Day_3_API_Integration_and_Data_Migration:
General E-Commerce Hekto Furniture website

I have faced many types of error with the given api templete and documentation

Then I have follow these steps and then finally I have resolved my issues

Table of Contents

❖ Configuring Environment Variables:

Begin by setting up your environment variables. If a .env.local file doesn't already exist in your project's root directory, create one. Then, add the following variables:

```
 NEXT_PUBLIC_SANITY_PROJECT_ID=your_project_id
NEXT_PUBLIC_SANITY_DATASET=production
SANITY_API_TOKEN=your_sanity_token
```

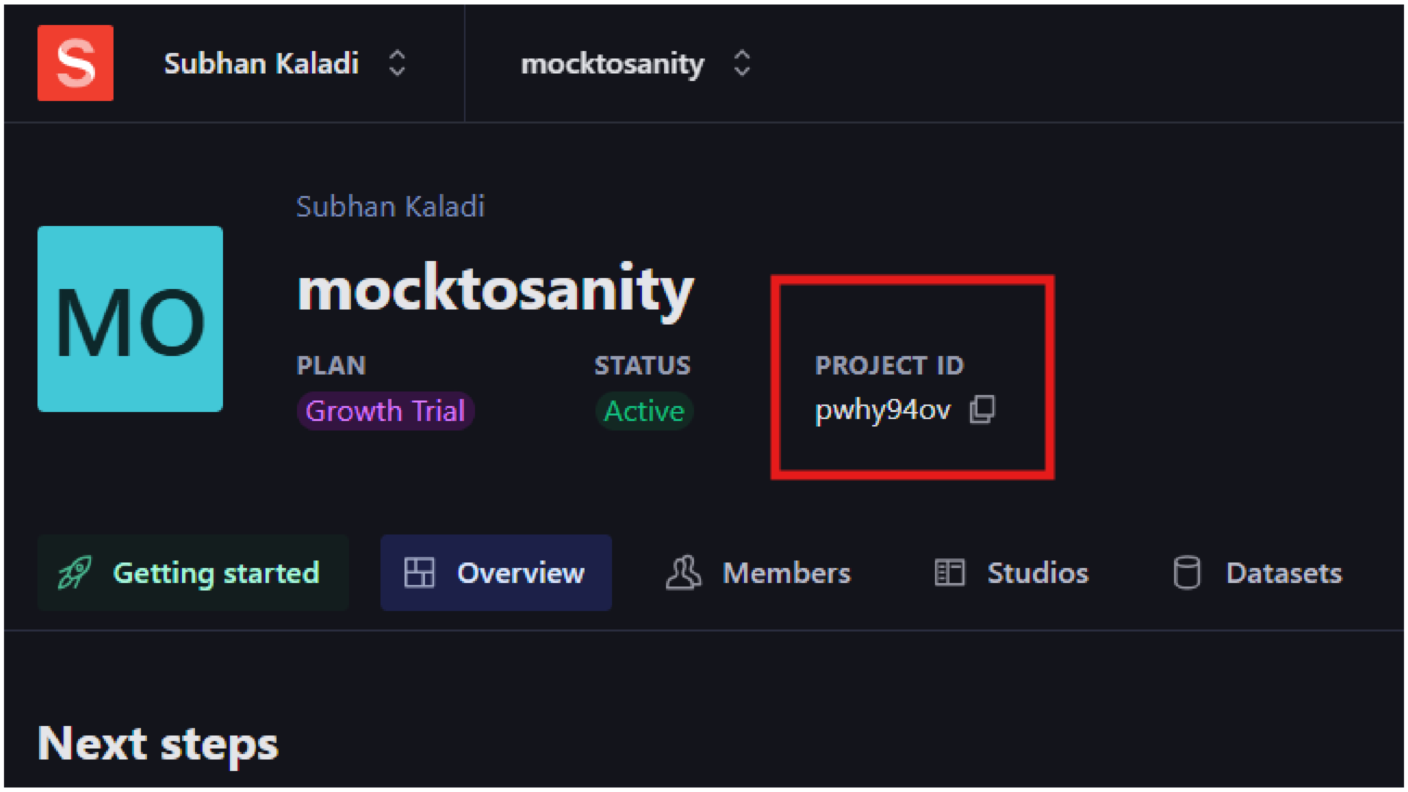
Remember, variables prefixed with NEXT_PUBLIC_ will be exposed to the browser, so be cautious about what you prefix.

