HACKATHON

Day3 API Integration and Data Migration

Created by: NADIA

nadiaawan400@gmail.com

Submitted to: AMEEN ALAM

Day 3 - API Integration and Data Migration Report - Bandage

Objective:

The goal for Day 3 was to integrate API data into Sanity CMS for the Bandage project,
enabling dynamic content updates for the marketplace. Instead of manually entering
data, the API integration provided a more efficient and salable solution.

1. Sanity CMS Schema Design:

To ensure the seamless handling of product data, I designed a schema called `product`
in Sanity CMS. The schema includes the following fields:

-

Main Fields:

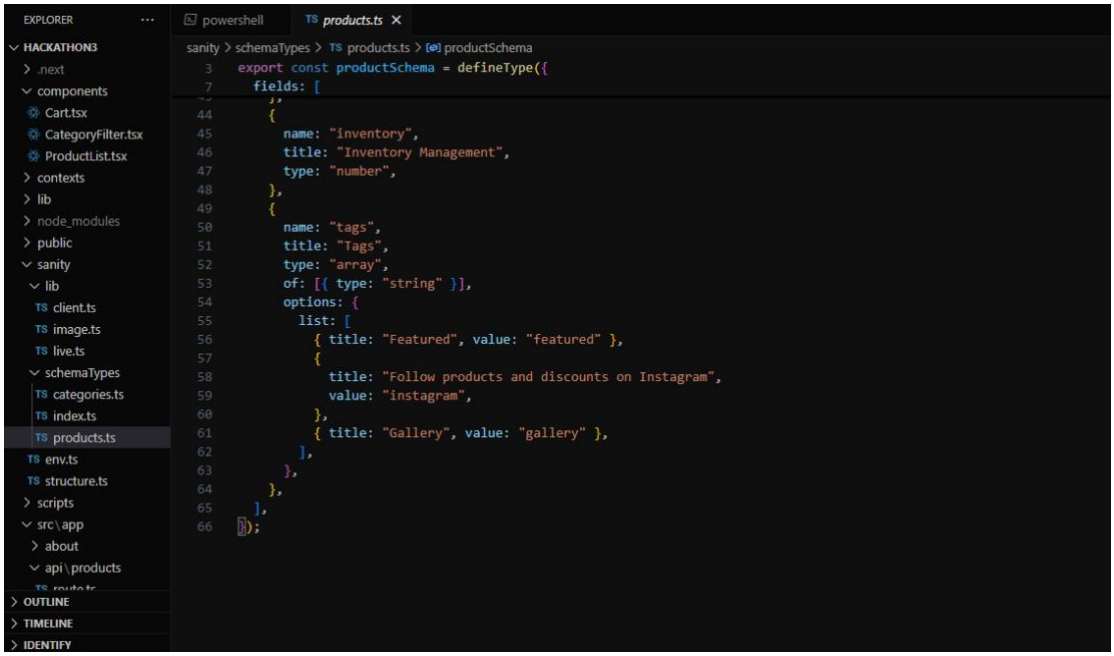
- `title`: The product title (string type).
- `description`: A detailed description of the product (text type).
- `product_Image`: The main product image (image type).
- `price`: The price of the product (number type).
- `tags`: An array of tags to categorize the product (array of strings).
- `discount_Percentage`: The discount percentage (number type).
- `is_New`: A boolean flag indicating if the product is new (boolean type).

