

Day 3 (overview)

Easyway is a versatile API integration framework designed to help you easily fetch data from external APIs, display it on the frontend, and manage it with a CMS like Sanity. This documentation will guide you through setting up and using Easyway in your Next.js project to achieve seamless data integration.

1. Installation

To get started with Easyway, follow these installation steps:

TERMINAL=>cmd =>

```
# Install necessary dependencies sanity
```

```
npm install @sanity/client
```

Manually Added data in schema Cms:

I manually added data in sanity CMS . To Do these I uses Sanity Studio and Populated Various Fields (such as image , title,description,price)according to the schema . After enter the relevant data in each fields. I saved it to ensured it was properly store in the CMS and would display correctly on the frontend.

!)Schema Adjustments Made:I made sanity schema to accommodate the data . these adjustments included adding fields such as list of fields e.g(image,title,description,price etc)

2)Fields Details:The data I added included (e.g:product,title,price,description,image etc)

Table of Contents

1. Setting Up Environment Variables

2. Obtaining Sanity Project ID and API Token

3. Creating the Sanity Schema

4. Setting Up the Data Import Script

5. Running the Import Script

1. Configuring Environment Variables:

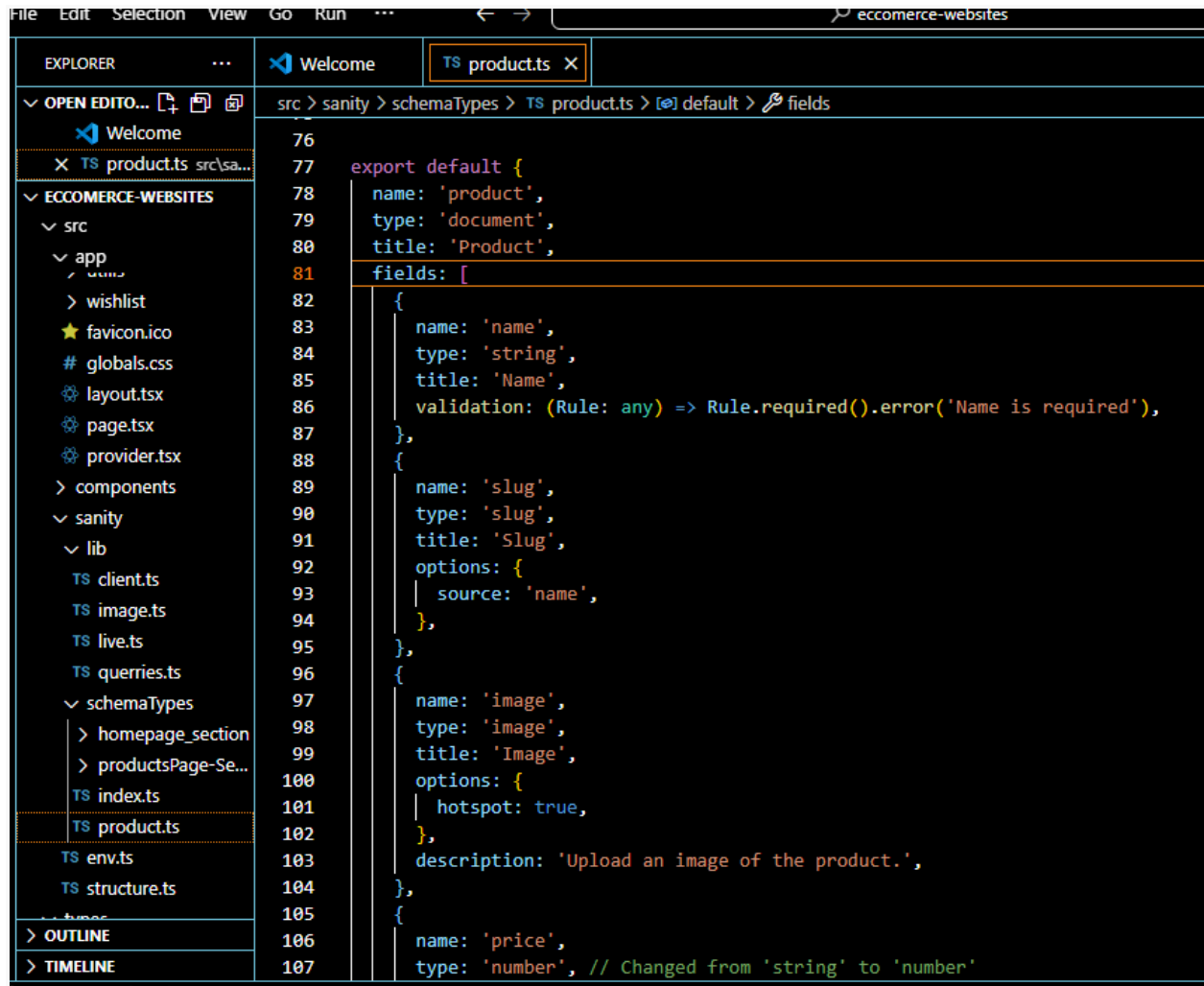
Begin by setting up your environment variables. If a `.env.local` file doesn't already exist in your project's root directory, create one. Then, add the following variables:

```
NEXT_PUBLIC_SANITY_PROJECT_ID="SANITY ID"  
NEXT_PUBLIC_SANITY_DATASET="production"  
SANITY_API_TOKEN="SANITY_TOKEN"
```

