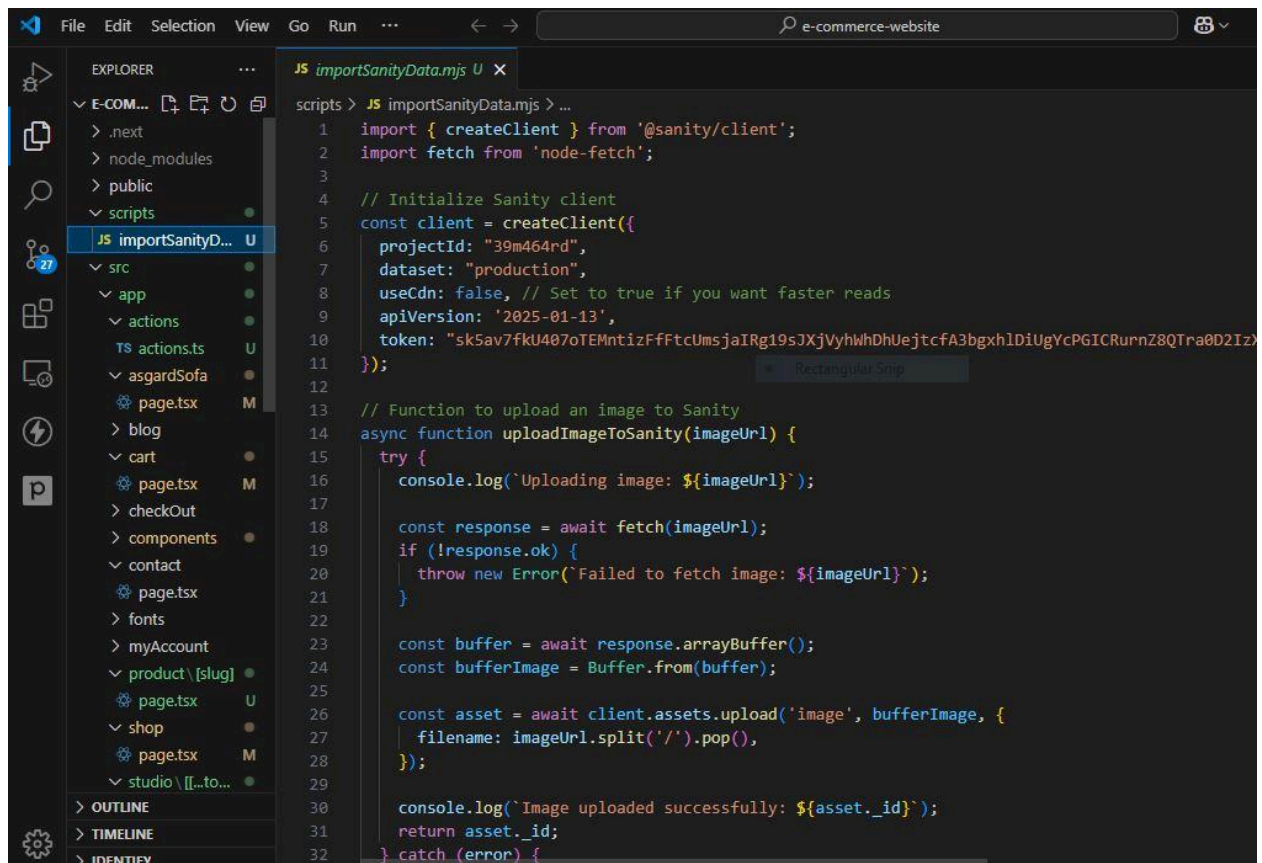


Day 3: API Integration and Data Migration for FurnitureHub Marketplace

This documentation outlines the work completed on Day 3 of the FurnitureHub Marketplace hackathon. It covers custom migration, data integration and sanity, schema creation, and displaying data using GROQ queries in the NextJS application. Each section is tailored based on the provided code images..

