Project Overview:

ASRA FURNITURE'S MARKETPLACE E-COMMERCE WEBSITE

Day 1: Marketplace Goals

- **Objective:** Build a marketplace where users can browse and purchase furniture products online.
- **Target Audience:** Homeowners, interior designers, and furniture enthusiasts looking for high-quality furniture.
- **Core Features:**
  - **Product Catalog:** Display a wide range of furniture products with detailed descriptions.
  - **Product Details:** Include product name, price, images, material, dimensions, and more.
  - **Shopping Cart:** Allow users to add products to the cart and proceed to checkout.
  - **User Profiles:** Provide an option for users to create accounts, track orders, and save preferences.
  - **Search and Filter:** Enable users to search products by category, price range, size, material, etc.

Day 2: Technical Breakdown

- **Technology Stack:**
  - **Frontend:** Next.js (React-based framework) for building the user interface and handling routing.
  - **Backend:** Sanity CMS for managing product data and other content.
  - **State Management:** React hooks for managing product states and handling dynamic updates.
  - **Styling:** Tailwind CSS for responsive and modern design.
  - **Image Hosting:** Next.js Image component to handle optimized image rendering for faster loading times.

- **Project Structure:**
  - **Sanity CMS Setup:**
    - Used Sanity to create a product schema and store data for various furniture items.
    - Implemented fields for name, price, description, dimensions, material, image, etc.
    - Set up product categories and optional attributes like discount percentage and stock quantity.
  - **Frontend Setup:**
    - Created a homepage with a product list showcasing various furniture items.
    - Developed a product detail page with a more detailed view of each item.
    - Integrated dynamic routing to fetch product data using the product ID.

Step 1: Fetching Products from Sanity CMS

- **Sanity Migration:**
  - Successfully ran the Sanity migration to set up the product schema for the furniture products.
  - **Fields in Product Schema:**
    - Name (String): The name of the furniture item.
    - Price (String): The price of the product.
    - Image (Image): The image associated with the product.
    - Description (Text): A detailed description of the product.
    - Dimensions (String): The dimensions of the product (e.g., 50 x 30 x 10 cm).
    - Material (String): The material used in the product (e.g., wood, metal).
    - Category (String): The category of the furniture (e.g., sofa, table).
  - **Sanity Query:**
    - Created a query to fetch product data based on the product ID (*[_type == "product" && _id == $id][0]) for detailed pages.
  - **Fetching Products:**
    - Implemented fetching logic in the frontend using useEffect to get product data when the page loads.
    - Used the useParams hook to capture the product ID from the URL and query Sanity for that specific product.
- **Displaying Product Data on Frontend:**
  - **Product Details Page:**
    - Displayed detailed information including the product name, image, description, price, dimensions, and material.
    - Implemented static details like dimensions and material directly in the component for display on the product page.
    - Ensured the UI is responsive and looks good on all devices using Tailwind CSS.
  - **Error Handling:**
    - Added error handling in case the product fetch fails (e.g., product not found).

API Migration:

```
import axios from 'axios';
import { client } from './sanityClient.js';

async function uploadImageToSanity(imageUrl: string): Promise<string> {
  try {
    // Fetch the image from the URL and convert it to a buffer
    const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
    const buffer = Buffer.from(response.data);

    // Upload the image to Sanity
    const asset = await client.assets.upload('image', buffer, {
      filename: imageUrl.split('/').pop(), // Extract the filename from URL
    });

    // Log the asset returned by Sanity for debugging
    console.log('Image uploaded successfully:', asset);

    return asset._id; // Return the uploaded image asset reference ID
  } catch (error) {
    console.error('❌ Failed to upload image:', imageUrl, error);
    throw error;
  }
}


// Function to import data
async function importData() {
  try {
    // Fetch data from the new API
    const response = await axios.get('https://template-0-beta.vercel.app/api/product');
    const products = response.data;

    // Iterate over the products
    for (const product of products) {
      let imageRef = '';

      // Upload image and get asset reference if it exists
      if (product.imagePath) {
        imageRef = await uploadImageToSanity(product.imagePath);
      }
```

```
  async function importData() {

      // Create the sanity product object
      const sanityProduct = {
        _id: `product-${product.id}`, // Prefix the ID to ensure validity
        _type: 'product',
        name: product.name,
        price: product.price,
        discountPercentage: product.discountPercentage || 0,
        tags: product.category ? [product.category] : [], // Adjust based on available data
        image: {
          _type: 'image',
          asset: {
            _type: 'reference',
            _ref: imageRef, // Set the correct asset reference ID
          },
        },
        description: product.description,
        rating: product.rating?.rate || 0,
        ratingCount: product.rating?.count || 0,
      };

      // Log the product before attempting to upload it to Sanity
      console.log('Uploading product:', sanityProduct);

      // Import data into Sanity
      await client.createOrReplace(sanityProduct);
      console.log(`✅ Imported product: ${sanityProduct.name}`);
    }

    console.log('✅ Data import completed!');
  } catch (error) {
    console.error('❌ Error importing data:', error);
  }
}

// Start the data import process
importData();
```