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`:

Schema=>product.ts

A screenshot of the Visual Studio Code editor interface. The Explorer sidebar on the left shows a project structure with folders 'src', 'app', 'components', 'sanity', and 'lib'. The 'sanity' folder is expanded, showing 'schemaTypes' with sub-items 'homepage_section', 'productsPage-Se...', 'index.ts', and 'product.ts'. The 'product.ts' file is selected and open in the editor. The editor shows a TypeScript schema definition for a product. The code includes an 'export default' block with fields for 'name', 'slug', 'image', and 'price'. The 'name' field has a validation rule. The 'slug' field has an option to source from 'name'. The 'image' field has a hotspot and a description. The 'price' field has a comment indicating it was changed from 'string' to 'number'.

```
76
77 export default {
78   name: 'product',
79   type: 'document',
80   title: 'Product',
81   fields: [
82     {
83       name: 'name',
84       type: 'string',
85       title: 'Name',
86       validation: (Rule: any) => Rule.required().error('Name is required'),
87     },
88     {
89       name: 'slug',
90       type: 'slug',
91       title: 'Slug',
92       options: {
93         source: 'name',
94       },
95     },
96     {
97       name: 'image',
98       type: 'image',
99       title: 'Image',
100      options: {
101        hotspot: true,
102      },
103      description: 'Upload an image of the product.',
104    },
105     {
106       name: 'price',
107       type: 'number', // Changed from 'string' to 'number'
```

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

```
import { type SchemaTypeDefinition } from 'sanity'
```

