Day-3 : API Integration & Data Migration

Objective:

Migrate API data into Sanity CMS and integrate it into a Next.js project for a furniture e-commerce website.

Following are some steps to complete Day-3 Task.

Key Steps:

1. Fetching data from API.
2. Migrate this data to Sanity CMS.
3. Render this data on the E Commerce Website.

Documentation:

API Integration:

API (Application Programming Interface) integration is the process of connecting different software systems, applications, or services through APIs.

How the API Integration Process Works:

1. **Understand Requirements:**
    - Identify the purpose: Determine the systems or services to connect and the specific functionalities needed.
    - Define the use case: For example, pulling data from a third-party app, sending data, or enabling two-way communication.
2. **Select the API:**
    - Choose the API(s) to integrate. APIs can be public (offered by third parties like Google, Twitter, or Stripe) or private (built for internal use).
3. **Authentication and Authorization:**
    - Most APIs require security measures like API keys, OAuth tokens, or JWT (JSON Web Tokens) to ensure secure communication.
    - Obtain credentials from the API provider and configure them in your system.
4. **Read API Documentation:**
    - Review the API documentation provided by the API provider. This includes:

        - Available endpoints (e.g., /users, /products).
        - HTTP methods (e.g., GET, POST, PUT, DELETE).
        - Request and response formats (e.g., JSON, XML).
        - Error codes and troubleshooting tips.

5. **Develop the Integration:**
    - Build the connection:
        - Use programming languages or frameworks (e.g., Next.js, TypeScript, React) to create API requests.
        - Send HTTP requests (GET for retrieving data, POST for sending data, etc.).
    - Parse the response:
        - Process the response data (e.g., JSON) and use it in your application.
    - Handle errors gracefully to manage failed requests or invalid inputs.
6. **Test the Integration:**
    - Test endpoints with tools like Postman or cURL.
    - Verify that requests and responses are functioning as expected.
    - Check for edge cases and error handling.
7. **Deploy and Monitor:**
    - Deploy the integration to a live environment.
    - Monitor for issues, such as API downtime, rate limits, or changes to the API by the provider.
8. **Maintenance:**
    - Regularly update the integration to accommodate changes in API versions or business requirements.
    - Ensure compliance with new security policies or protocols.

Adjustment made of Schemas:

Adjustments to schemas refer to modifications or updates made to data structures (schemas) that define how data is organized, stored, and processed. A schema acts as a blueprint, dictating the structure of data (e.g., tables, fields, relationships, and constraints) in databases, APIs, or data models.

How Schema Adjustments Work:

1. **Identify the Need for Adjustment:**
    - Analyze why the schema needs changes. Common reasons include:
        - Adding new features or functionalities.
        - Addressing performance bottlenecks.
        - Correcting inconsistencies or errors.
        - Accommodating new data types or structures.
2. **Plan the Changes:**
    - Define what changes need to be made. Examples include:
        - Adding or removing columns in a database table.
        - Renaming fields or updating data types.
        - Modifying relationships between tables (e.g., adding foreign keys).
        - Updating API request/response schemas (e.g., adding new fields or endpoints).
3. **Evaluate Impact:**

    Assess the potential impact of schema changes on:
    - Existing data.
    - Applications or systems relying on the schema.
    - Users or customers interacting with the data.

    Plan for backward compatibility or migration strategies if necessary.
4. **Test the Adjustments:**

Verify the changes in a staging or development environment to ensure:

- Data integrity is maintained.
- Applications consuming the schema function correctly.
- There are no performance regressions or unexpected issues.
-

5. **Migrate or Transform Data (if needed):**

- If schema changes affect existing data, migrate or transform the data to align with the updated structure.
- Use tools or scripts for seamless migration.

6. **Deploy and Monitor:**

- Deploy schema adjustments to production.
- Monitor for errors or unintended side effects.
- Ensure continuous system functionality.

Migration Steps and Tools Used:

Migrating or integrating APIs involves transferring or adapting data, functionality, or services from one API to another or updating an existing API version.