❖ Fetching Sanity Project ID and API Token

Project ID

To find your Sanity project ID:

- Log in to your Sanity account at https://www.sanity.io/manage
- Select your project
- In the project dashboard, you'll see the project ID listed

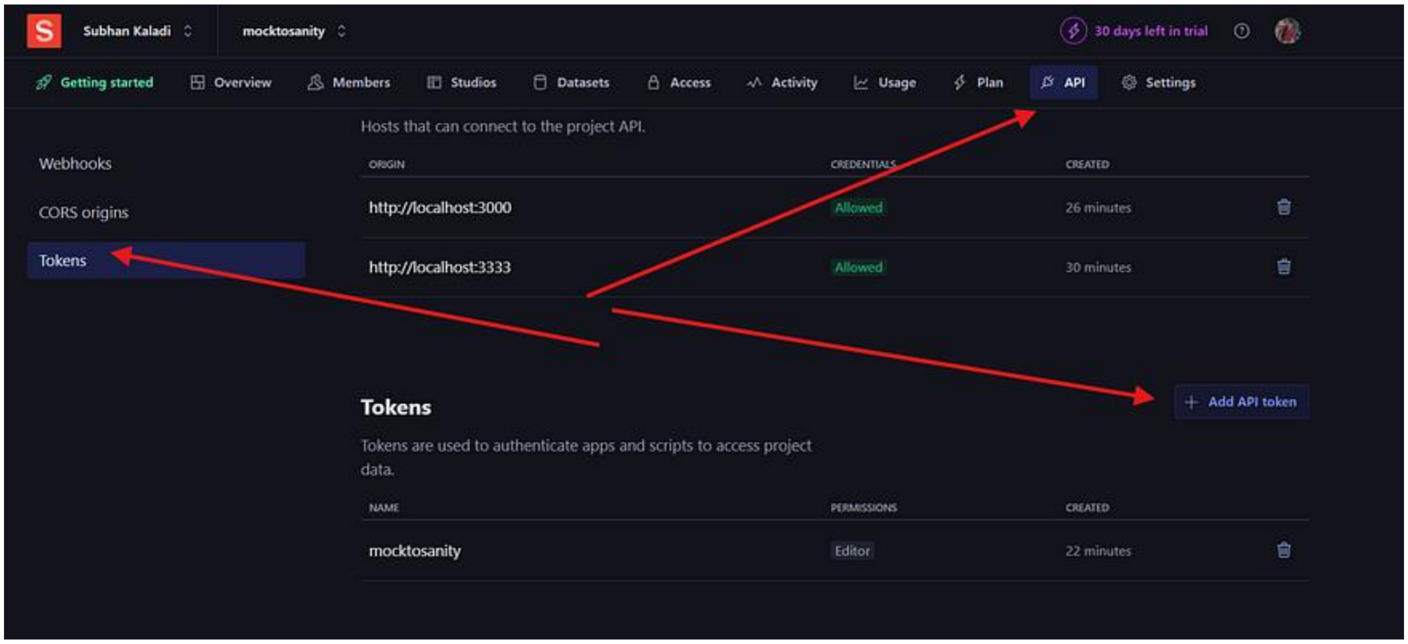


Use this ID for the `NEXT_PUBLIC_SANITY_PROJECT_ID` in your `.env.local` file.

API Token:

To generate a Sanity API token:

1. Go to https://www.sanity.io/manage and select your project
2. Navigate to the “API" tab
3. Under "Tokens," click "Add API token"
4. Give your token a name and select the appropriate permissions (usually "Editor" for full read/write access)
5. Copy the generated token



Hosts that can connect to the project API.

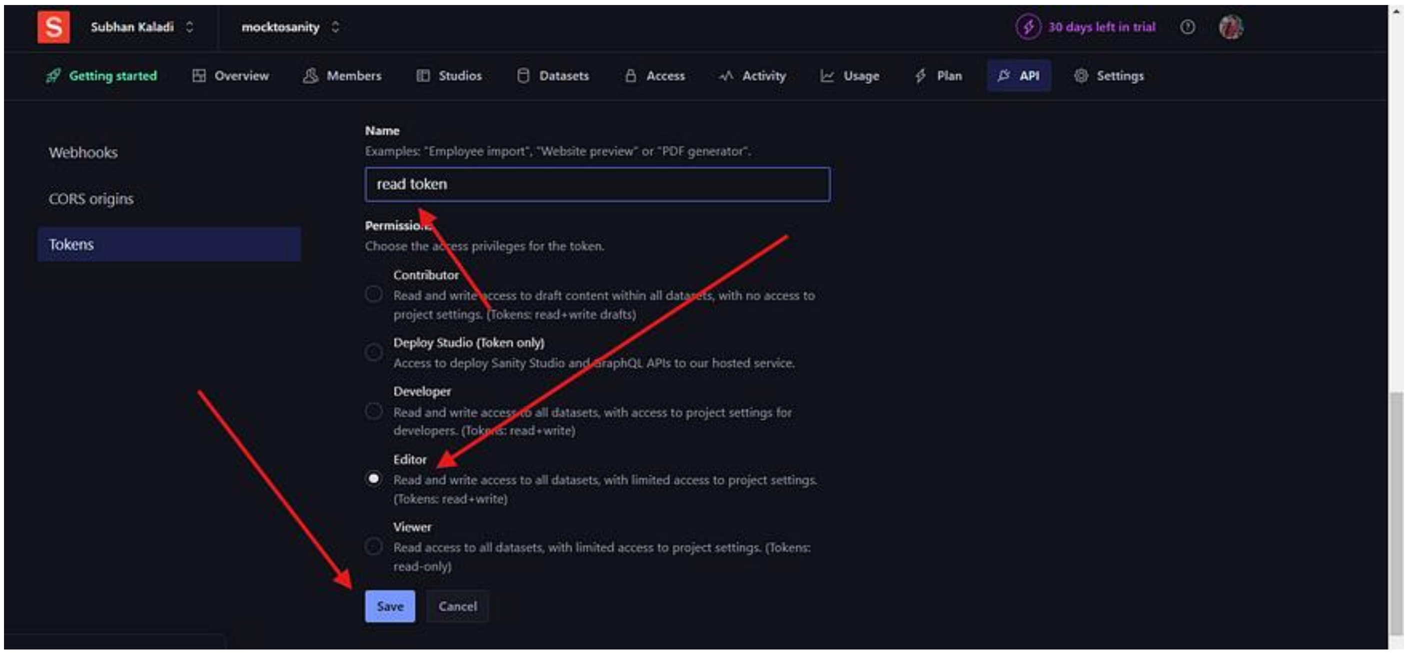
ORIGIN	CREDENTIALS	CREATED
http://localhost:3000	Allowed	26 minutes
http://localhost:3333	Allowed	30 minutes