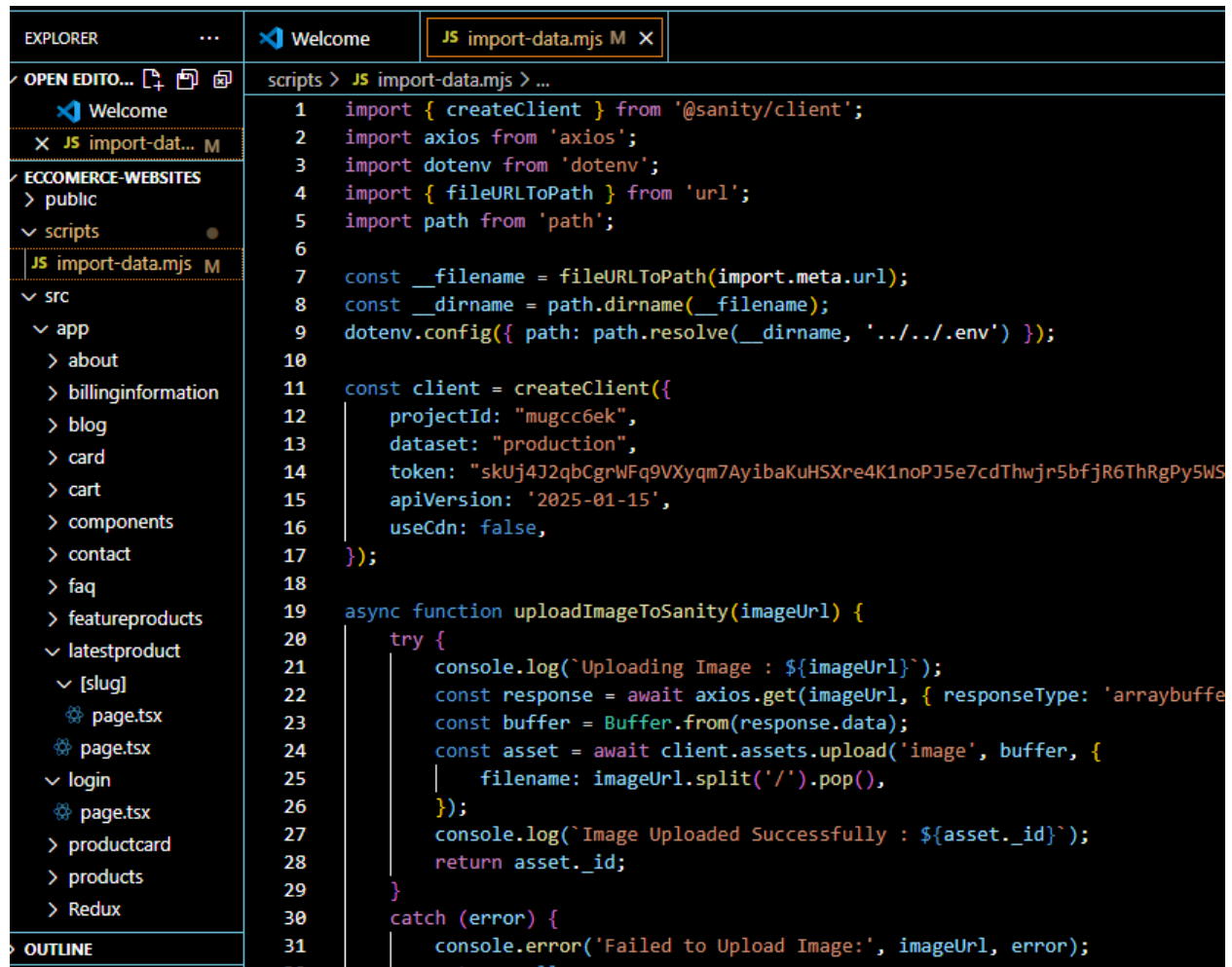
```
import product from './product'
```

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

```
  types: [product], }
```

4. Setting Up the Data Import Script

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



```
1 import { createClient } from '@sanity/client';
2 import axios from 'axios';
3 import dotenv from 'dotenv';
4 import { fileURLToPath } from 'url';
5 import path from 'path';
6
7 const __filename = fileURLToPath(import.meta.url);
8 const __dirname = path.dirname(__filename);
9 dotenv.config({ path: path.resolve(__dirname, '../.env') });
10
11 const client = createClient({
12   projectId: "mugcc6ek",
13   dataset: "production",
14   token: "skUj4J2qbCgrWFq9VXym7AyibaKuHSXre4K1noPJ5e7cdThwjr5bfjR6ThRgPy5WS",
15   apiVersion: '2025-01-15',
16   useCdn: false,
17 });
18
19 async function uploadImageToSanity(imageUrl) {
20   try {
21     console.log(`Uploading Image : ${imageUrl}`);
22     const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
23     const buffer = Buffer.from(response.data);
24     const asset = await client.assets.upload('image', buffer, {
25       filename: imageUrl.split('/').pop(),
26     });
27     console.log(`Image Uploaded Successfully : ${asset._id}`);
28     return asset._id;
29   } catch (error) {
30     console.error('Failed to Upload Image:', imageUrl, error);
31   }
32 }
```