Steps for API Integration and Migration

1. Plan the Migration

- Define the Scope: Understand what needs to be migrated (e.g., endpoints, data models, authentication mechanisms).
- Identify Dependencies: List all applications, services, or users consuming the API.
- Assess Compatibility:
  - Review differences between the old and new API (e.g., endpoint changes, schema updates, authentication updates).
  - Ensure backward compatibility or plan for adjustments.
- Set Objectives: Define goals, such as improved performance, scalability, or enhanced features.

2. Analyze the New API

- Review the new API's documentation:
    - Endpoints (products →tems).
    - Authentication protocols (e.g., API keys → OAuth 2.0).
    - Data format changes (e.g., XML → JSON).
    - Rate limits or quotas.
- Map old API features to new ones, identifying any gaps or changes.

3. Build a Migration Strategy

- Incremental Migration:
    - Migrate a portion of the API while keeping the old API functional for existing users.
    - Gradually shift consumers to the new API.
- Big Bang Migration:
    - Replace the old API with the new one in a single rollout.
    - Requires extensive testing and preparation.
- Versioning:
    - Maintain multiple API versions (e.g., /v1, /v2) to ensure smooth transitions.

4. Develop and Test the Migration

- Integration Development:
    - Update or create code to interact with the new API.
    - Implement authentication and authorization mechanisms for the new API.
    - Adapt to schema changes in requests and responses.
- Testing:
    - Use tools like Postman or cURL to test individual endpoints.
    - Test edge cases, error handling, and performance.
    - Validate data mapping between the old and new APIs.

4. **Data Migration (if applicable)**

    - If data structures or storage methods change, migrate existing data to align with the new API schema:
    - Use scripts or ETL tools to transfer data.
    - Test for data integrity and consistency.

6. Deploy the Migration

- Deploy the new API or updated integration in a staging environment first.
- Monitor for issues and perform final testing.
- Roll out changes to production, ideally during low-traffic hours.

7. Monitor and Optimize

- Continuously monitor API usage and performance.
- Address bugs, errors, or bottlenecks.
- Deprecate the old API once all users have migrated successfully.

Tools for API Integration and Migration

1. API Development and Testing Tools

- Postman: For testing and documenting API requests and responses.
- cURL: Command-line tool for testing API calls.
- Swagger/OpenAPI: For documenting and generating API client libraries.
- Insomnia: A lightweight API testing and debugging tool.

2. Data Migration Tools

- ETL Tools:
  - Talend: Extract, transform, and load data between systems.
  - Apache NiFi: For automating data flow between APIs or databases.
- Custom Scripts:
  - Python (e.g., using libraries like requests, pandas).
  - Node.js (e.g., using axios or fetch).

3. Version Control and CI/CD Tools

- Git: Version control for managing migration scripts and code changes.
- Jenkins/GitHub Actions/Travis CI: Automate the testing and deployment of API changes.

4. Monitoring and Logging Tools

- New Relic: Monitor API performance and uptime.
- Datadog: Comprehensive monitoring for APIs and backend systems.
- Elastic Stack (ELK): For centralized logging and error analysis.