Custom Migration Code

The migration code is responsible for transferring data from the sanity CMS to the FurnitureHub database. the custom migration script performs the following steps

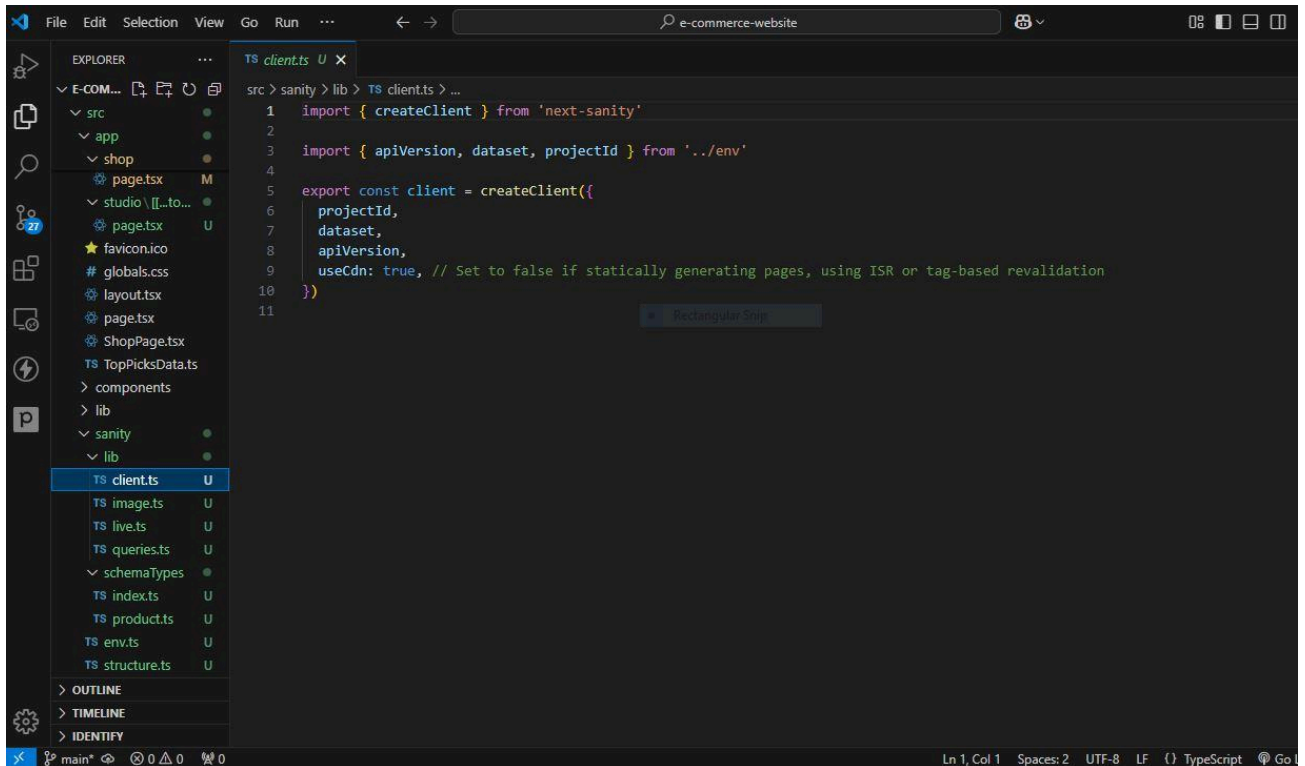


The screenshot shows a Visual Studio Code editor window with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with folders like .next, node_modules, public, and scripts. The scripts folder is expanded, showing a file named 'importSanityData.mjs'. The code editor displays the content of this file, which is a JavaScript script for importing data from Sanity CMS to a database. The script includes imports for 'createClient' and 'fetch', initializes a Sanity client, and defines an async function 'uploadImageToSanity' that handles image uploads and error handling.

```
1 import { createClient } from '@sanity/client';
2 import fetch from 'node-fetch';
3
4 // Initialize Sanity client
5 const client = createClient({
6   projectId: "39m464rd",
7   dataset: "production",
8   useCdn: false, // Set to true if you want faster reads
9   apiVersion: '2025-01-13',
10  token: "sk5av7fkU407oTEMntizFFftcUmsjaIRg19sJXjVyhWhDhUejtcfA3bgxh1DiUgYcPGICRurnZ8QTra0D2Iz",
