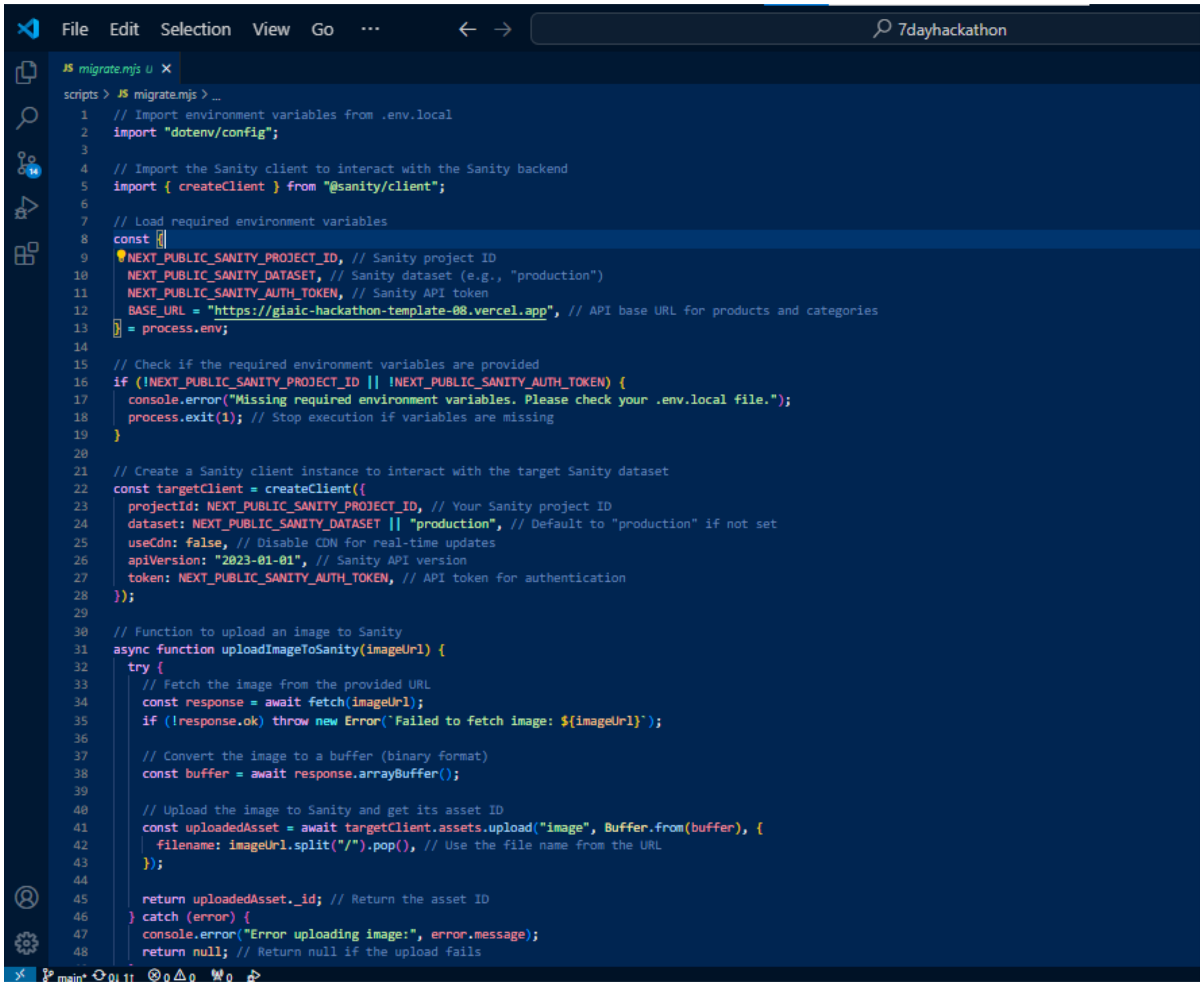
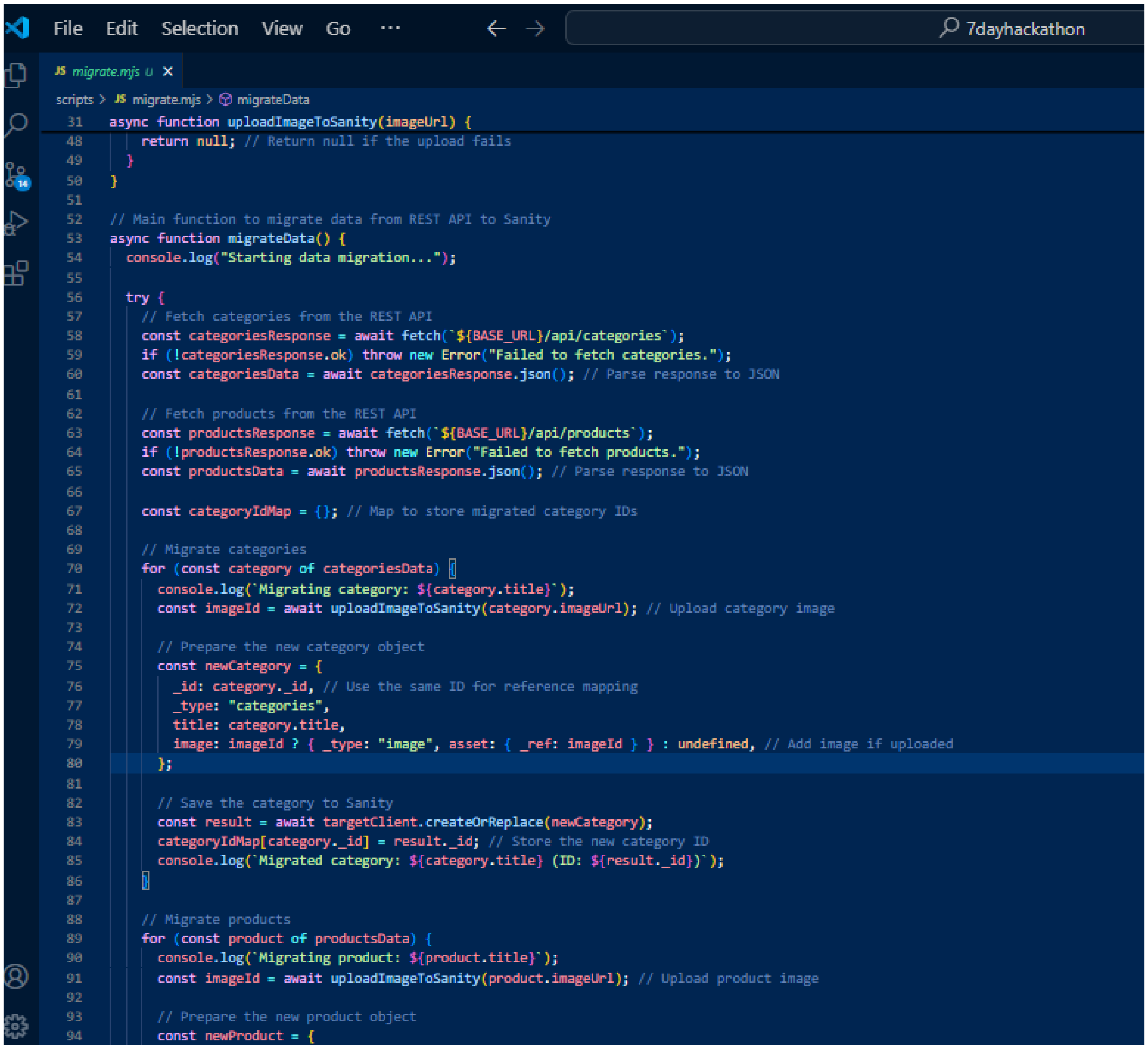
DAY03: API INTEGRATION AND DATA MIGRATION