Tokens

+ Add API token

Tokens are used to authenticate apps and scripts to access project data.

NAME	PERMISSIONS	CREATED
mocktosanity	Editor	22 minutes



Name

Examples: "Employee import", "Website preview" or "PDF generator".

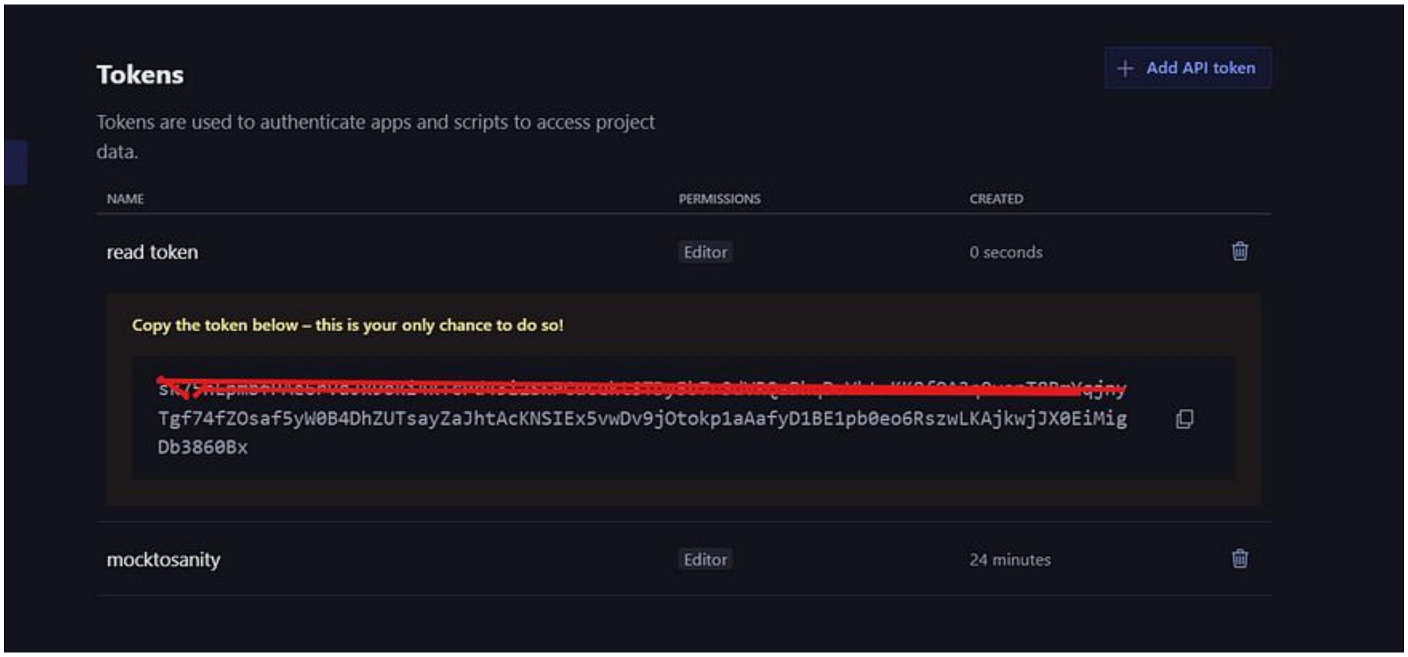
read token

Permissions

Choose the access privileges for the token.

- ○ **Contributor** — Read and write access to draft content within all datasets, with no access to project settings. (Tokens: read+write drafts)
- ○ **Deploy Studio (Token only)** — Access to deploy Sanity Studio and GraphQL APIs to our hosted service.
- ○ **Developer** — Read and write access to all datasets, with access to project settings for developers. (Tokens: read+write)
- ● **Editor** — Read and write access to all datasets, with limited access to project settings. (Tokens: read+write)
- ○ **Viewer** — Read access to all datasets, with limited access to project settings. (Tokens: read-only)

Save Cancel


Tokens

Tokens are used to authenticate apps and scripts to access project data.