Code Snippet:

```
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",
```

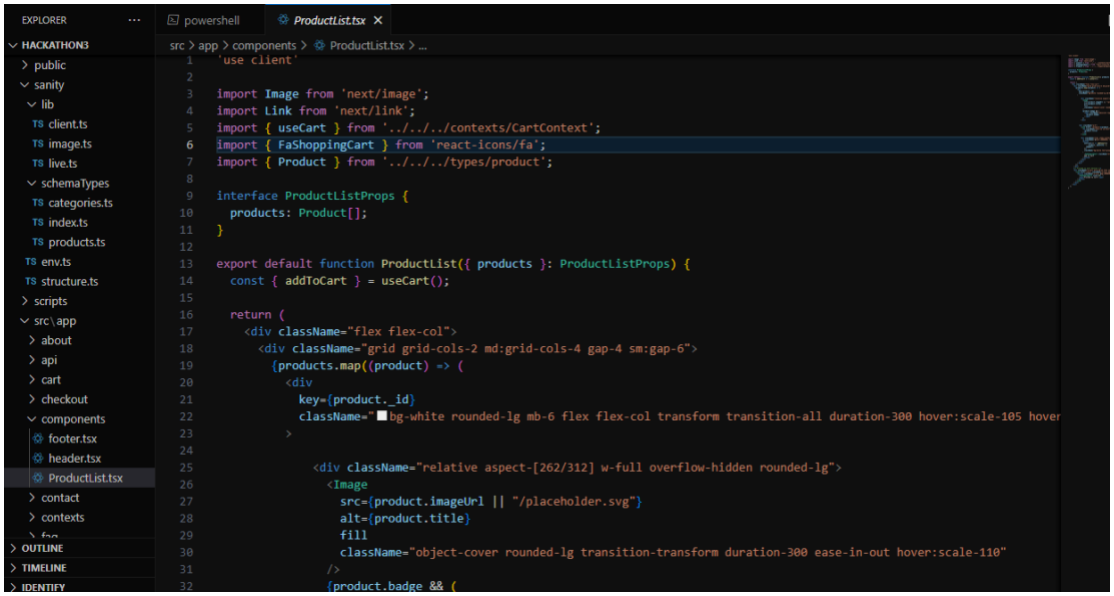
```
  fields: [
      name: "image",
      title: "Product Image",
      type: "image",
    },
    {
      name: "category",
      title: "Category",
      type: "reference",
      to: [{ type: "categories" }],
    },
    {
      name: "description",
      title: "Product Description",
      type: "text",
    },
    {
      name: "inventory",
      title: "Inventory Management",
      type: "number",
    },
    {
      name: "tags",
      title: "Tags",
      type: "array",
      of: [{ type: "string" }],
      options: {
        list: [
          { title: "Featured", value: "featured" },
          {
```

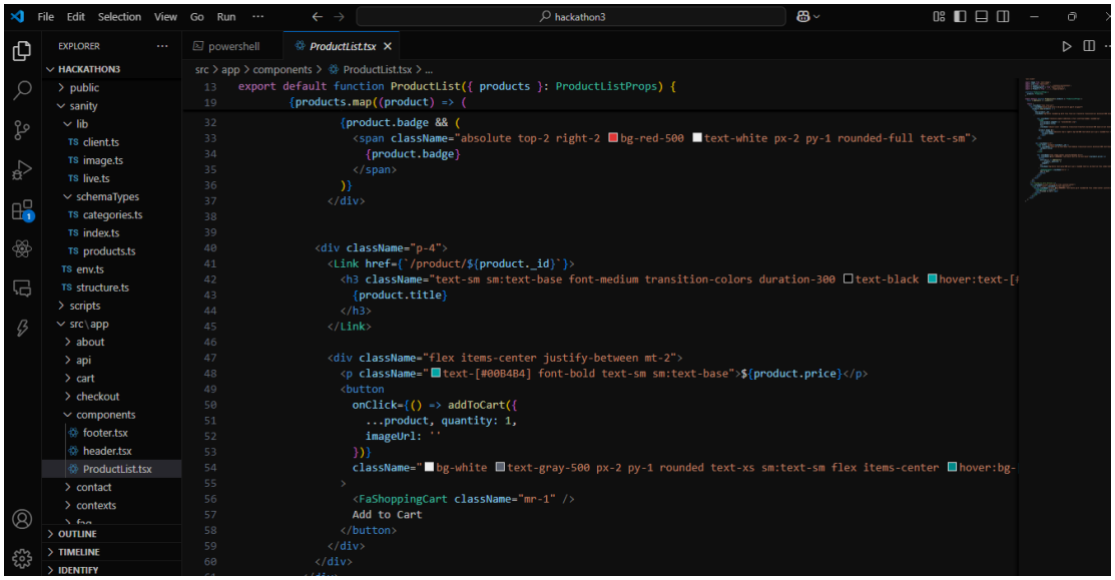


2. API Integration and Data Migration:

API Data Fetching:

I fetched product data from an external API, which included details like images, titles, descriptions, prices, and tags. This data was then mapped to the
corresponding fields in the Sanity CMS schema.



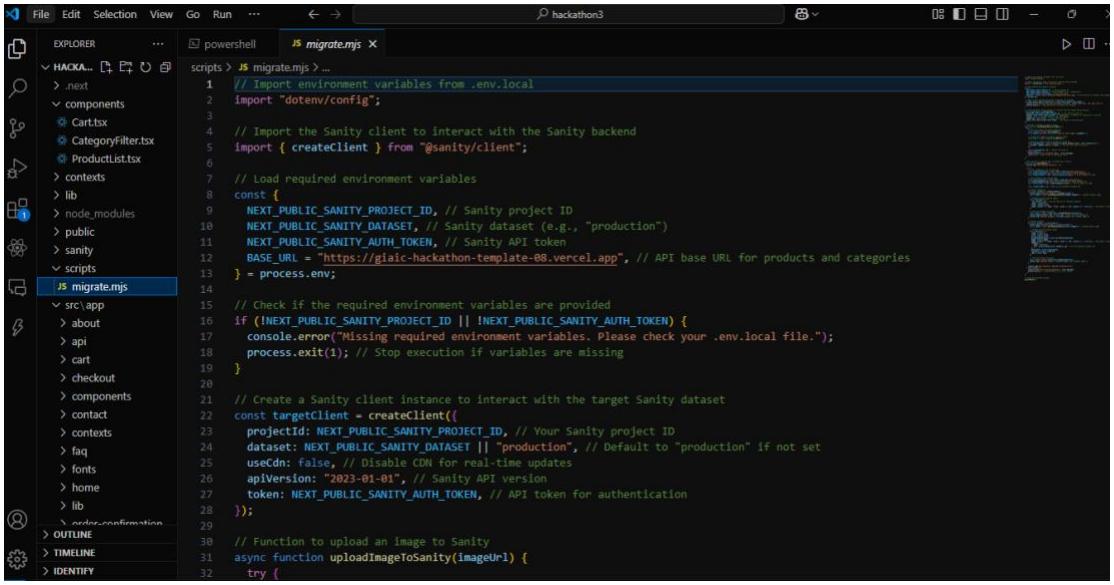


Data Population in Sanity CMS:

After fetching the API data, I populated the product fields in Sanity CMS dynamically. This allowed for the automated population of product information, ensuring consistency and accuracy across the platform.

Data Migration:

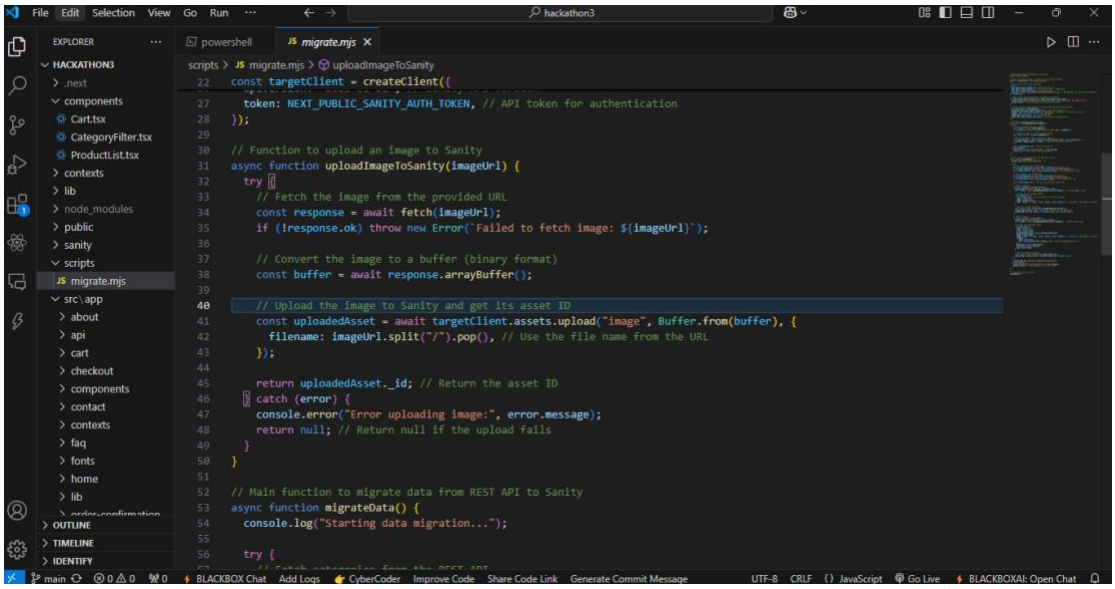
Using the Sanity CLI, I exported the data-set from Sanity CMS for backup
purposes and later re-imported it for testing. This migration ensured that all
data was properly structured and displayed as intended on the front end.



3. Steps Taken for Data Migration:

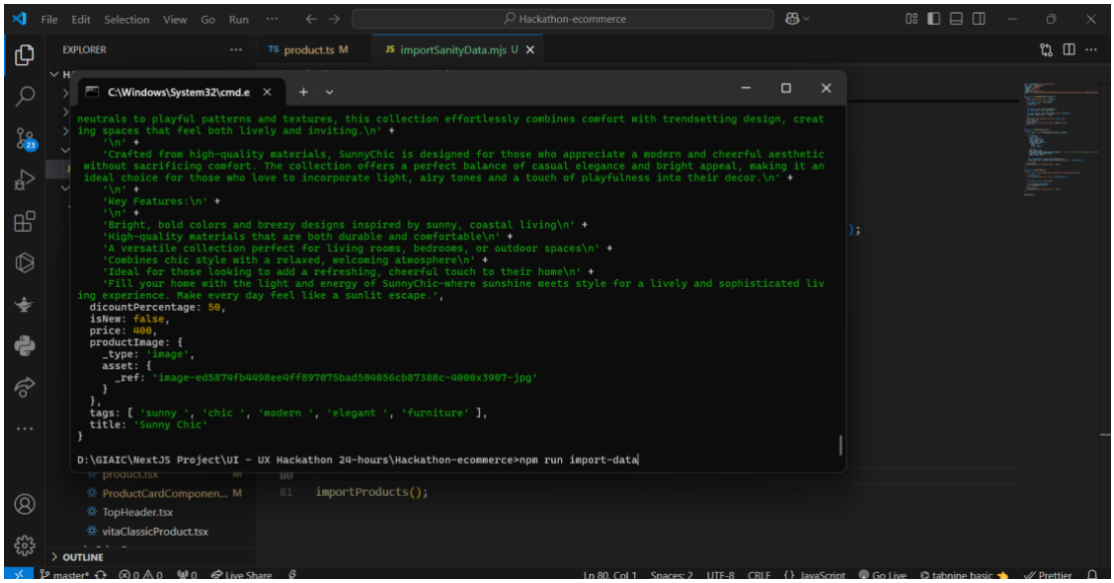
Exporting Data:

The first step was exporting the data from Sanity CMS using the Sanity CLI.
This ensured that all product data was safely backed up before any further operations.



