"Hackahton 3 (Day 3)"

API INTEGRATION AND DATA MIGRATION

Api Data

Pretty-print [['imageUn': 'https://dom.sanity.io/images/?xt4qcah/production/2210cafc285cc13a2d5f8maa36cbea852al1735-305x375.ppg", "price".210. 'tags': ['rustic ", 'vase ", 'home door", 'vintage ", 'interior design'), 'dicountPercentage '18, 'description' 'Bring the chair of nature into your home with the Nastic Vase Set Perfor for those who appreciate timeless bearry and yours, Inviting carfet the control of the control of the natural production of the uniquely designed years, and with its own character. Their earthy tones, textured finishes, and artisanal touch capture the essence of the countryside, making them ideal for shoucasing fresh flowers, dried arrangements, or simply as stand-alone decor pieces. Whether placed on a mantel, coffee table, or dining area, these vases effortles penals and an environment with the natural standard and unability. The natural, imperfect surfaces of the vases give them a distinct, hand-crafted finishes, and artisanal touch capture the essence of the countryside, making them ideal for shoucasing fresh flowers, dried arrangements, or simply as stand-alone decor pieces. Which has the natural. Anador-crafted finish-perfect to fresh disployage flowers, greenery, see asses effortles when the control of the following the standard and the standard an

Api Product Schema

```
o: • = =
                                                                                                                                                                                           т ...
                                                                                              ф
      ▼ FINAL P... (*) □ Final-Project-Ecommerce-Website > final-project > src > sanity > schemaTypes > TS product.ts > ...
                                       import { defineType } from "sanity

√ app

ڡڕ
                                       export const product = defineType({
           # globals.css
                                           name: "product",
title: "Product"
          layout.tsx
           🏶 page.tsx
                                            fields: [
           > lib
(3)

√ schemaTypes

                                                     validation: (rule) => rule.required(),
           TS index.ts
           TS env.ts
                                                     name: "description",
           TS structure.ts
                                                     validation: (rule) => rule.required(),
         gitignore
                                                     title: "Description",
         TS next.config.ts
                                                     type: "image",
validation: (rule) => rule.required(),
         {} package.json

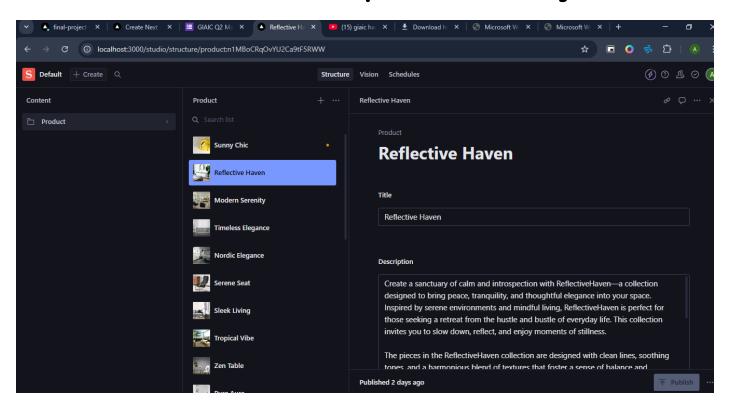
 README.md

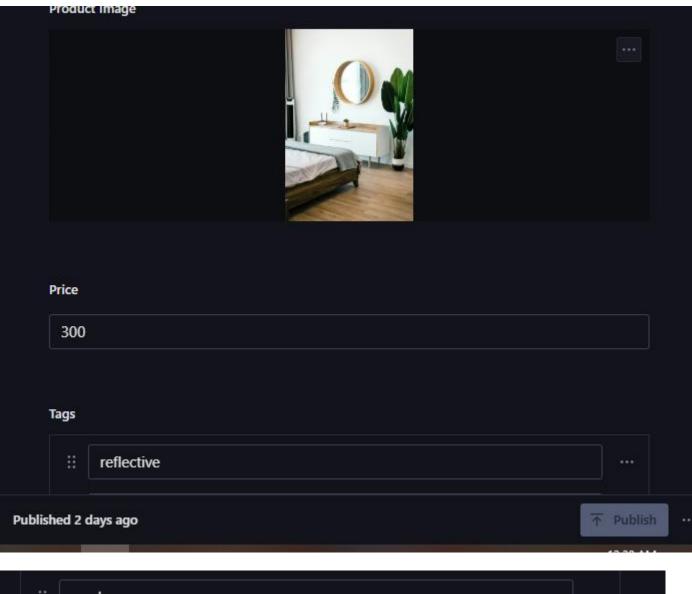
                                                     name: "price",
                                                     type:
                                                     validation: (rule) => rule.required(),
      OUTLINE
       TIMELINE
                                                                                                             Ln 1, Col 1 Spaces: 4 UTF-8 CRLF () TypeScript 😮 Completions limit reached 🛷 Prettier 🚨
         ain → ⊗ 0 Δ 0 💖 0
```