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

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

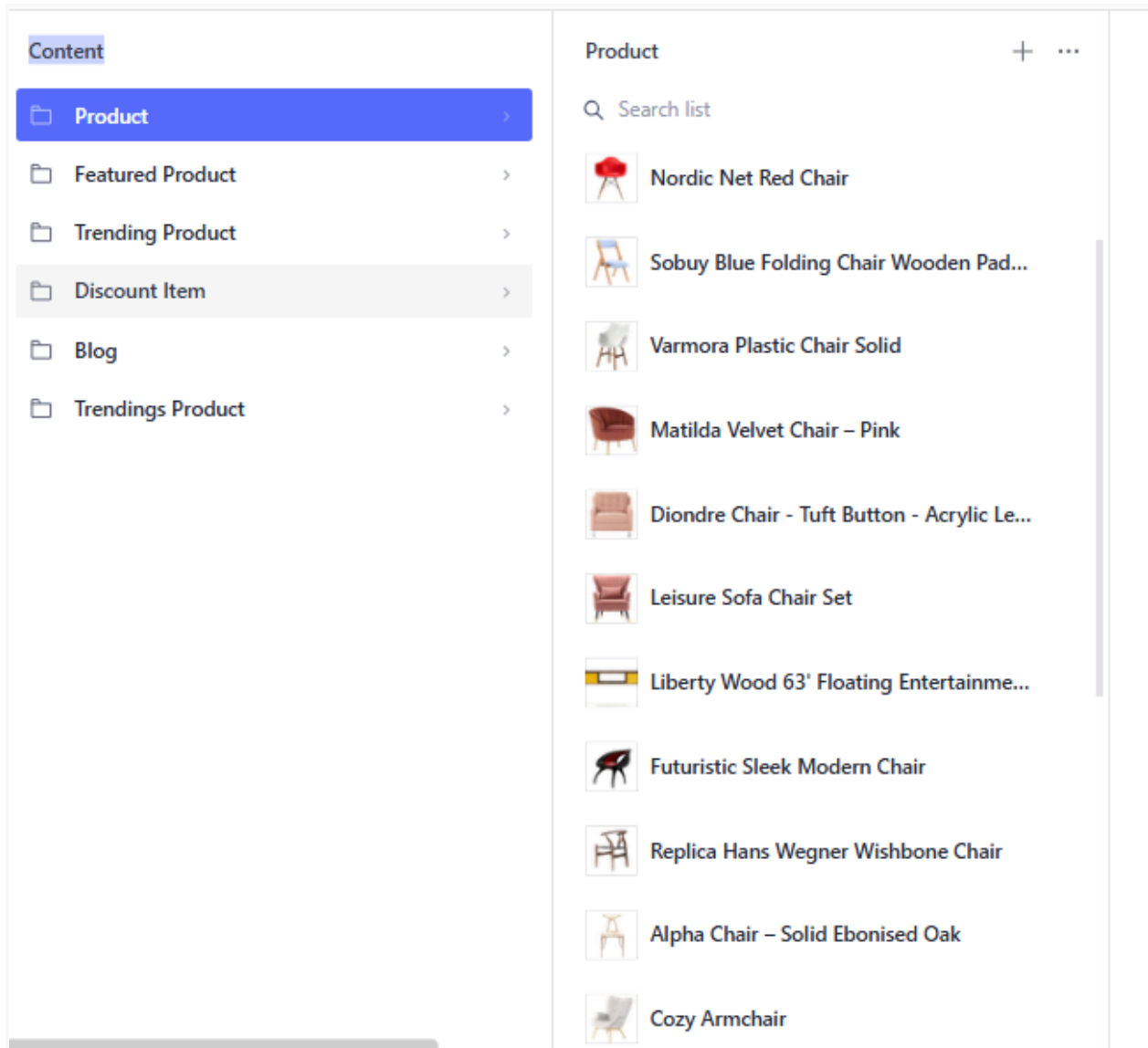
5. Running the Import Script

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 --turbo",  
  
  "build": "next build",  
  
  "start": "next start",  
  
  "lint": "next lint",  
  
  "import-data": "node scripts/importSanityData.mjs"  
  
}
```

Now you can run the import script using:

`npm run import-dataApi` Data fetch in Sanity



`Manually Data fetch in Sanity`

S

+ Create

Q

StructureVisionSchedules

Content

Product

Featured Product

Trending Product


Discount Item

Blog

Trendings Product

Featured Product

Q Search list


 Cantilever Chair

Cantilever Chair

Featured Product

Cantilever Chair

Feature Product Image



Heading Text

Cantilever Chair

Content

Product

Featured Product

Trending Product


Discount Item

Blog

Trendings Product

Trending Product

Q Search list

 Unique Features Of Latest &