DAY 3 - API INTEGRATION AND DATA MIGRATION

Step-3:

Data Migration Options:

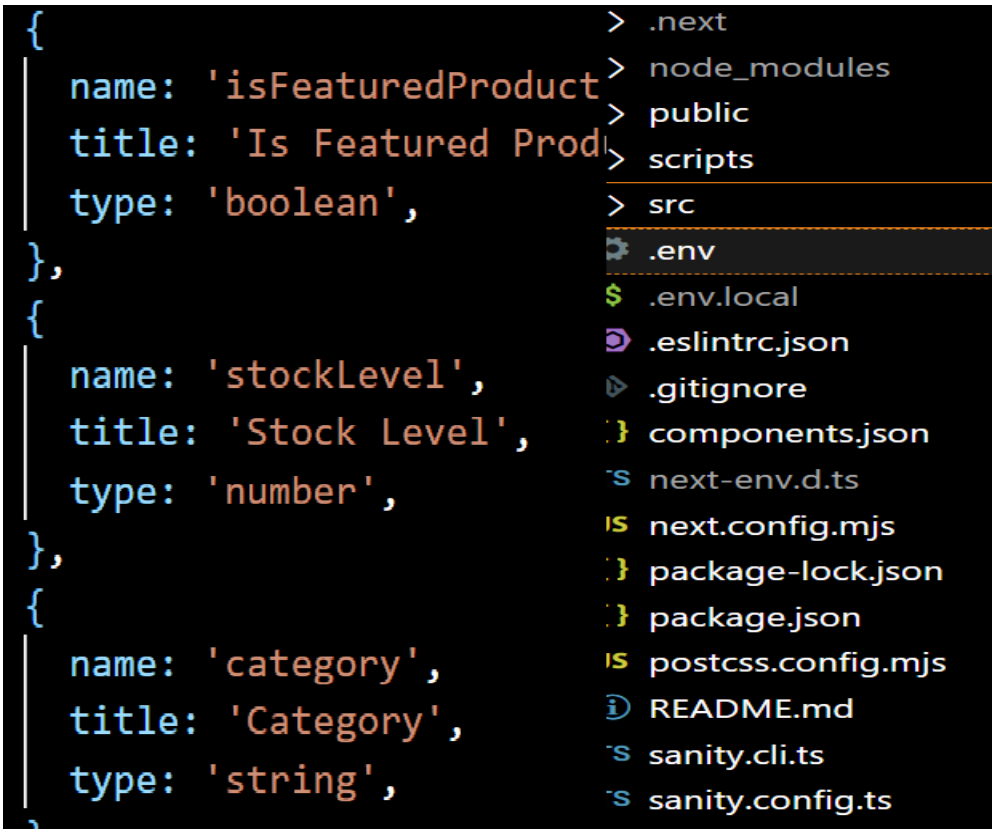
1: Make a folder at root of project .

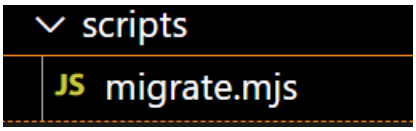
Like this here script folder is at the root of the project.

Make a file inside the script folder name like migrate.mjs.

Then copy the given code in the documentation for migration into the file.

As shown in the screen shots





And here is the script

```
// 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;
console.log(NEXT_PUBLIC_SANITY_PROJECT_ID ,"project id is consoling ")
console.log(NEXT_PUBLIC_SANITY_DATASET ,"project dataset  is consoling ")
console.log(NEXT_PUBLIC_SANITY_AUTH_TOKEN ,"project id is consoling ")
// 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
  }
}

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

  try {



    // 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









    // 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 =
```

```
_type: 'product'   id:
product id
```

```
        name: product.name, imagePath:
        product.imageId,
        price: parseFloat(product.price), description:
        product.description,
        discountPercentage: product.discountPercentage,
        isFeaturedProduct: product.isFeaturedProduct, stockLevel:
        product.stockLevel,
        category: product.category,
      };

      // 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);
    console.log("error while sending data to the sanity ")
    process.exit(1); // Stop execution if an error occurs
  }
}
```