NAME	PERMISSIONS	CREATED
read token	Editor	0 seconds
mocktosanity	Editor	24 minutes

Copy the token below – this is your only chance to do so!

```
[redacted]
Tgf74fZOsaf5yW0B4DhZUTsayZaJhtAcKNSIEx5vwDv9jOtokp1aAafyD1BE1pb0eo6RszwLKAjkwjJX0EiMig
Db3860Bx
```

Use this token for the `SANITY_API_TOKEN` in your `.env.local` file.

3. Creating the Sanity Schema

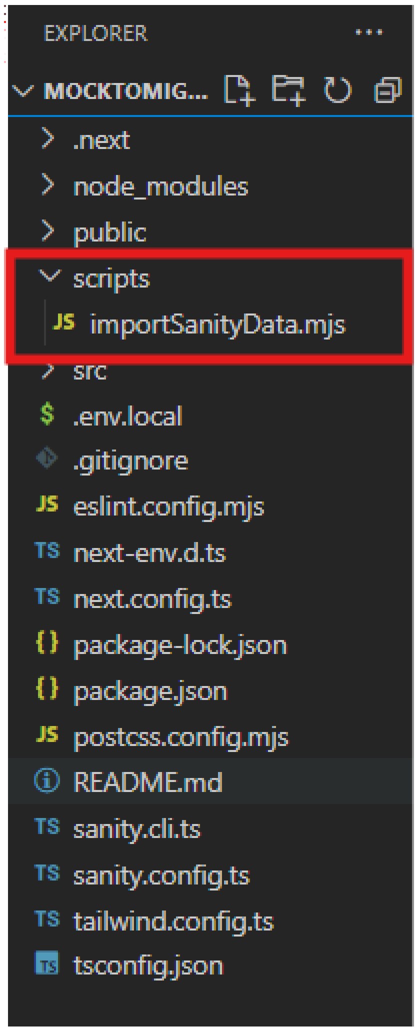
Now, let's create a schema for our products. In your Sanity schema folder (usually `sanity/schemaTypes`), create a new file called `product.ts`:

```
export default {
  name: 'product',
  type: 'document',
  title: 'Product',
  fields: [
    {
      name: 'name',
      type: 'string',
      title: 'Name',
      validation: (Rule: any) => Rule.required().error('Name is required'),
    },
{
  name: 'image',
  type: 'image',
  title: 'Image',
  options: {
    hotspot: true,
  },
  description: 'Upload an image of the product.',
},
{
  name: 'price',
  type: 'string',
  title: 'Price',
  validation: (Rule: any) => Rule.required().error('Price is required'),
},
{
  name: 'description',
  type: 'text',
  title: 'Description',
  validation: (Rule: any) =>
    Rule.max(150).warning('Keep the description under 150 characters.'),
},
{
  name: 'discountPercentage',
  type: 'number',
      title: 'Discount Percentage',
      validation: (Rule: any) =>
        Rule.min(0).max(100).warning('Discount must be between 0 and 100.'),
    },
    {
      name: 'isFeaturedProduct',
      type: 'boolean',
      title: 'Is Featured Product',
    },
    {
      name: 'stockLevel',
      type: 'number',
      title: 'Stock Level',
      validation: (Rule: any) => Rule.min(0).error('Stock level must be a positive
number.'),
    },
    {
      name: 'category',
      type: 'string',
      title: 'Category',
      options: {
        list: [
          { title: 'Chair', value: 'Chair' },
          { title: 'Sofa', value: 'Sofa' },
        ],
      },
      validation: (Rule: any) => Rule.required().error('Category is required'),
    },
  ],
};
```

Then, update your `sanity/schemaTypes/index.ts` file to include the new product schema:

```
import { type SchemaTypeDefinition } from 'sanity'
import product from './product'

export const schema: { types: SchemaTypeDefinition[] } = {
  types: [product],
}
```

Setting Up the Data Import Script



```
.next
node_modules
public
scripts
  importSanityData.mjs
src
.env.local
.gitignore
eslint.config.mjs
next-env.d.ts
next.config.ts
package-lock.json
package.json
postcss.config.mjs
README.md
sanity.cli.ts
sanity.config.ts
tailwind.config.ts
tsconfig.json
```

Now, let's create a script to import data from an external API into Sanity. Create a new file `scripts/import-data.mjs` in your project root:

import { createClient } from '@sanity/client';

import axios from 'axios';

import dotenv from 'dotenv';

import { fileURLToPath } from 'url';

import path from 'path';

const __filename = fileURLToPath(import.meta.url);

const __dirname = path.dirname(__filename);