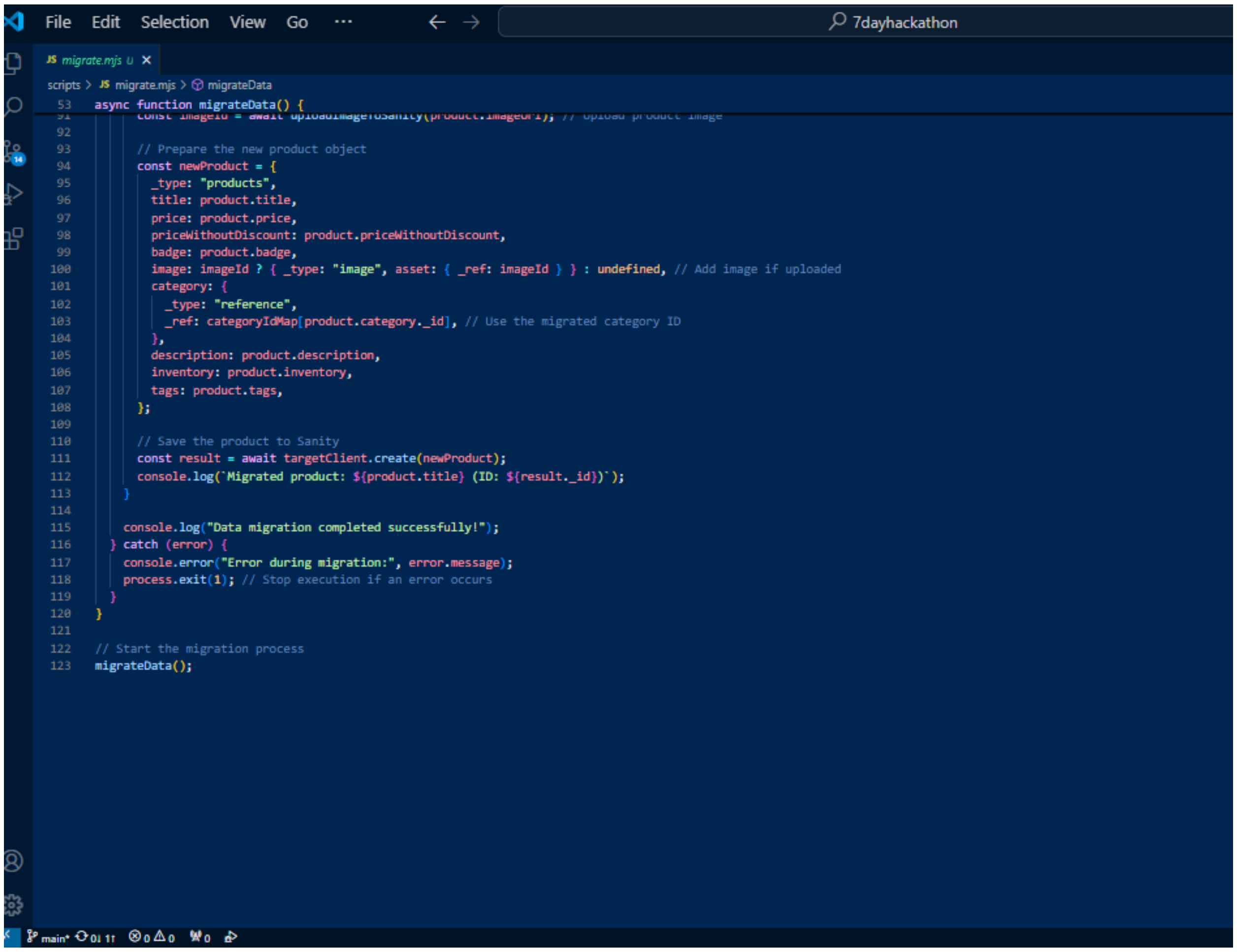
```javascript
// Import environment variables from .env.local
import "dotenv/config";

// Import the Sanity client to interact with the Sanity backend
import { createClient } from "@sanity/client";

// Load required environment variables
const {
  NEXT_PUBLIC_SANITY_PROJECT_ID, // Sanity project ID
  NEXT_PUBLIC_SANITY_DATASET, // Sanity dataset (e.g., "production")
  NEXT_PUBLIC_SANITY_AUTH_TOKEN, // Sanity API token
  BASE_URL = "https://giaic-hackathon-template-08.vercel.app", // API base URL for products and categories
} = process.env;

// Check if the required environment variables are provided
if (!NEXT_PUBLIC_SANITY_PROJECT_ID || !NEXT_PUBLIC_SANITY_AUTH_TOKEN) {
  console.error("Missing required environment variables. Please check your .env.local file.");
  process.exit(1); // Stop execution if variables are missing
}

// Create a Sanity client instance to interact with the target Sanity dataset
const targetClient = createClient({
  projectId: NEXT_PUBLIC_SANITY_PROJECT_ID, // Your Sanity project ID
  dataset: NEXT_PUBLIC_SANITY_DATASET || "production", // Default to "production" if not set
  useCdn: false, // Disable CDN for real-time updates
  apiVersion: "2023-01-01", // Sanity API version
  token: NEXT_PUBLIC_SANITY_AUTH_TOKEN, // API token for authentication
});

// Function to upload an image to Sanity
async function uploadImageToSanity(imageUrl) {
  try {
    // Fetch the image from the provided URL
    const response = await fetch(imageUrl);
    if (!response.ok) throw new Error(`Failed to fetch image: ${imageUrl}`);

    // Convert the image to a buffer (binary format)
    const buffer = await response.arrayBuffer();

    // Upload the image to Sanity and get its asset ID
    const uploadedAsset = await targetClient.assets.upload("image", Buffer.from(buffer), {
      filename: imageUrl.split("/").pop(), // Use the file name from the URL
    });

    return uploadedAsset._id; // Return the asset ID
  } catch (error) {
    console.error("Error uploading image:", error.message);
    return null; // Return null if the upload fails
```