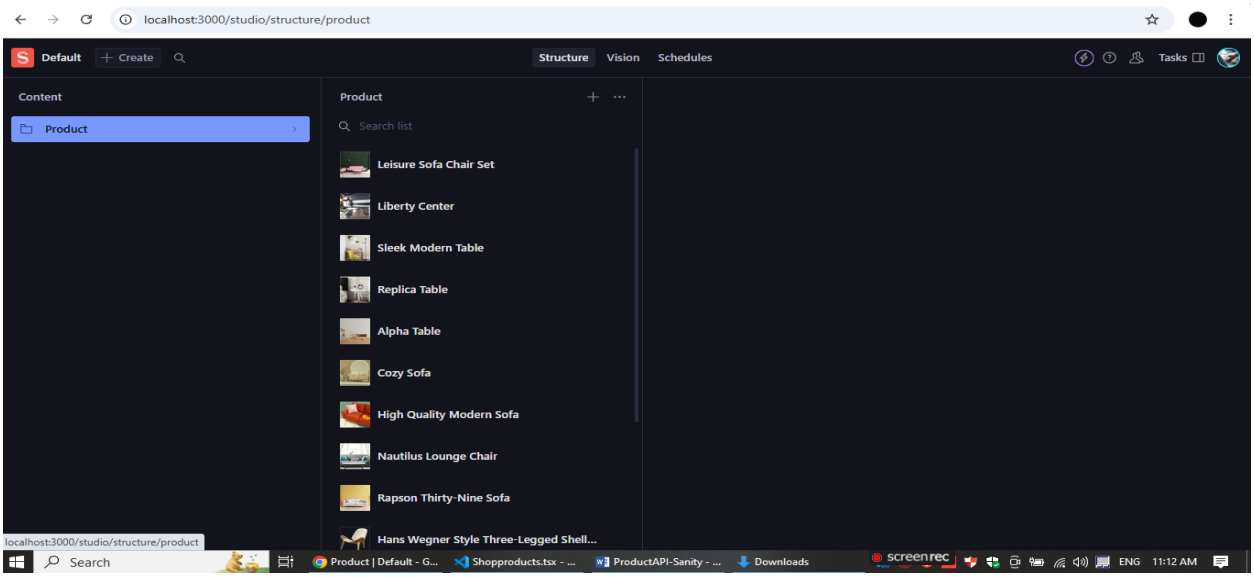
API Integration:

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

  sha1hash: '8a2f7bdc46bcef15dca44c2d55c9597e6eedb0a8',
  size: 474941,
  uploadId: 'YvfmyhZ9s3CxZ0IXFnIQLpggMeQFlhKt',
  url: 'https://cdn.sanity.io/images/vata4oje/production/8a2f7bdc46bcef15dca44c2d55c9597e6eedb0a8-2070x1381.jpg'
}
Uploading product: {
  _id: 'product-13',
  _type: 'product',
  name: 'Liberty Center',
  price: '1100',
  discountPercentage: 8,
  tags: [ 'Table' ],
  image: {
    _type: 'image',
    asset: {
      _type: 'reference',
      _ref: 'image-8a2f7bdc46bcef15dca44c2d55c9597e6eedb0a8-2070x1381-jpg'
    }
  },
  description: 'Versatile entertainment chair for modern interiors.',
  rating: 0,
  ratingCount: 0
}