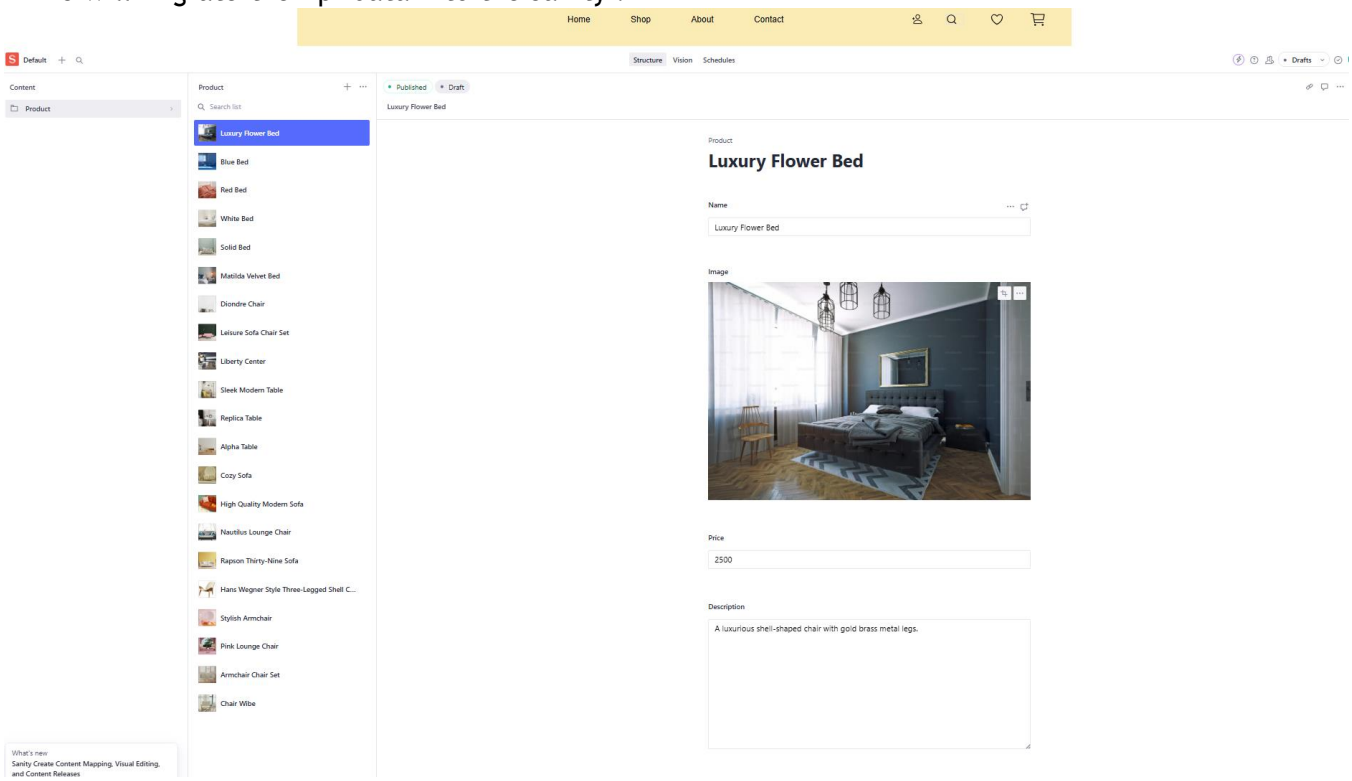
Remeber to make these changes before running this code

1:make change in the file Packadge.json

Then run npm run migrate

```
  "scripts": {
    "dev": "next dev",
    "build": "next build",
    "start": "next start",
    "lint": "next lint",
    "migrate": "node scripts/migrate.mjs"
  }
```

This will migrate the Api data into the sanity .



Like this.

Then fetch the stored data into next.js project by using GroqQueries.

*[_type == "product" CC id == '9'][0]{ name,price,description,imagePath,category,stockLevel }