DAY03: API INTEGRATION AND DATA MIGRATION



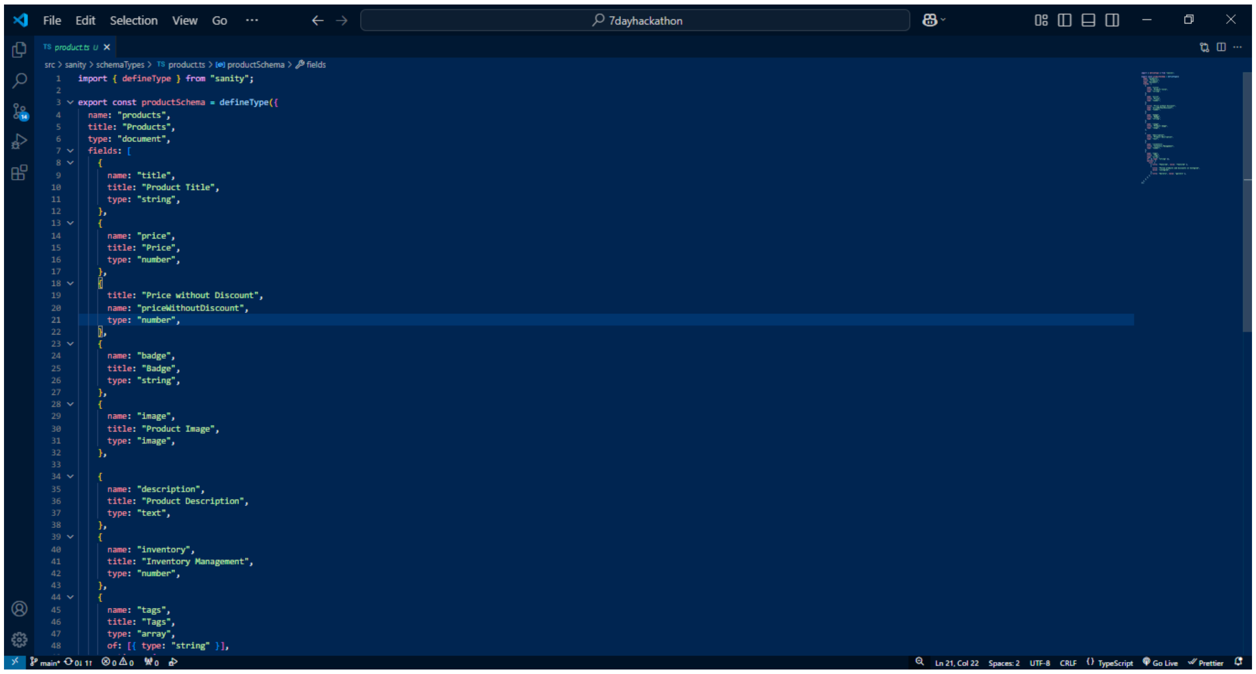
```javascript
async function uploadImageToSanity(imageUrl) {
    return null; // Return null if the upload fails
  }
}

// Main function to migrate data from REST API to Sanity
async function migrateData() {
  console.log("Starting data migration...");

  try {
    // Fetch categories from the REST API
    const categoriesResponse = await fetch(`${BASE_URL}/api/categories`);
    if (!categoriesResponse.ok) throw new Error("Failed to fetch categories.");
    const categoriesData = await categoriesResponse.json(); // Parse response to JSON

    // Fetch products from the REST API
    const productsResponse = await fetch(`${BASE_URL}/api/products`);
    if (!productsResponse.ok) throw new Error("Failed to fetch products.");
    const productsData = await productsResponse.json(); // Parse response to JSON

    const categoryIdMap = {}; // Map to store migrated category IDs

    // Migrate categories
    for (const category of categoriesData) {
      console.log(`Migrating category: ${category.title}`);
      const imageId = await uploadImageToSanity(category.imageUrl); // Upload category image

      // Prepare the new category object
      const newCategory = {
        _id: category._id, // Use the same ID for reference mapping
        _type: "categories",
        title: category.title,
        image: imageId ? { _type: "image", asset: { _ref: imageId } } : undefined, // Add image if uploaded
      };

      // Save the category to Sanity
      const result = await targetClient.createOrReplace(newCategory);
      categoryIdMap[category._id] = result._id; // Store the new category ID
      console.log(`Migrated category: ${category.title} (ID: ${result._id})`);
    }

    // Migrate products
    for (const product of productsData) {
      console.log(`Migrating product: ${product.title}`);
      const imageId = await uploadImageToSanity(product.imageUrl); // Upload product image

      // Prepare the new product object
      const newProduct = {
```