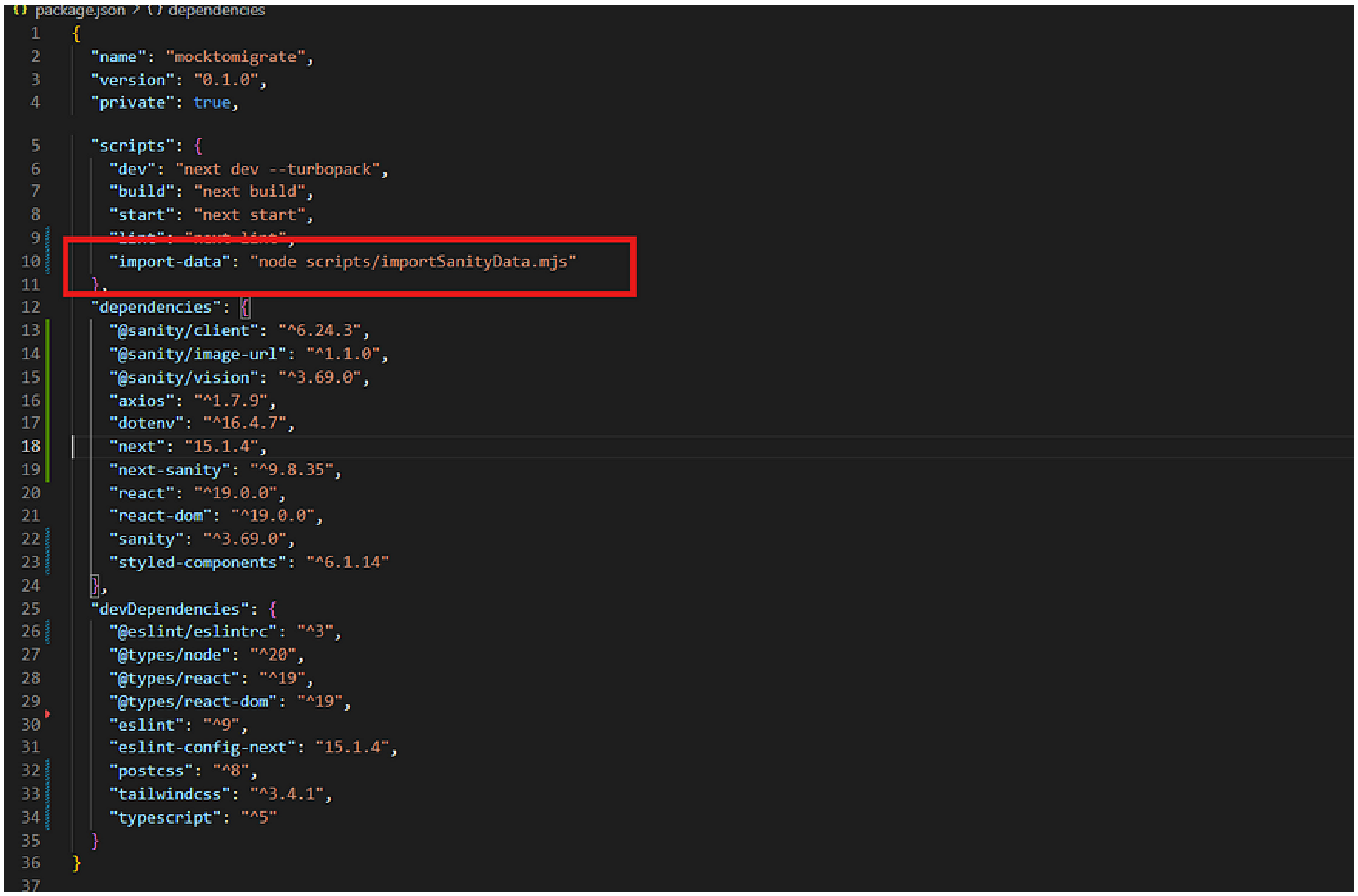
dotenv.config({ path: path.resolve(__dirname, '../../.env') });

```
const client = createClient({
  projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
  dataset: process.env.NEXT_PUBLIC_SANITY_DATASET,
  token: process.env.SANITY_API_TOKEN,
  apiVersion: '2025-01-15',
  useCdn: false,
});
async function uploadImageToSanity(imageUrl) {
  try {
    console.log(`Uploading Image : ${imageUrl}`);
    const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
    const buffer = Buffer.from(response.data);
    const asset = await client.assets.upload('image', buffer, {
      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 importData() {
  try {
    console.log('Fetching Product Data From API ...');
    const response = await axios.get("https://next-ecommerce-template-
4.vercel.app/api/product")
    const products = response.data.products;
    for (const item of products) {
      console.log(`Processing Item: ${item.name}`);
      let imageRef = null;
      if (item.imagePath) {
        imageRef = await uploadImageToSanity(item.imagePath);
      }
      const sanityItem = {
        _type: 'product',
        name: item.name,
        category: item.category || null,
        price: item.price,
        description: item.description || '',
        discountPercentage: item.discountPercentage || 0,
        stockLevel: item.stockLevel || 0,
        isFeaturedProduct: item.isFeaturedProduct,
        image: imageRef
          ? {
              _type: 'image',
              asset: {
                _type: 'reference',
                _ref: imageRef,
              },
          }
        : undefined,
      };
      console.log(`Uploading ${sanityItem.category} - ${sanityItem.name} to Sanity !`);
      const result = await client.create(sanityItem);
      console.log(`Uploaded Successfully: ${result._id}`);
      console.log("----------------------------------------------------------")
      console.log("\n\n")
    }
    console.log('Data Import Completed Successfully !');
  } catch (error) {
    console.error('Error Importing Data : ', error);
  }
}
importData();
```

Now, let's install the necessary packages. Run the following command in your terminal:

```
npm install @sanity/client axios dotenv
```