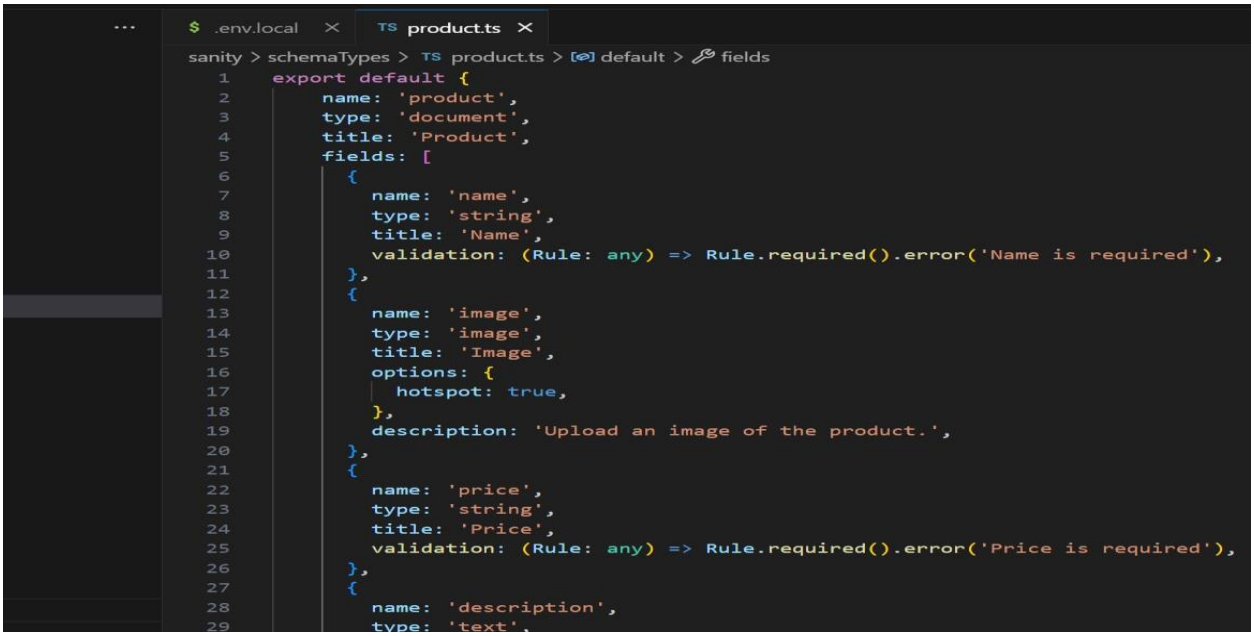
Screenshots

Started Project by Sanity Configuration in E Commerce Website:

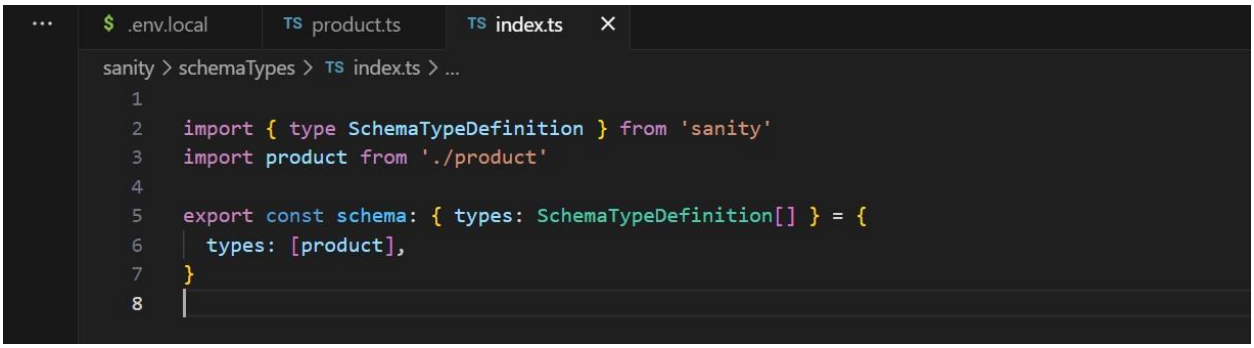
```
added 901 packages, changed 1 package, and audited 1319 packages in 9m

242 packages are looking for funding
  run `npm fund` for details

1 moderate severity vulnerability

To address all issues, run:
  npm audit fix --force

Run `npm audit` for details.

added 8 packages, and audited 1327 packages in 31s

242 packages are looking for funding
  run `npm fund` for details

1 moderate severity vulnerability

To address all issues, run:
  npm audit fix --force

Run `npm audit` for details.

Success! Your Sanity configuration files has been added to this project
```

Configuration Environment Variable:

Snippet of updated Product Schema:

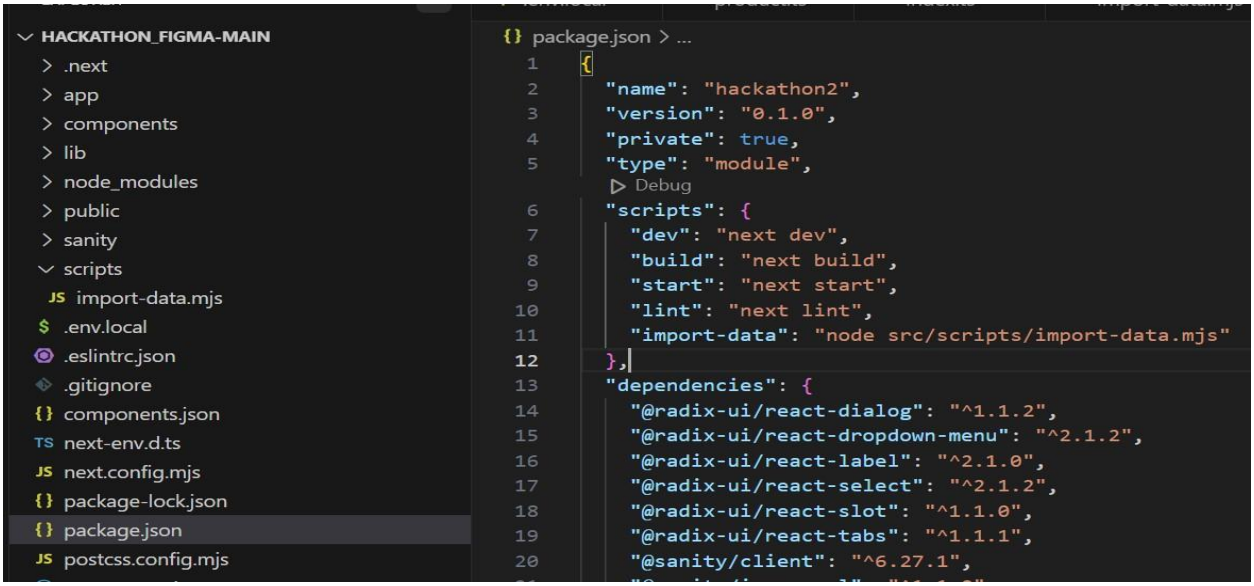


```
sanity > schemaTypes > product.ts > default > fields
export default {
    name: 'product',
    type: 'document',
    title: 'Product',
    fields: [
      {
        name: 'name',
        type: 'string',
        title: 'Name',
        validation: (Rule: any) => Rule.required().error('Name is required'),
      },
      {
        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',
```