DAY03: API INTEGRATION AND DATA MIGRATION


File Edit Selection View Go ···
7dayhackathon
migrate.mjs U
scripts > migrate.mjs > migrateData
main*

```javascript
async function migrateData() {
      const imageId = await uploadImageToSanity(product.imageUrl); // Upload product image

      // Prepare the new product object
      const newProduct = {
        _type: "products",
        title: product.title,
        price: product.price,
        priceWithoutDiscount: product.priceWithoutDiscount,
        badge: product.badge,
        image: imageId ? { _type: "image", asset: { _ref: imageId } } : undefined, // Add image if uploaded
        category: {
          _type: "reference",
          _ref: categoryIdMap[product.category._id], // Use the migrated category ID
        },
        description: product.description,
        inventory: product.inventory,
        tags: product.tags,
      };

      // Save the product to Sanity
      const result = await targetClient.create(newProduct);
      console.log(`Migrated product: ${product.title} (ID: ${result._id})`);
    }

    console.log("Data migration completed successfully!");
  } catch (error) {
    console.error("Error during migration:", error.message);
    process.exit(1); // Stop execution if an error occurs
  }
}

// Start the migration process
migrateData();
```

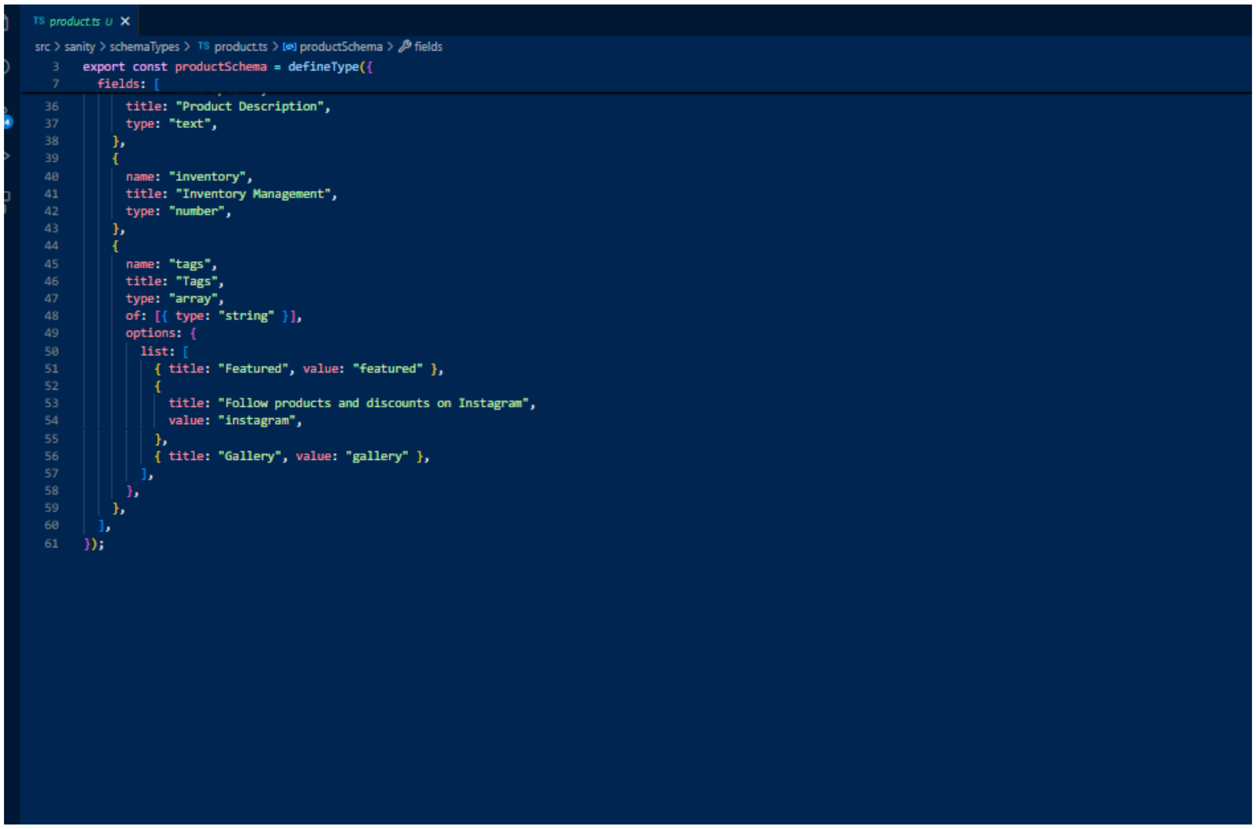
DAY03: API INTEGRATION AND DATA MIGRATION



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

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

export const productSchema = defineType({
  name: "products",
  title: "Products",
  type: "document",
  fields: [
    {
      name: "title",
      title: "Product Title",
      type: "string",
    },
    {
      name: "price",
      title: "Price",
      type: "number",
    },
    {
      title: "Price without Discount",
      name: "priceWithoutDiscount",
      type: "number",
    },
    {
      name: "badge",
      title: "Badge",
      type: "string",
    },
    {
      name: "image",
      title: "Product Image",
      type: "image",
    },