11 });
12
13 // Function to upload an image to Sanity
14 async function uploadImageToSanity(imageUrl) {
15   try {
16     console.log(`Uploading image: ${imageUrl}`);
17
18     const response = await fetch(imageUrl);
19     if (!response.ok) {
20       throw new Error(`Failed to fetch image: ${imageUrl}`);
21     }
22
23     const buffer = await response.arrayBuffer();
24     const bufferImage = Buffer.from(buffer);
25
26     const asset = await client.assets.upload('image', bufferImage, {
27       filename: imageUrl.split('/').pop(),
28     });
29
30     console.log(`Image uploaded successfully: ${asset._id}`);
31     return asset._id;
32   } catch (error) {
```

Client Page Code

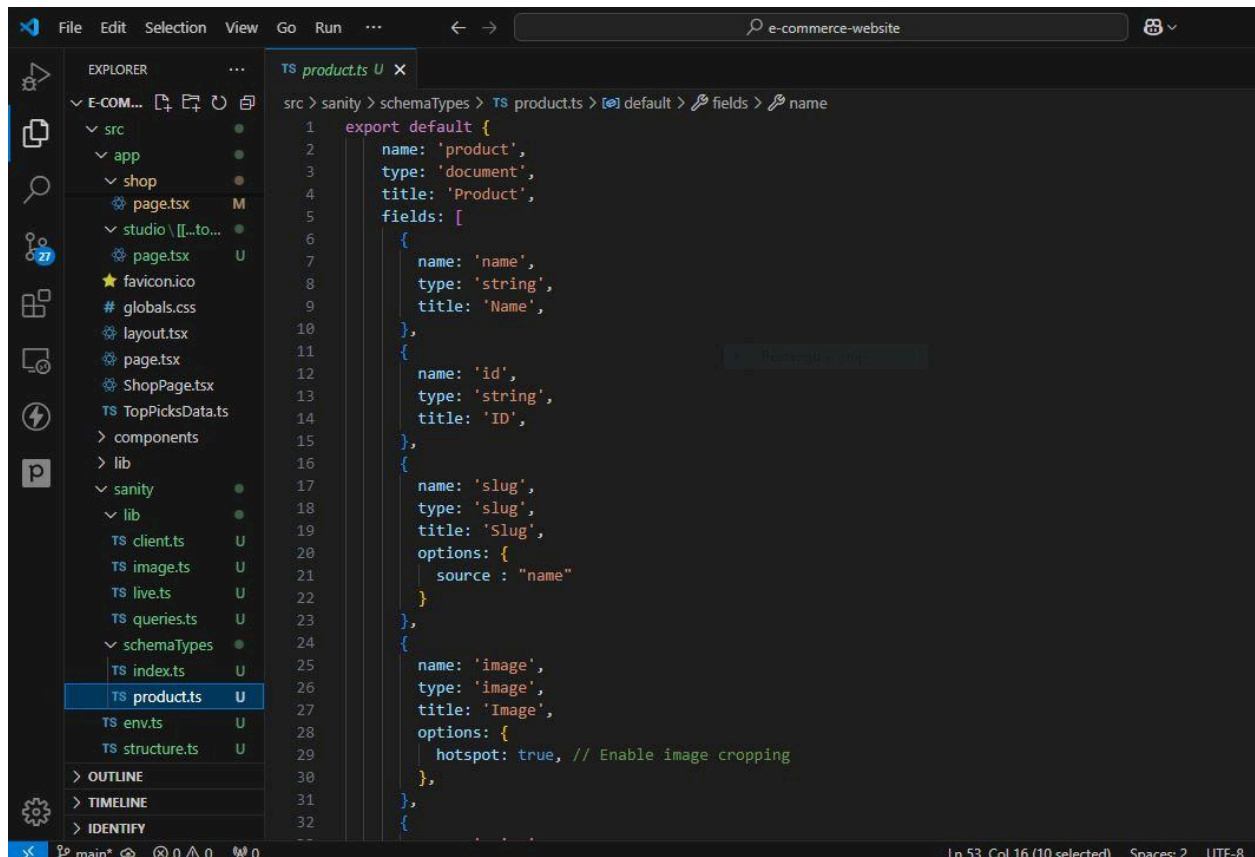
This is the client-side code for rendering the FurnitureHub data in a NextJS page: Here's the code:



```
src > sanity > lib > TS clients > ...
1  import { createClient } from 'next-sanity'
2
3  import { apiVersion, dataset, projectId } from '../env'
4
5  export const client = createClient({
6    projectId,
7    dataset,
8    apiVersion,
9    useCdn: true, // Set to false if statically generating pages, using ISR or tag-based revalidation
10 })
11
```