Re-importing Data:

After verification, the data-set was re-imported into Sanity CMS. This confirmed that the data migration was successful and the system was working as expected.

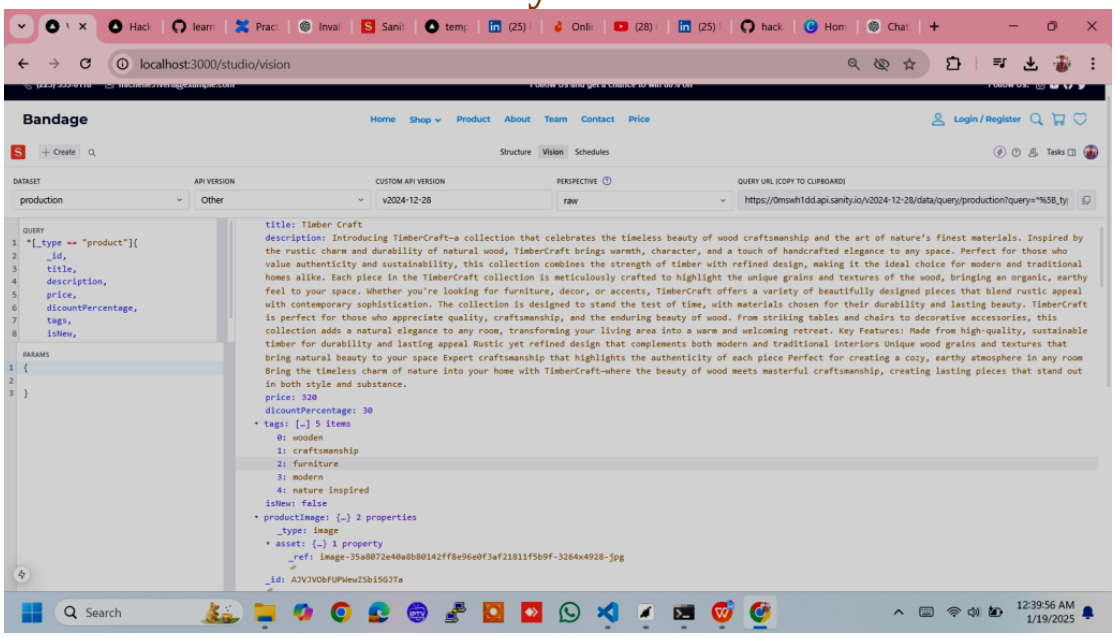


4. Tools Used:

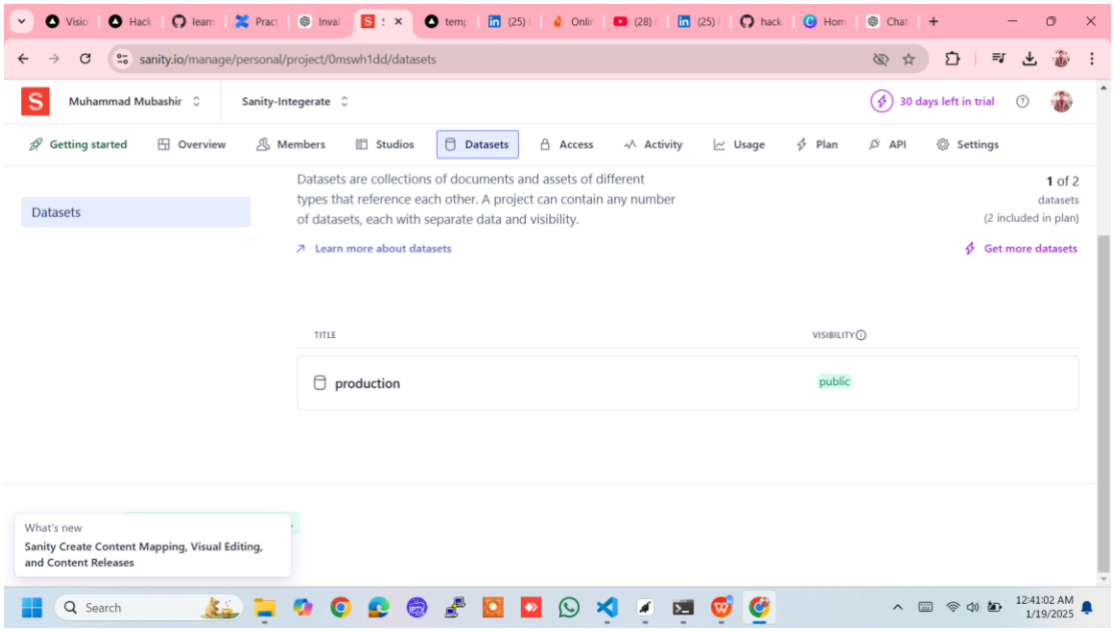
Sanity Studio:

Used for schema creation, content management, and displaying product data.

Sanity Vision



Sanity Database:



Sanity CLI:

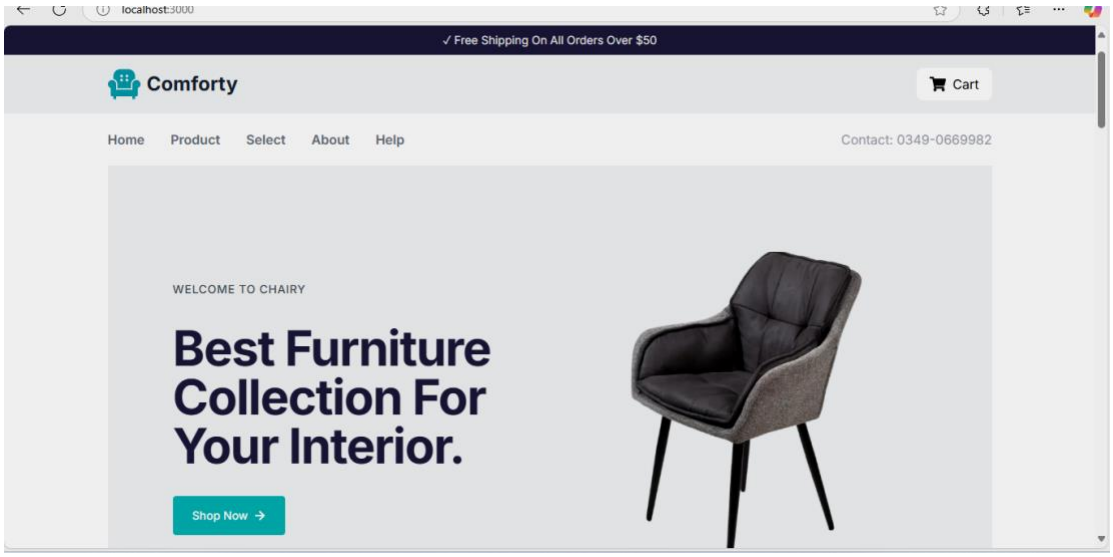
Utilized for exporting and importing the data-set, ensuring data consistency and backup.