    {
      name: "description",
      title: "Product Description",
      type: "text",
    },
    {
      name: "inventory",
      title: "Inventory Management",
      type: "number",
    },
    {
      name: "tags",
      title: "Tags",
      type: "array",
      of: [{ type: "string" }],
```

DAY03: API INTEGRATION AND DATA MIGRATION

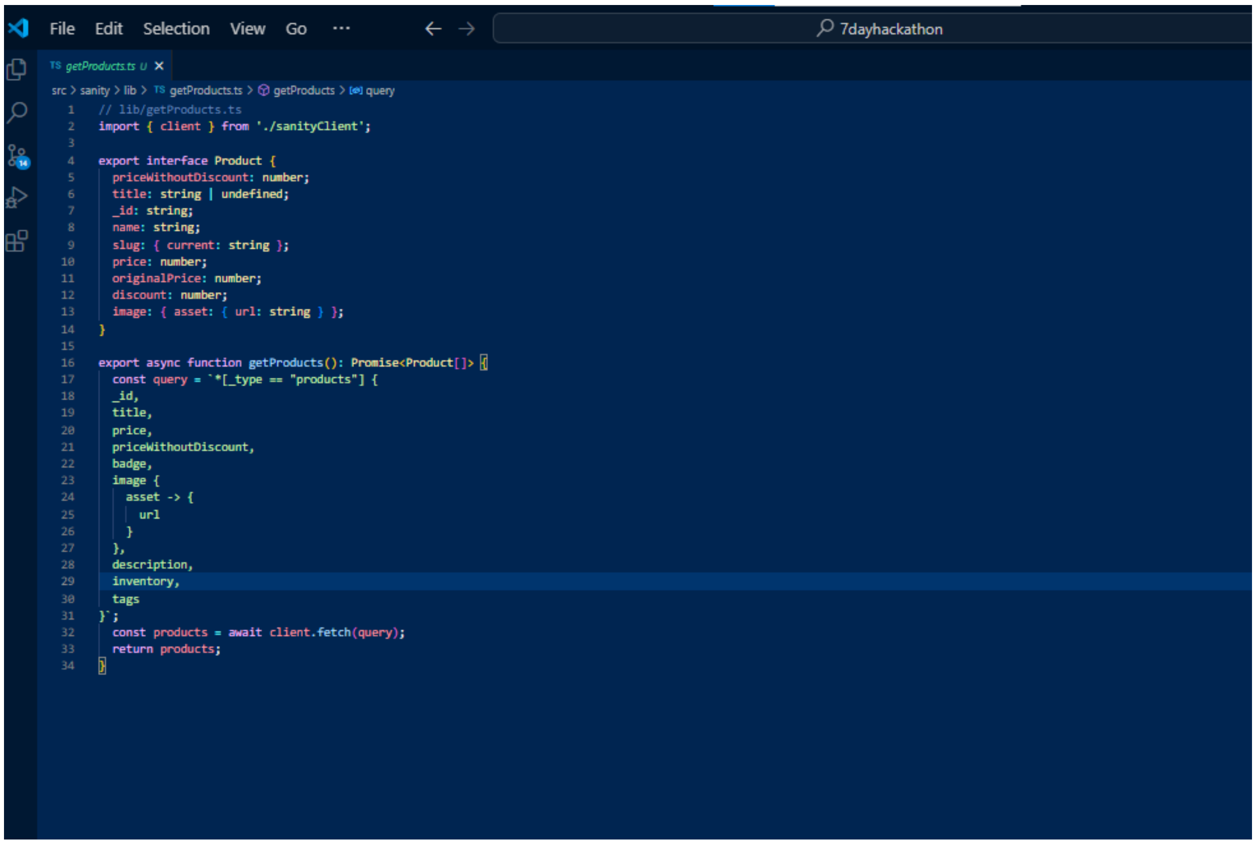


product.ts U

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

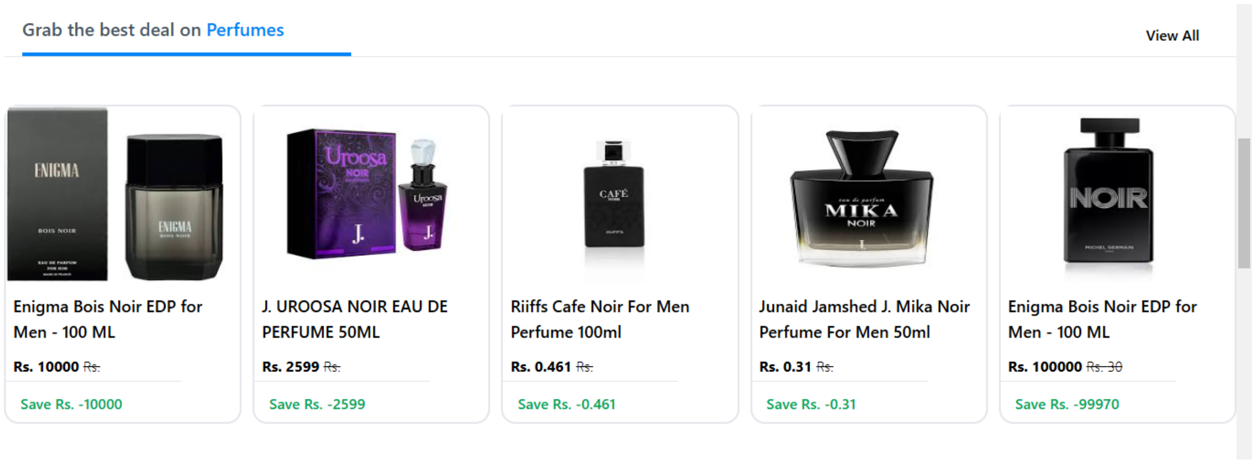
```
export const productSchema = defineType({
  fields: [
        title: "Product Description",
        type: "text",
      },
      {
        name: "inventory",
        title: "Inventory Management",
        type: "number",
      },
      {
        name: "tags",
        title: "Tags",
        type: "array",
        of: [{ type: "string" }],
        options: {
          list: [
            { title: "Featured", value: "featured" },
            {
              title: "Follow products and discounts on Instagram",
              value: "instagram",
            },
            { title: "Gallery", value: "gallery" },
          ],
        },
      },
    ],
});
```

DAY03: API INTEGRATION AND DATA MIGRATION


File Edit Selection View Go ···
7dayhackathon
getProducts.ts U
src > sanity > lib > getProducts.ts > getProducts > query