Running the Import Script



```
{
  "name": "mocktomigrate",
  "version": "0.1.0",
  "private": true,

  "scripts": {
    "dev": "next dev --turbopack",
    "build": "next build",
    "start": "next start",
    "lint": "next lint",
    "import-data": "node scripts/importSanityData.mjs"
  },
  "dependencies": {
    "@sanity/client": "^6.24.3",
    "@sanity/image-url": "^1.1.0",
    "@sanity/vision": "^3.69.0",
    "axios": "^1.7.9",
    "dotenv": "^16.4.7",
    "next": "15.1.4",
    "next-sanity": "^9.8.35",
    "react": "^19.0.0",
    "react-dom": "^19.0.0",
    "sanity": "^3.69.0",
    "styled-components": "^6.1.14"
  },
  "devDependencies": {
    "@eslint/eslintrc": "^3",
    "@types/node": "^20",
    "@types/react": "^19",
    "@types/react-dom": "^19",
    "eslint": "^9",
    "eslint-config-next": "15.1.4",
    "postcss": "^8",
    "tailwindcss": "^3.4.1",
    "typescript": "^5"
  }
}
```

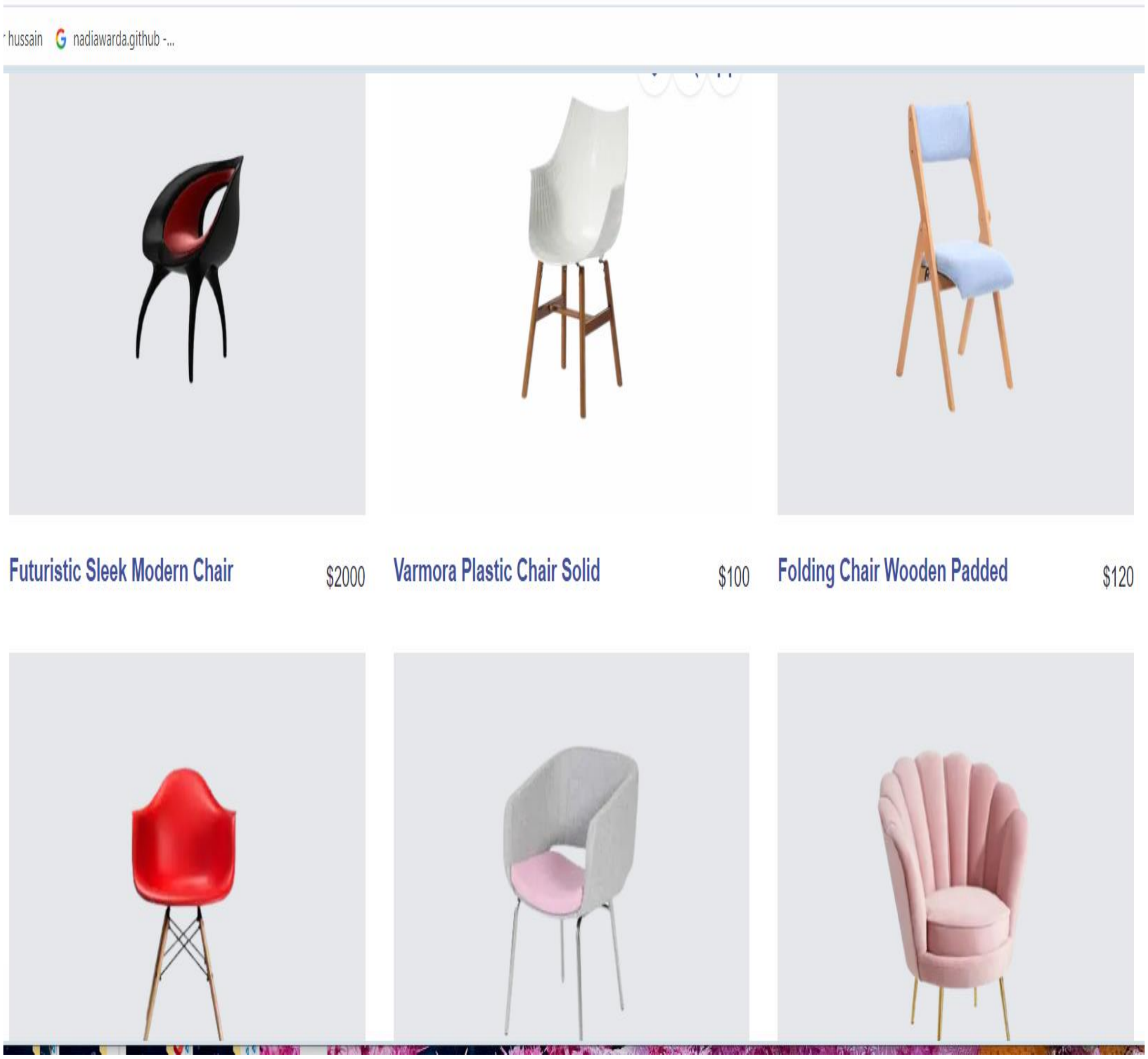
To run the import script, we need to add a new script to our `package.json` file. Open your `package.json` and add the following to the `"scripts"` section:

```
"scripts": {
  "dev": "next dev --turbopack",
  "build": "next build",
  "start": "next start",
  "lint": "next lint",
  "import-data": "node scripts/import-data.mjs"
}
```

Now you can run the import script using:

npm run import-data

This script will fetch products from the FakeStoreAPI, upload any associated images to Sanity's asset store, and then create new product documents in your Sanity dataset.