Schema Code

The schema defines the structure of the FurnitureHub content in the Sanity CMS.

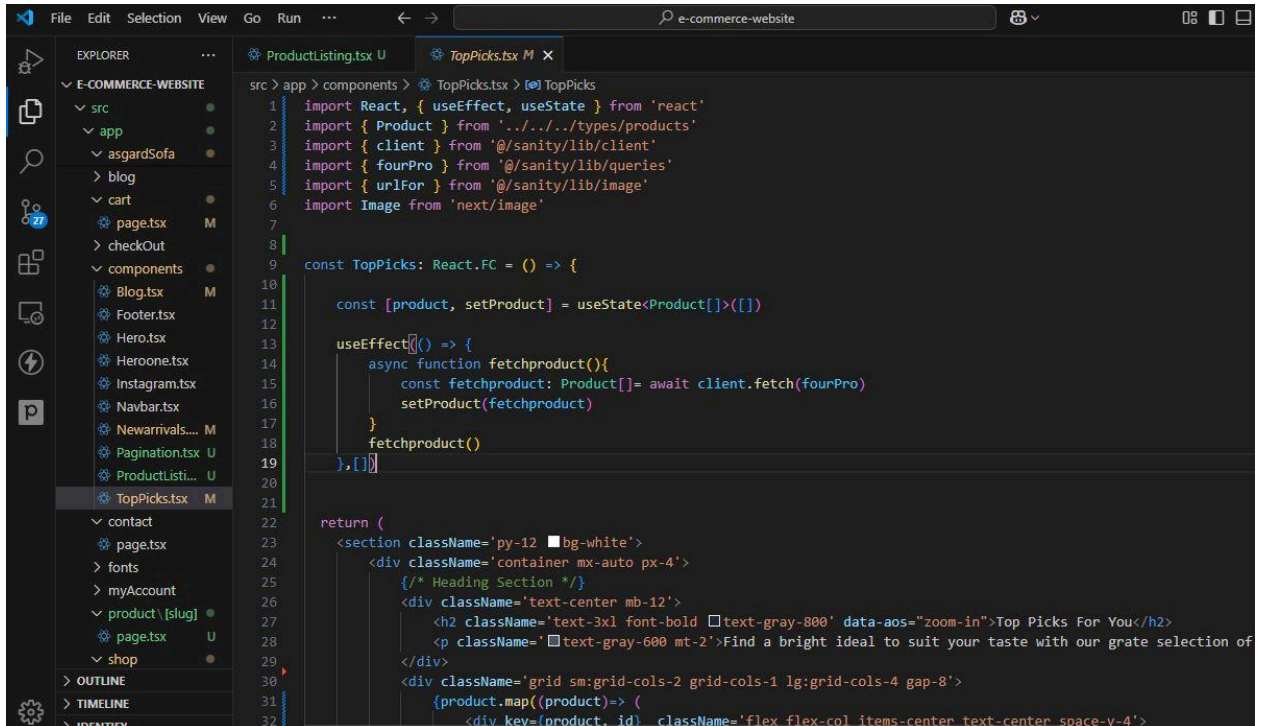


The screenshot shows a Visual Studio Code editor with a project named 'e-commerce-website'. The Explorer sidebar on the left shows the file structure, with 'src > sanity > schemaTypes > TS product.ts' selected. The main editor displays the content of 'product.ts', which defines a Sanity schema for a product. The schema includes fields for name, id, slug, and image, each with specific types and options. The breadcrumb at the top of the editor indicates the current path: 'src > sanity > schemaTypes > TS product.ts > default > fields > name'.

```
1 export default {
2   name: 'product',
3   type: 'document',
4   title: 'Product',
5   fields: [
6     {
7       name: 'name',
8       type: 'string',
9       title: 'Name',
10    },
11    {
12      name: 'id',
13      type: 'string',
14      title: 'ID',
15    },
16    {
17      name: 'slug',
18      type: 'slug',
19      title: 'Slug',
20      options: {
21        source: "name"
22      }
23    },
24    {
25      name: 'image',
26      type: 'image',
27      title: 'Image',
28      options: {
29        hotspot: true, // Enable image cropping
30      },
31    },
32  ],
33 }
```

TopPicks Card Code

This code represents the design and functionality 4 furniture toppicks card code:



```
File Edit Selection View Go Run ... < -> e-commerce-website
EXPLORER
  E-COMMERCE-WEBSITE
    src
      app
        asgardSofa
        blog
        cart
          page.tsx M
        checkOut
        components
          Blog.tsx M
          Footer.tsx
          Hero.tsx
          HeroOne.tsx
          Instagram.tsx
          Navbar.tsx
          Newarrivals... M
          Pagination.tsx U