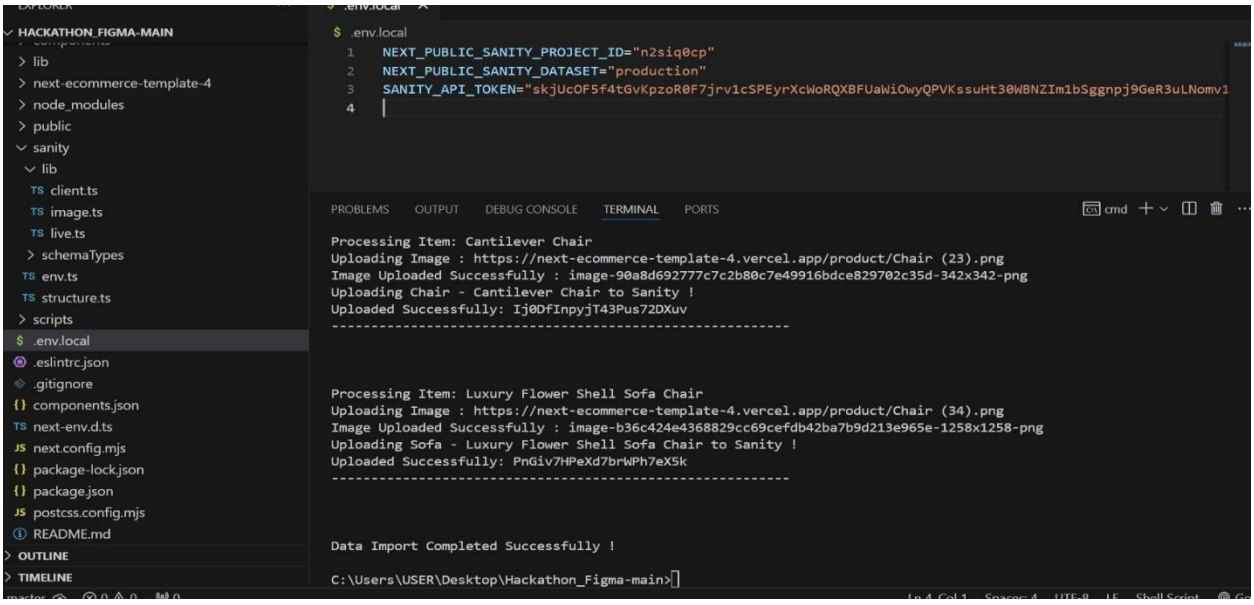


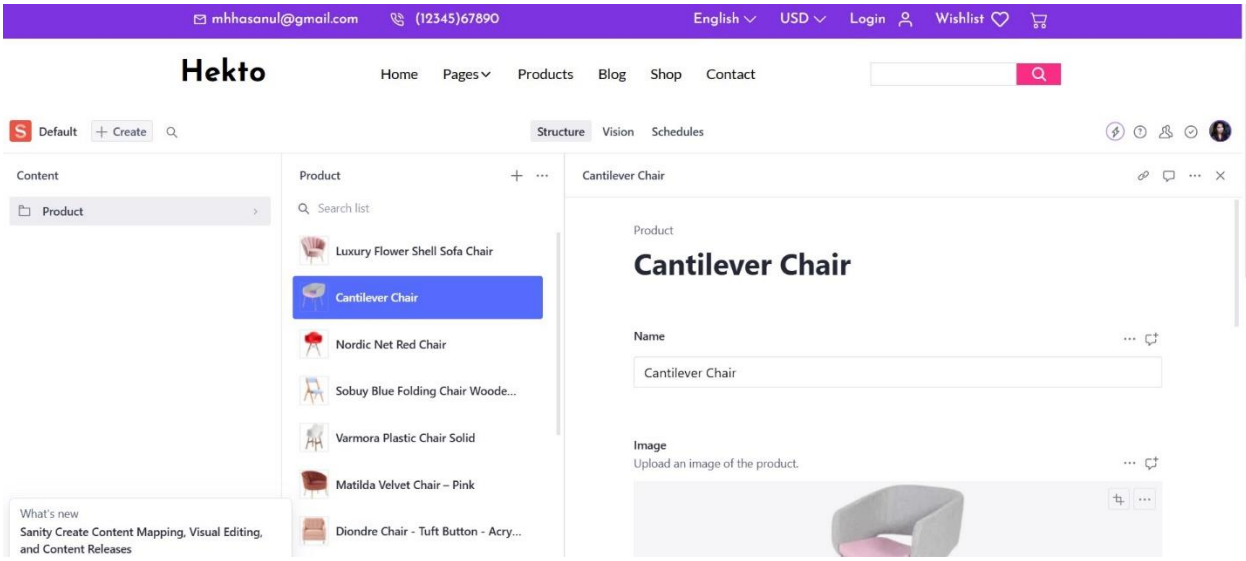
```
sanity > schemaTypes > index.ts > ...

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

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

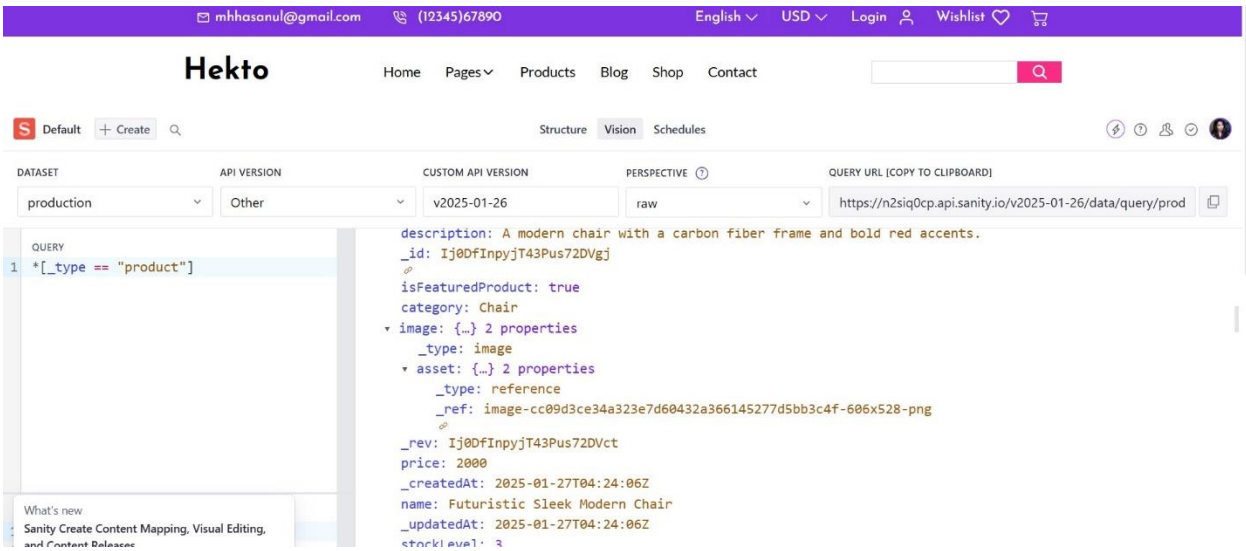
Setting Up the Data Import Script:



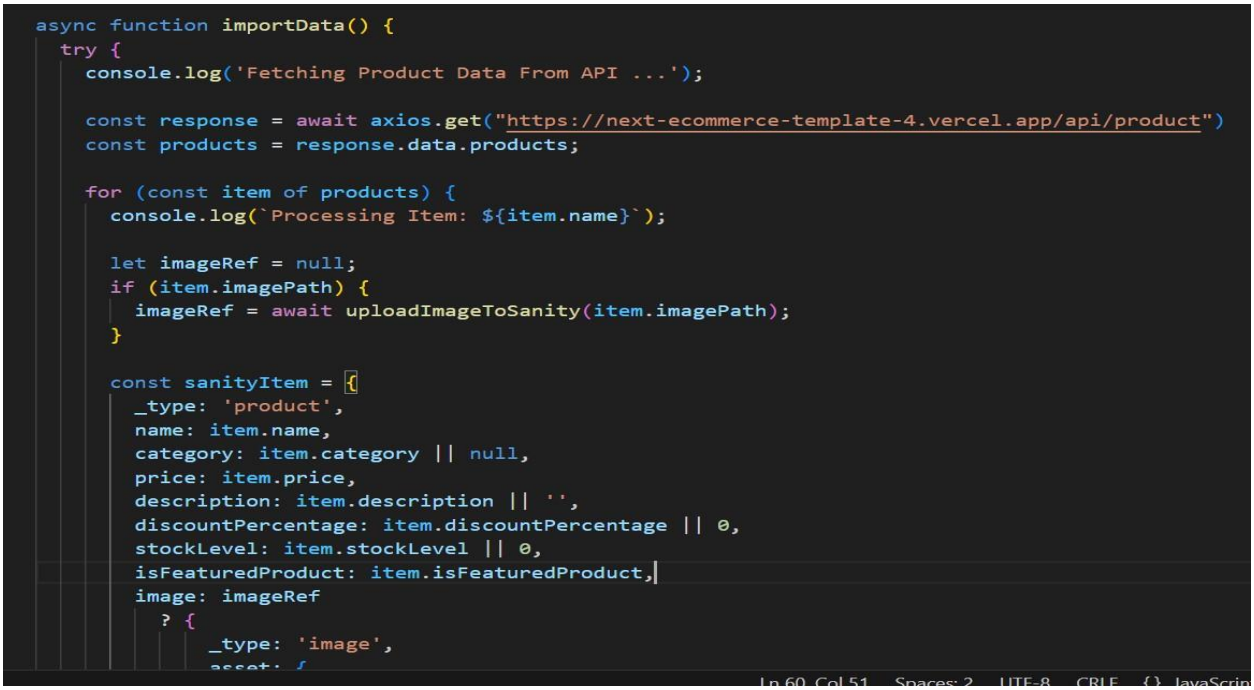
Import Data:



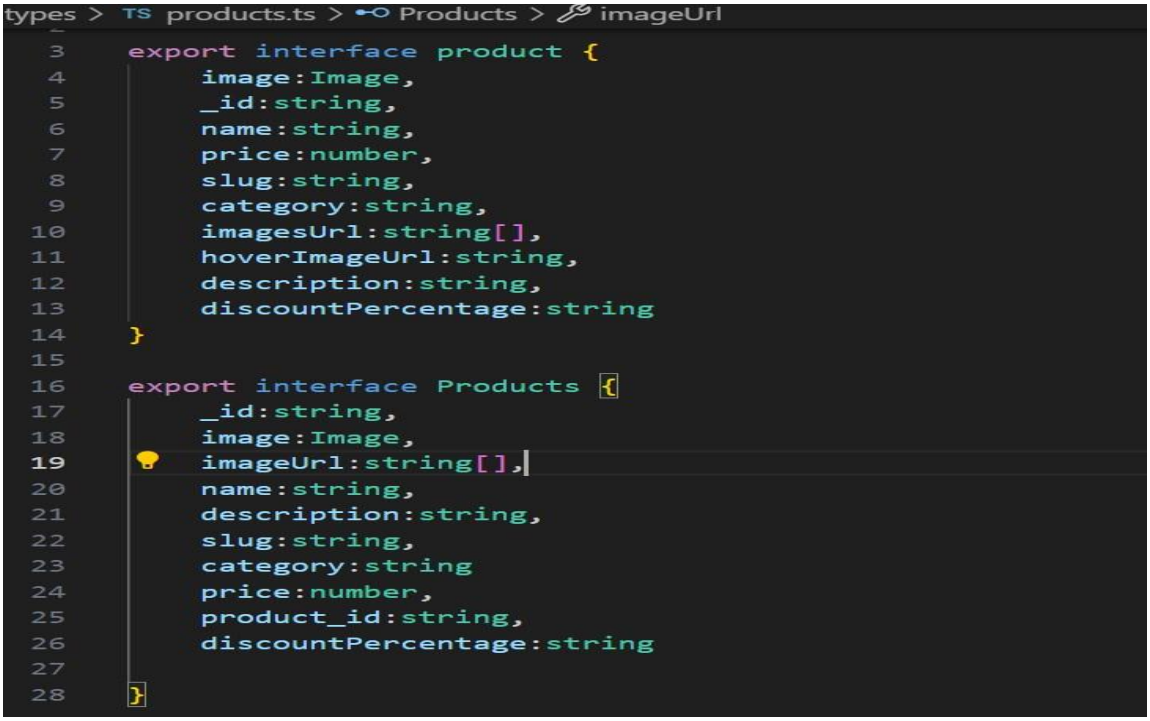

mhhasanul@gmail.com
(12345)67890
English
USD
Login
Wishlist
Hekto
Home
Pages
Products
Blog
Shop
Contact
Default
Create
Structure
Vision
Schedules
Content
Product
Search list
Luxury Flower Shell Sofa Chair
Cantilever Chair
Nordic Net Red Chair
Sobuy Blue Folding Chair Woode...
Varmora Plastic Chair Solid
Matilda Velvet Chair – Pink