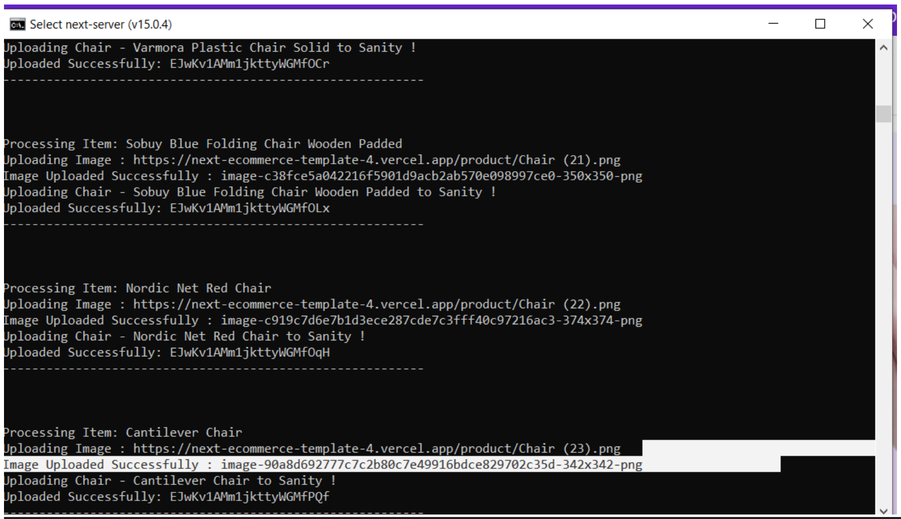


hussain nadiawarda.github -...