          ProductList... U
          TopPicks.tsx M
        contact
          page.tsx
        fonts
        myAccount
        product\ [slug]
          page.tsx U
        shop
      OUTLINE
      TIMELINE
      IDENTIFY

src > app > components > TopPicks.tsx > TopPicks
1 import React, { useEffect, useState } from 'react'
2 import { Product } from '../../types/products'
3 import { client } from '@sanity/lib/client'
4 import { fourPro } from '@sanity/lib/queries'
5 import { urlFor } from '@sanity/lib/image'
6 import Image from 'next/image'
7
8
9 const TopPicks: React.FC = () => {
10
11   const [product, setProduct] = useState<Product[]>([])
12
13   useEffect(() => {
14     async function fetchproduct(){
15       const fetchproduct: Product[] = await client.fetch(fourPro)
16       setProduct(fetchproduct)
17     }
18     fetchproduct()
19   }, [])
20
21
22   return (
23     <section className='py-12 bg-white'>
24       <div className='container mx-auto px-4'>
25         <div className='text-center mb-12'>
26           <h2 className='text-3xl font-bold text-gray-800 data-aos="zoom-in">Top Picks For You</h2>
27           <p className='text-gray-600 mt-2'>Find a bright ideal to suit your taste with our grate selection of
28         </div>
29         <div className='grid sm:grid-cols-2 grid-cols-1 lg:grid-cols-4 gap-8'>
30           {product.map((product)=> (
31             <div key={product.id} className='flex flex-col items-center text-center space-y-4'>
32
```

Environment Variable

The `.env.local` file includes sensitive configuration for the FurnitureHub application. key entries:

The screenshot shows the VS Code interface. On the left, the Explorer sidebar displays a file tree for a project named 'E-COMMERCE-WEBSITE'. The tree structure is as follows:

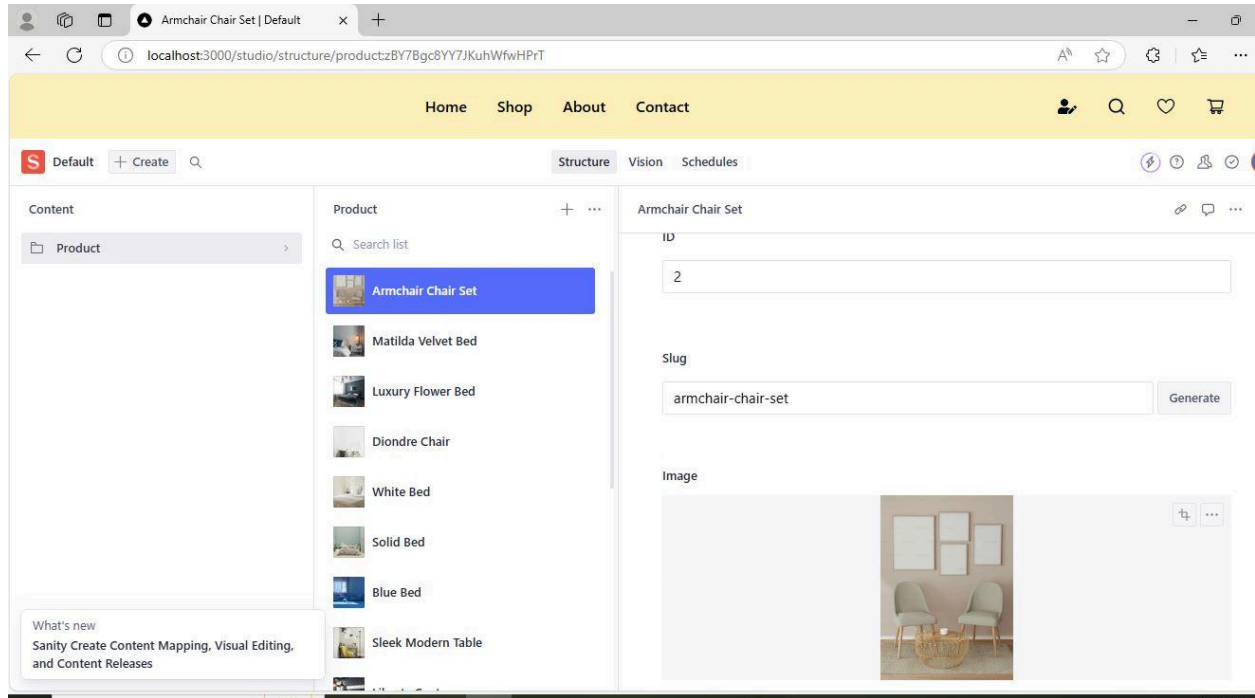
- E-COMMERCE-WEBSITE
 - src
 - app
 - ShopPage.tsx
 - TopPicksData.ts
 - components
 - lib
 - sanity
 - lib
 - client.ts
 - image.ts
 - live.ts
 - queries.ts
 - schemaTypes
 - index.ts
 - product.ts
 - env.ts
 - structure.ts
 - types

At the bottom of the Explorer sidebar, the file '.env.local' is selected. The main editor area on the right displays the contents of this file:

```

1  NEXT_PUBLIC_SANITY_PROJECT_ID=
2  NEXT_PUBLIC_SANITY_DATASET=
3  SANITY_API_TOKEN=
  
```

Sanity Product Schema



Frontend Integration

Each field in the schema is accessible via GROQ queries, allowing developers to fetch the exact data they need. For example:

The screenshot displays the Sanity Studio interface in a web browser at `localhost:3000/studio/vision`. The top navigation bar includes links for Home, Shop, About, and Contact. Below this, the 'Vision' tab is active, showing a query editor and results panel.

