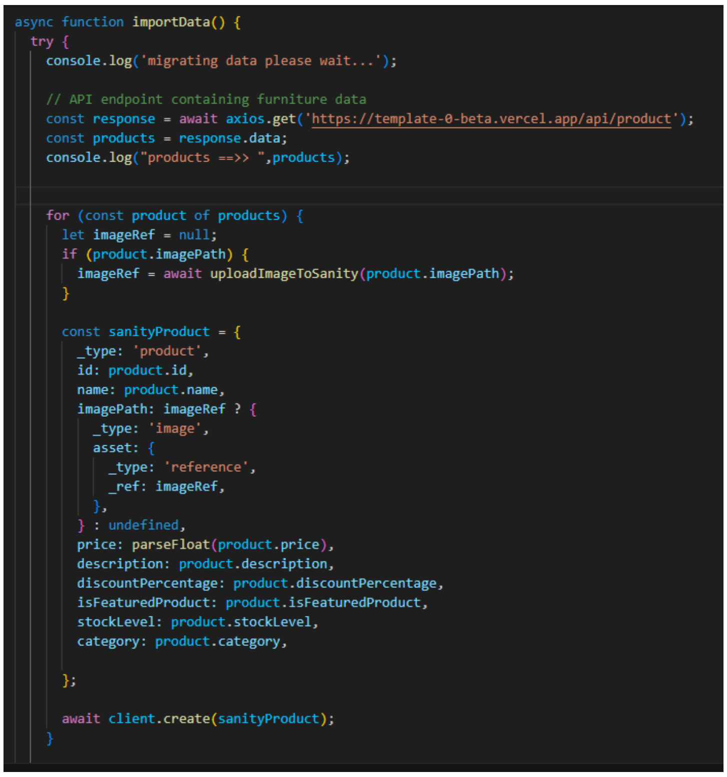
Migration Steps:

1. **Data Mapping**:
   - The structure of the MOCK API data was analyzed and mapped to match the custom schemas in Sanity CMS.
2. **Data Transformation**:
   - A script was written to transform the MOCK API data into the required format for Sanity.
3. **Bulk Upload**:
   - The Sanity JavaScript client was used to programmatically upload the transformed data.

Tools Used:

- **Sanity JavaScript Client** for API integration.
- **JavaScript/Node.js** for scripting the data transformation and migration.

Migration script :



```javascript
async function importData() {
  try {
    console.log('migrating data please wait...');

    // API endpoint containing furniture data
    const response = await axios.get('https://template-0-beta.vercel.app/api/product');
    const products = response.data;
    console.log("products ==>> ",products);


    for (const product of products) {
      let imageRef = null;
      if (product.imagePath) {
        imageRef = await uploadImageToSanity(product.imagePath);
      }

      const sanityProduct = {
        _type: 'product',
        id: product.id,
        name: product.name,
        imagePath: imageRef ? {
          _type: 'image',
          asset: {
            _type: 'reference',
            _ref: imageRef,
          },
        } : undefined,
        price: parseFloat(product.price),
        description: product.description,
        discountPercentage: product.discountPercentage,
        isFeaturedProduct: product.isFeaturedProduct,
        stockLevel: product.stockLevel,
        category: product.category,

      };

      await client.create(sanityProduct);
    }
```

Conclusion :

This report documents the successful integration of a MOCK API, migration to Sanity CMS, and data retrieval for the frontend. Screenshots and code snippets validate the processes followed, ensuring a scalable and efficient e-commerce platform.