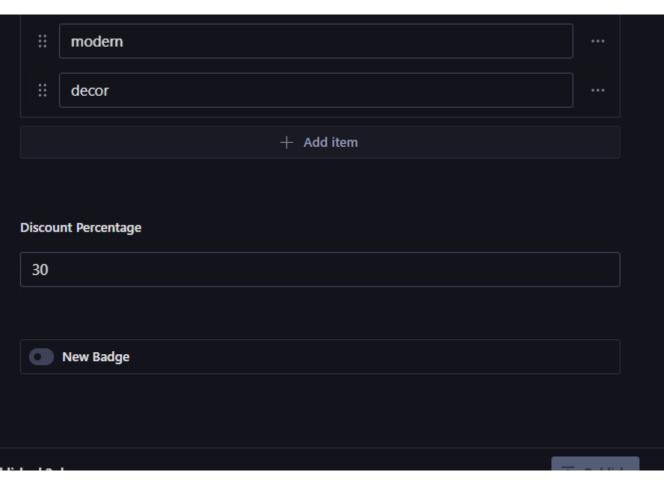
Import Data Code Fetching Code

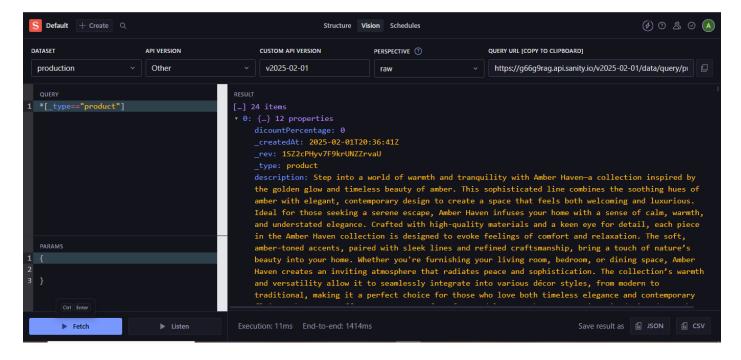
```
File Edit Selection View Go Run
                                                                                                                                                                    JS importData.js X ॐ Fotter.tsx
          import { createClient } from '@sanity/client';
           const client = createClient({
  projectId: "g66g9rag" ,
  dataset: 'production' ,
  useCdn: true,
  apiVersion: '2025-01-13' ,
             token: "skTUO1kgn8gwsd2P2JYBpalGwH0kn4rUudnDdZrUrFxLIyJwIGlmxVgN3kqNPPeJVjxYO1bFGdjFfX7Rsx3j7Deuwn4fR5KiEva4qhxCRtywXCB5812Nl100j6noym6V6rc5FW
           async function uploadImageToSanity(imageUrl) {
               console.log(`Uploading image: ${imageUrl}`);
                const response = await fetch(imageUrl);
               if (!response.ok) {
   throw new Error(`Failed to fetch image: ${imageUrl}`);
               const buffer = await response.arrayBuffer();
               const asset = await client.assets.upload('image', bufferImage, {
                 filename: imageUrl.split('/').pop(),
               console.log(`Image uploaded successfully: ${asset._id}`);
return asset._id;
             } catch (error) {
  console.error('Failed to upload image:', imageUrl, error);
```

Show Product Api Data in Sanity

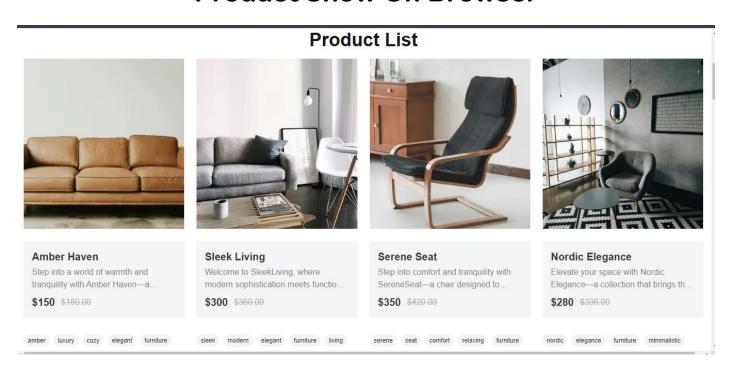


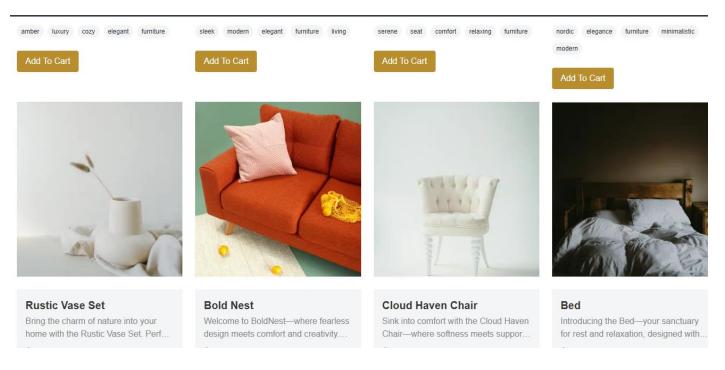






Product Show On Browser





This is my complete work of day 3 hackathon 3