powershell
node

main*   ⊗ 0 ⚠ 0   0
Ln 7, Col 6   Spaces: 2   UTF-8   CRLF   {} TypeScript   Go Live   Prettier
```

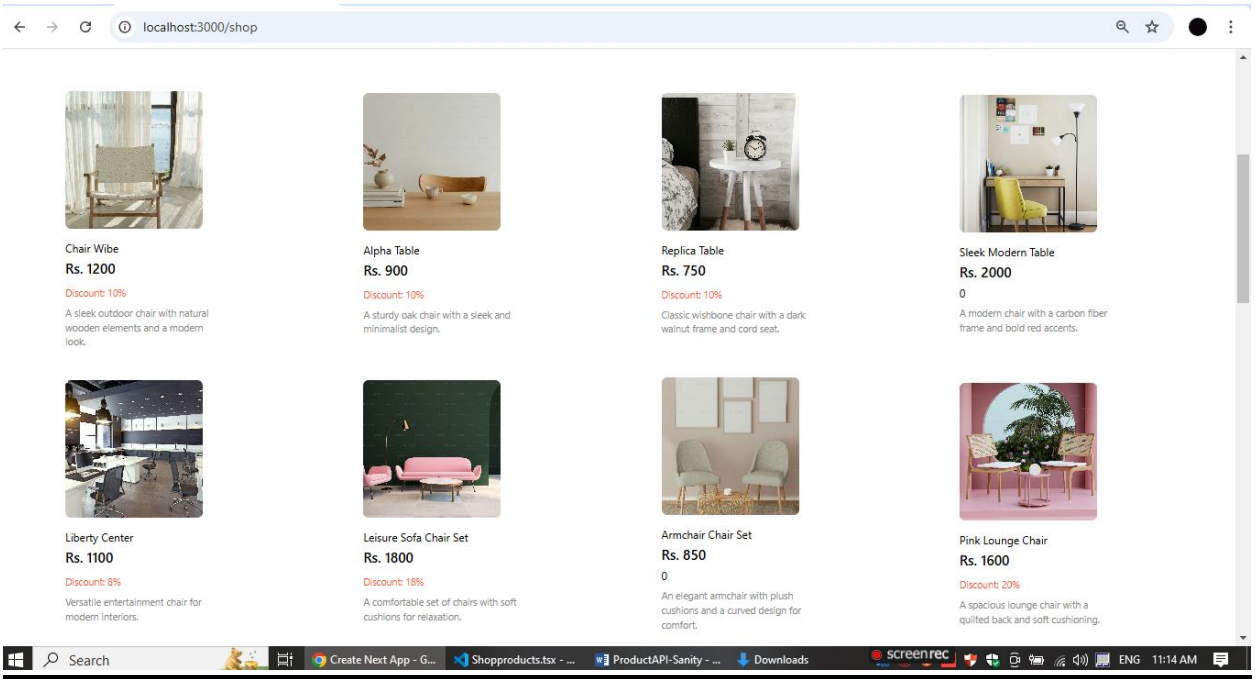
Schema (product.ts):

```
File  Edit  Selection  View  Go  Run  ...   hackathon-3 [Administrator]

TS product.ts U

sanity > schemaTypes > TS product.ts > default > fields

export default {
  name: 'product',
  title: 'Product',
  type: 'document',
  fields: [
    {
      name: 'name',
      title: 'Name',
      type: 'string',
    },
    {
      name: 'discountPercentage',
      title: 'Discount Percentage',
      type: 'number',
    },
    {
      name: 'category',
      title: 'Category',
      type: 'string',
    },
    {
      name: 'description',
      title: 'Description',
      type: 'text',
    },
    {
      (property) title: string
      title: 'Image',
      type: 'image',
      options: {
        hotspot: true,
      },

main*   ⊗ 0 ⚠ 0   0
Ln 14, Col 36   Spaces: 2   UTF-8   CRLF   {} TypeScript   Go Live   Prettier
```

Sanity Studio:



Successfully fetched:



Self-Validation Checklist

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

Future Steps and Enhancements

- **User Authentication:** Add user login and account management.
- **Shopping Cart and Checkout:** Implement the shopping cart functionality and integrate a payment gateway.
- **Search and Filters:** Add a search bar and filters for users to easily find products based on categories, price, and material.
- **Product Reviews and Ratings:** Allow users to rate and review products to provide feedback and enhance user trust.