```ts
// lib/getProducts.ts
import { client } from './sanityClient';

export interface Product {
  priceWithoutDiscount: number;
  title: string | undefined;
  _id: string;
  name: string;
  slug: { current: string };
  price: number;
  originalPrice: number;
  discount: number;
  image: { asset: { url: string } };
}

export async function getProducts(): Promise<Product[]> {
  const query = `*[_type == "products"] {
  _id,
  title,
  price,
  priceWithoutDiscount,
  badge,
  image {
    asset -> {
      url
    }
  },
  description,
  inventory,
  tags
}`;
  const products = await client.fetch(query);
  return products;
}
```

PAGE1



Grab the best deal on Perfumes

View All

Enigma Bois Noir EDP for Men - 100 ML
Rs. 10000 ~~Rs.~~
Save Rs. -10000

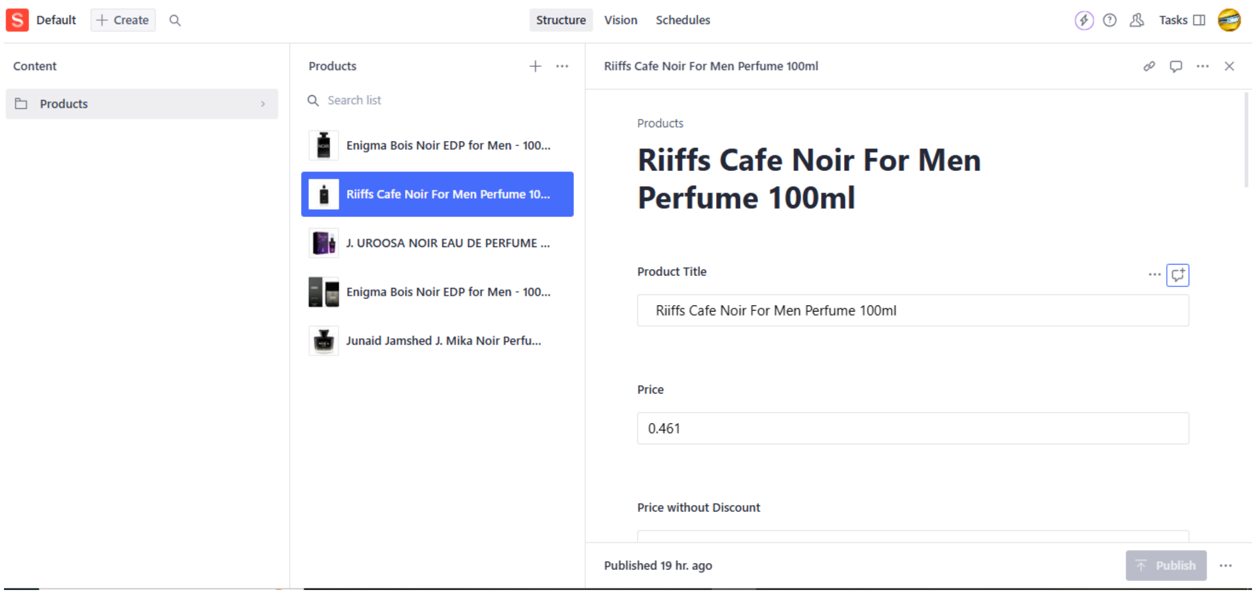
J. UROOSA NOIR EAU DE PERFUME 50ML
Rs. 2599 ~~Rs.~~
Save Rs. -2599

Riiffs Cafe Noir For Men Perfume 100ml
Rs. 0.461 ~~Rs.~~
Save Rs. -0.461

Junaid Jamshed J. Mika Noir Perfume For Men 50ml
Rs. 0.31 ~~Rs.~~
Save Rs. -0.31