5. Screenshots and Frontend Display:

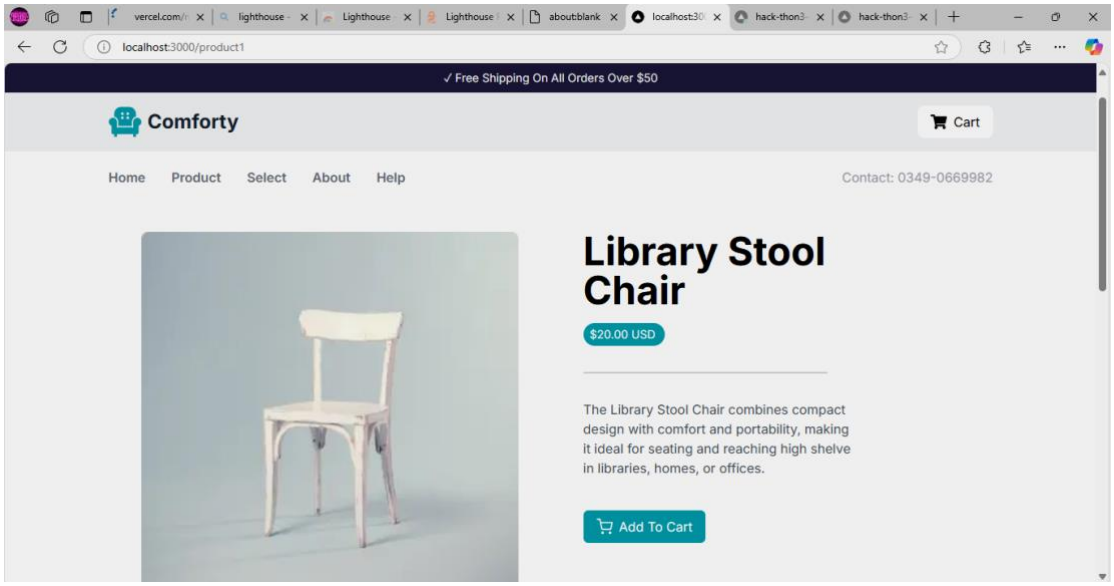
Sanity CMS Fields:

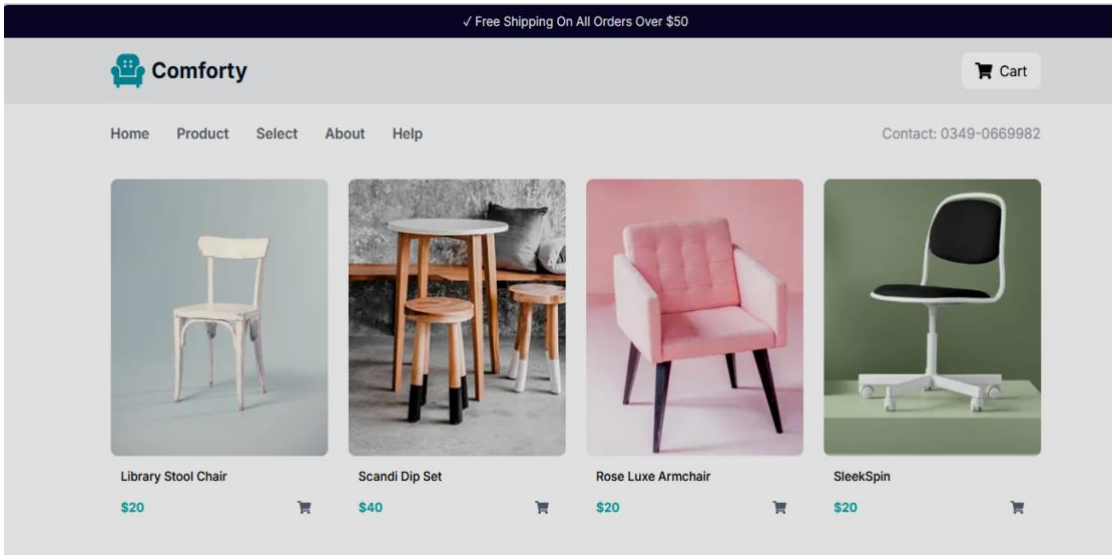
Frontend Display:

Screenshot demonstrating how the product data is displayed dynamically on the front end of the marketplace.



Product Detail Page Display:





Conclusion:

The API integration and data migration were successfully completed, enhancing the efficiency and scalability of the Bandage project. This integration streamlined the process of adding and updating product data in the marketplace, while the migration steps ensured data consistency and accuracy across the system. With this setup, the Bandage project is now more dynamic and easier to maintain.

Checklist

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