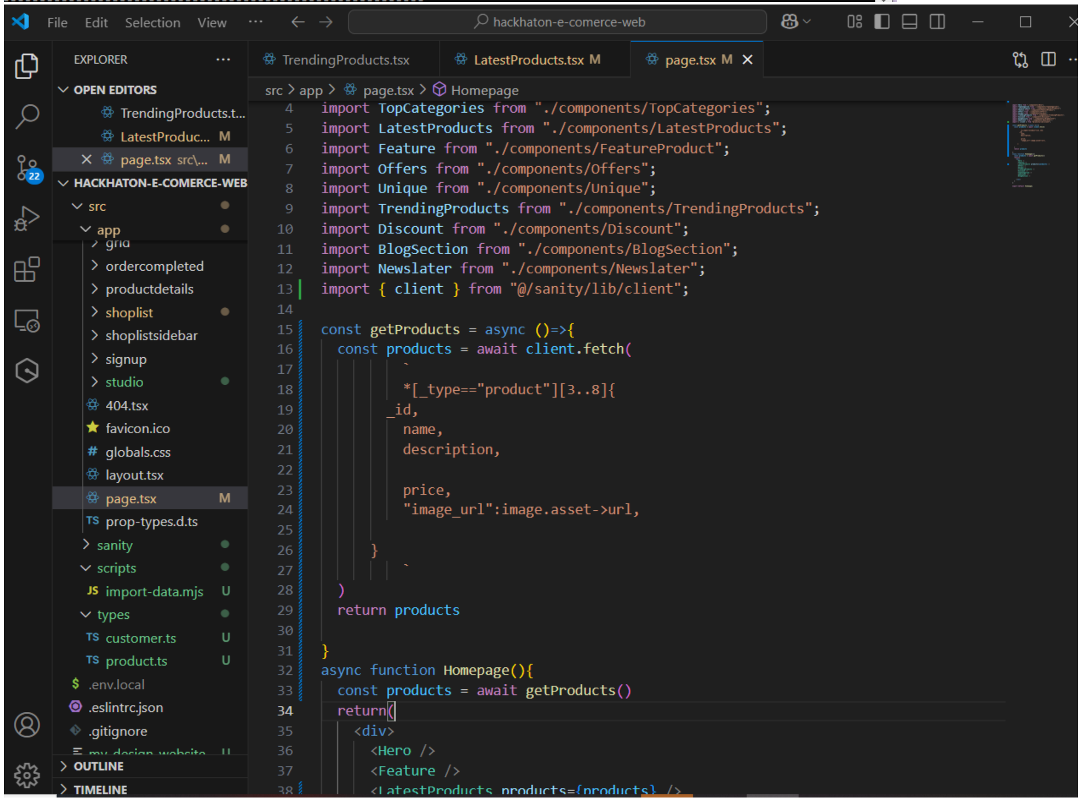
Futuristic Sleek Modern Chair $2000

Varmora Plastic Chair Solid $100

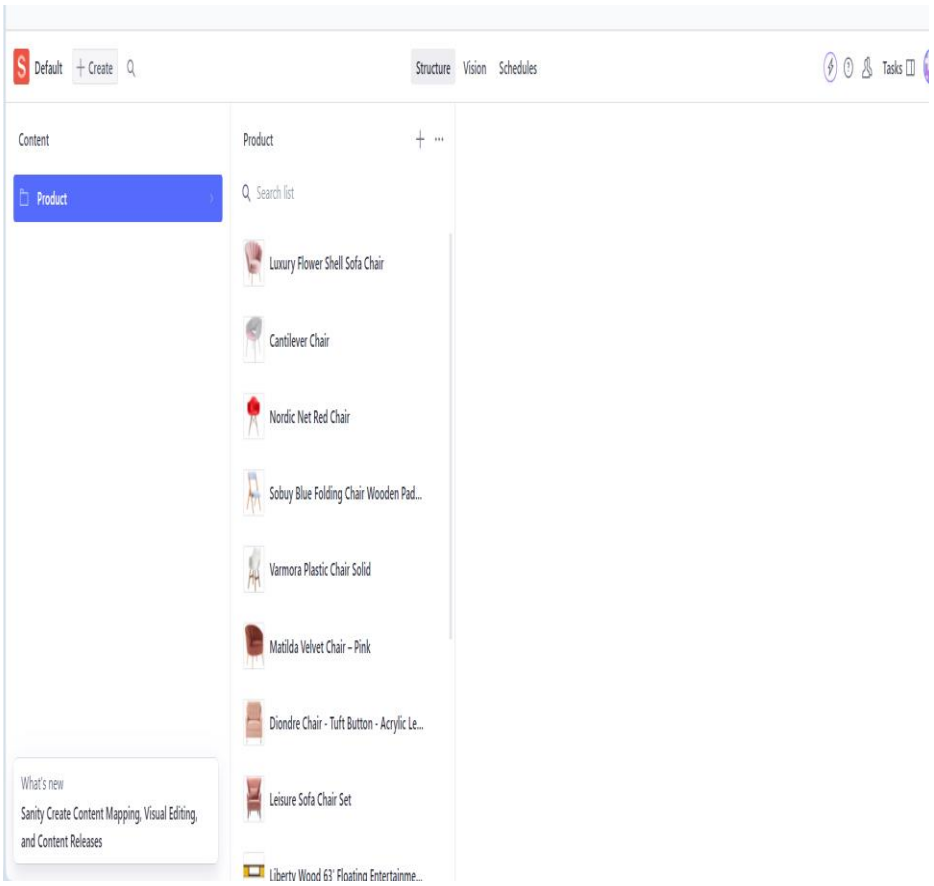
Folding Chair Wooden Padded $120



```
Uploading Chair - Varmora Plastic Chair Solid to Sanity !
Uploaded Successfully: EJwKv1AMm1jkttyWGMfOCr
----------------------------------------------------------



Processing Item: Sobuy Blue Folding Chair Wooden Padded
Uploading Image : https://next-ecommerce-template-4.vercel.app/product/Chair (21).png
Image Uploaded Successfully : image-c38fce5a042216f5901d9acb2ab570e098997ce0-350x350-png
Uploading Chair - Sobuy Blue Folding Chair Wooden Padded to Sanity !
Uploaded Successfully: EJwKv1AMm1jkttyWGMfOLx
----------------------------------------------------------



Processing Item: Nordic Net Red Chair
Uploading Image : https://next-ecommerce-template-4.vercel.app/product/Chair (22).png
Image Uploaded Successfully : image-c919c7d6e7b1d3ece287cde7c3fff40c97216ac3-374x374-png
Uploading Chair - Nordic Net Red Chair to Sanity !
Uploaded Successfully: EJwKv1AMm1jkttyWGMfOqH
----------------------------------------------------------



Processing Item: Cantilever Chair
Uploading Image : https://next-ecommerce-template-4.vercel.app/product/Chair (23).png
Image Uploaded Successfully : image-90a8d692777c7c2b80c7e49916bdce829702c35d-342x342-png
Uploading Chair - Cantilever Chair to Sanity !
Uploaded Successfully: EJwKv1AMm1jkttyWGMfPQf
```



```tsx
import TopCategories from "./components/TopCategories";
import LatestProducts from "./components/LatestProducts";
import Feature from "./components/FeatureProduct";
import Offers from "./components/Offers";
import Unique from "./components/Unique";
import TrendingProducts from "./components/TrendingProducts";
import Discount from "./components/Discount";
import BlogSection from "./components/BlogSection";
import Newslater from "./components/Newslater";
import { client } from "@/sanity/lib/client";

const getProducts = async ()=>{
  const products = await client.fetch(
          `
          *[_type=="product"][3..8]{
         _id,
          name,
          description,

          price,
          "image_url":image.asset->url,

        }
          `
  )
  return products

}
async function Homepage(){
  const products = await getProducts()
  return(
    <div>
      <Hero />
      <Feature />
      <LatestProducts products={products} />
```



Luxury Flower Shell Sofa Chair

Cantilever Chair

Nordic Net Red Chair

Sobuy Blue Folding Chair Wooden Pad...

Varmora Plastic Chair Solid

Matilda Velvet Chair – Pink

Diondre Chair - Tuft Button - Acrylic Le...

Leisure Sofa Chair Set

Liberty Wood 63' Floating Entertainme...

What's new

Sanity Create Content Mapping, Visual Editing, and Content Releases