Query Editor:

```
QUERY
1 *[_type == "product"]{
2   _id,
3   name,
4   description,
5   "image":image.asset._ref,
6   _type,
7   price,
8 }
```

Params:

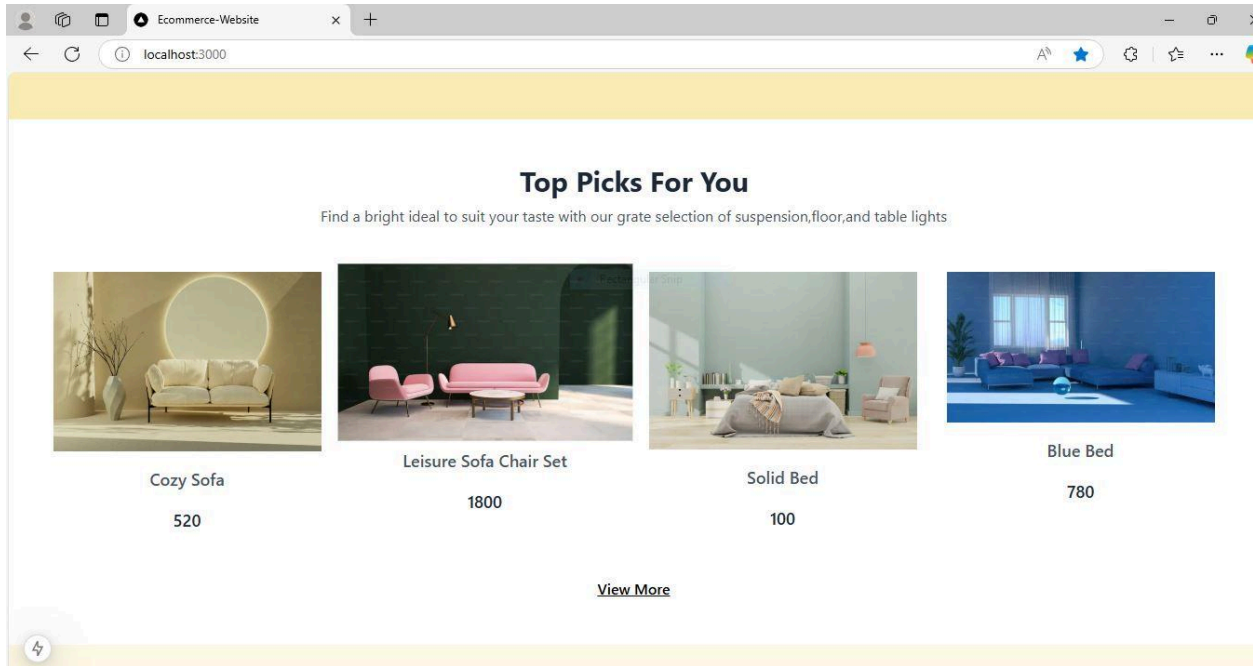
```
1 {
2
3 }
```

Results:

[...] 21 items

- 0: {...} 6 properties
 - _id: Ij0DfInpyjT43Pus58rt7P
 - name: Pink Lounge Chair
 - description: A spacious lounge chair with a quilted back and soft cushioning.
 - image: image-640bd1eecf37b399e04badfba49d99eecf8a1cf7-1964x2455-jpg
 - _type: product
 - price: 1600
- 1: {...} 6 properties
 - _type: product
 - price: 520
 - _id: Ij0DfInpyjT43Pus58rxdd
 - name: Cozy Sofa
 - description: Compact armchair with a cozy design for small spaces.
 - image: image-4cf37cc068700f14b6f8969b1c5bc5f3fd8ded26-2065x1385-jpg
- 2: {...} 6 properties

Show Products in UI



Day 3 Checklist

Day 3 Checklist:

Self-Validation Checklist:

API Understanding:

- ✓ or ✗

Schema Validation:

- ✓ or ✗

Data Migration:

- ✓ or ✗

API Integration in Next.js:

- ✓ or ✗

Submission Preparation:

- ✓ or ✗