Diondre Chair - Tuft Button - Acry...
What's new
Sanity Create Content Mapping, Visual Editing, and Content Releases
Cantilever Chair
Product
Cantilever Chair
Name
Cantilever Chair
Image
Upload an image of the product.


mhhasanul@gmail.com
(12345)67890
English
USD
Login
Wishlist
Hekto
Home
Pages
Products
Blog
Shop
Contact
Default
Create
Structure
Vision
Schedules
DATASET
API VERSION
CUSTOM API VERSION
PERSPECTIVE
QUERY URL (COPY TO CLIPBOARD)
production
Other
v2025-01-26
raw
https://n2siq0cp.api.sanity.io/v2025-01-26/data/query/prod
QUERY
*[_type == "product"]
description: A modern chair with a carbon fiber frame and bold red accents.
_id: Ij0DfInpyjT43Pus72DVgj
isFeaturedProduct: true
category: Chair
image: {…} 2 properties
_type: image
asset: {…} 2 properties
_type: reference
_ref: image-cc09d3ce34a323e7d60432a366145277d5bb3c4f-606x528-png
_rev: Ij0DfInpyjT43Pus72DVct
price: 2000
_createdAt: 2025-01-27T04:24:06Z
name: Futuristic Sleek Modern Chair
_updatedAt: 2025-01-27T04:24:06Z
stockLevel: 3
What's new
Sanity Create Content Mapping, Visual Editing,

Code Snippets for API Integration and Migration:

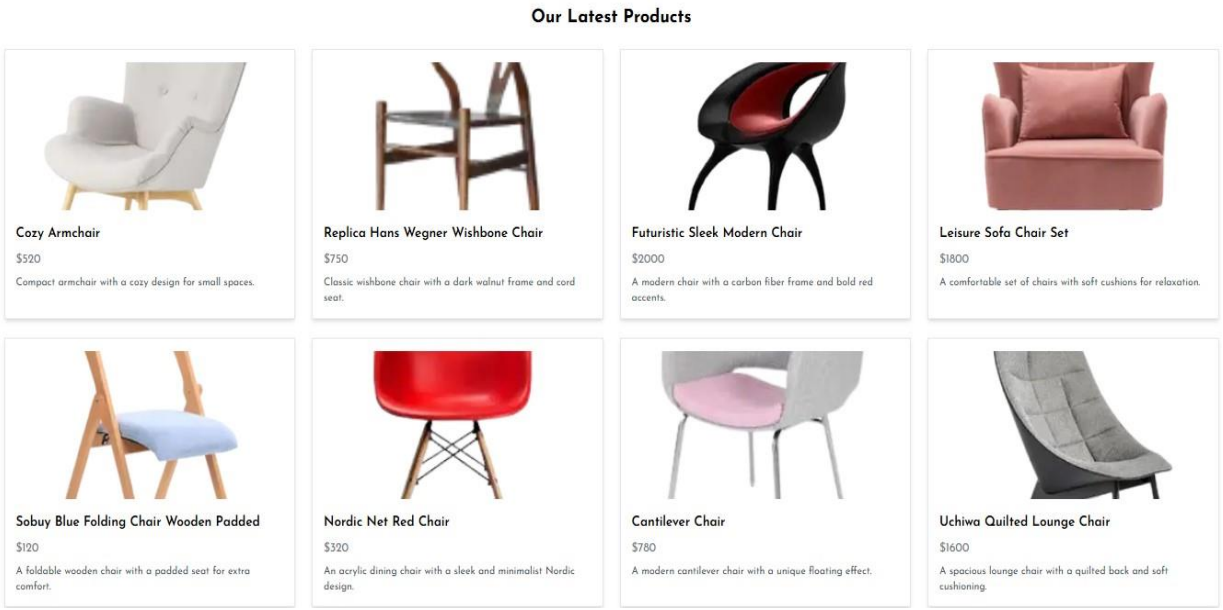


```
async function importData() {
  try {
    console.log('Fetching Product Data From API ...');

    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: {
```



```
types > TS products.ts > Products > imageUrl
export interface product {
    image:Image,
    _id:string,
    name:string,
    price:number,
    slug:string,
    category:string,
    imagesUrl:string[],
    hoverImageUrl:string,
    description:string,
    discountPercentage:string
}

export interface Products {
    _id:string,
    image:Image,
    imageUrl:string[],
    name:string,
    description:string,
    slug:string,
    category:string
    price:number,
    product_id:string,
    discountPercentage:string

}
```

Data Successfully Displayed Frontend



Day-3 Checklist

Self-Validation Check List				
API Understanding	Schema Validation	Data Migration	API Integration in Next.js	Submission Preparation
✓	✓	✓	✓	✓

Conclusion:

By following these steps, you can successfully integrate an external API into Sanity CMS and fetch data into your Next.js e-commerce project. This setup ensures scalability and maintainability for your furniture website.