Enigma Bois Noir EDP for Men - 100 ML
Rs. 100000 ~~Rs. 30~~
Save Rs. -99970

DAY03: API INTEGRATION AND DATA MIGRATION



Default + Create Structure Vision Schedules Tasks

Content

Products

Products

Search list

- Enigma Bois Noir EDP for Men - 100...
- Riiffs Cafe Noir For Men Perfume 10...
- J. UROOSA NOIR EAU DE PERFUME ...
- Enigma Bois Noir EDP for Men - 100...
- Junaid Jamshed J. Mika Noir Perfu...

Riiffs Cafe Noir For Men Perfume 100ml

Products

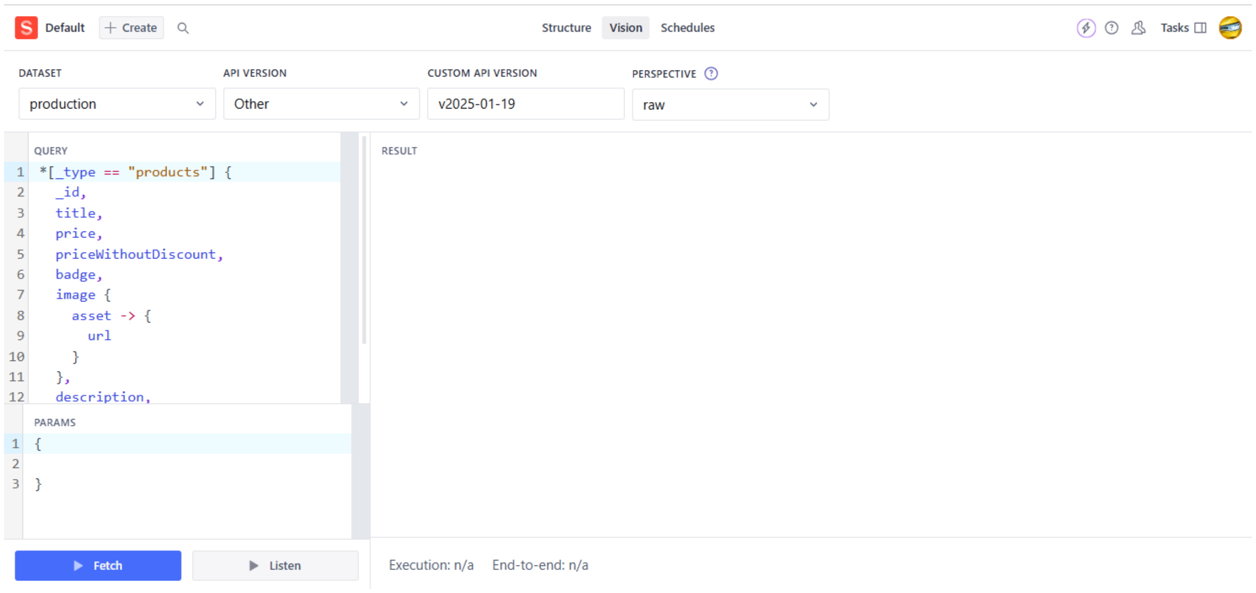
Riiffs Cafe Noir For Men Perfume 100ml

Product Title: Riiffs Cafe Noir For Men Perfume 100ml

Price: 0.461

Price without Discount

Published 19 hr. ago Publish



Default + Create Structure Vision Schedules Tasks

DATASET: production

API VERSION: Other

CUSTOM API VERSION: v2025-01-19

PERSPECTIVE: raw

QUERY

```
*[_type == "products"] {
  _id,
  title,
  price,
  priceWithoutDiscount,
  badge,
  image {
    asset -> {
      url
    }
  },
  description,
```

PARAMS

```
{

}
```

RESULT

Fetch Listen

Execution: n/a End-to-end: n/a