



MARKETPLACE HACKATHON BUILDER 2025

INTRODUCTION

WELCOME TO THE MARKETPLACE BUILDER HACKATHON 2025, THIS IS A 7-DAY CHALLENGE DESIGNED TO HELP YOU LEARN, BUILD, AND LAUNCH YOUR VERY OWN ONLINE MARKETPLACE.

[SIR AMEEN ALAM]

DAY 1: LAYING THE FOUNDATION FOR YOUR MARKETPLACE JOURNEY

INTRODUCTION

On Day 1, it's all about mind storming—just thinking about what I want to create. The main goal of my website is to build an online platform for interior design where users can easily find sofas, lighting, and other interior products, all in one place.

GENERAL E-COMMERCE

1. Business Goals

The goal of my website is to provide a one-stop platform where users can easily find and purchase interior design products like sofas, lighting, and décor without wasting time searching through multiple sites.

2. Target Audience

Our platform is for people who want to decorate or furnish their home. Whether you're a homeowner, renter, designer, we have stylish and affordable furniture, lighting, and décor for you.

I want to create a responsive, pixel-perfect website that makes shopping for stylish interior products easy and enjoyable.

DAY 2 PLANNING THE TECHNICAL FOUNDATION

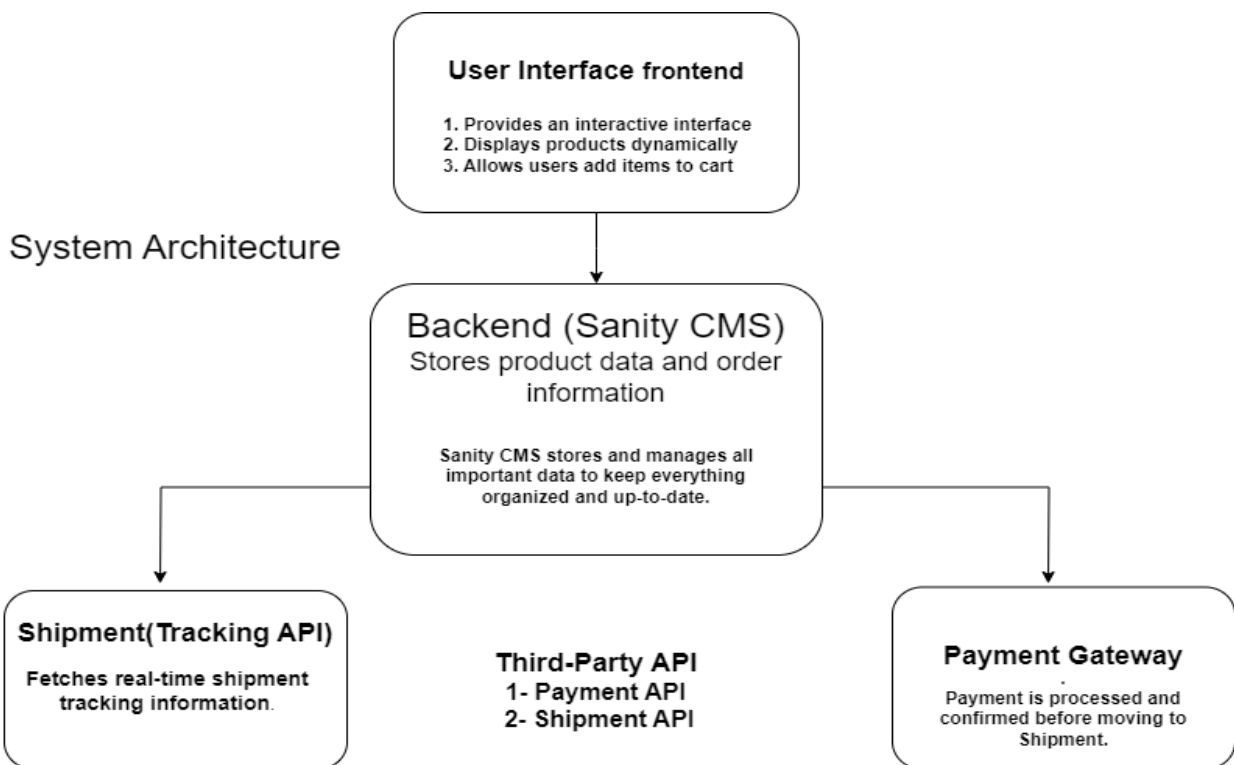
INTRODUCTION

On Day 2, we focus on how the website works. We'll use Sanity for managing content, Next.js for the frontend, and APIs to connect everything, with Sanity handling the backend. Sanity CMS serves as the backend.

Marketplace Technical Foundation-InteriorDesigningWebsite

Flowchart

Flowchart explains the relationship between the frontend, backend (Sanity CMS), and third-party APIs.



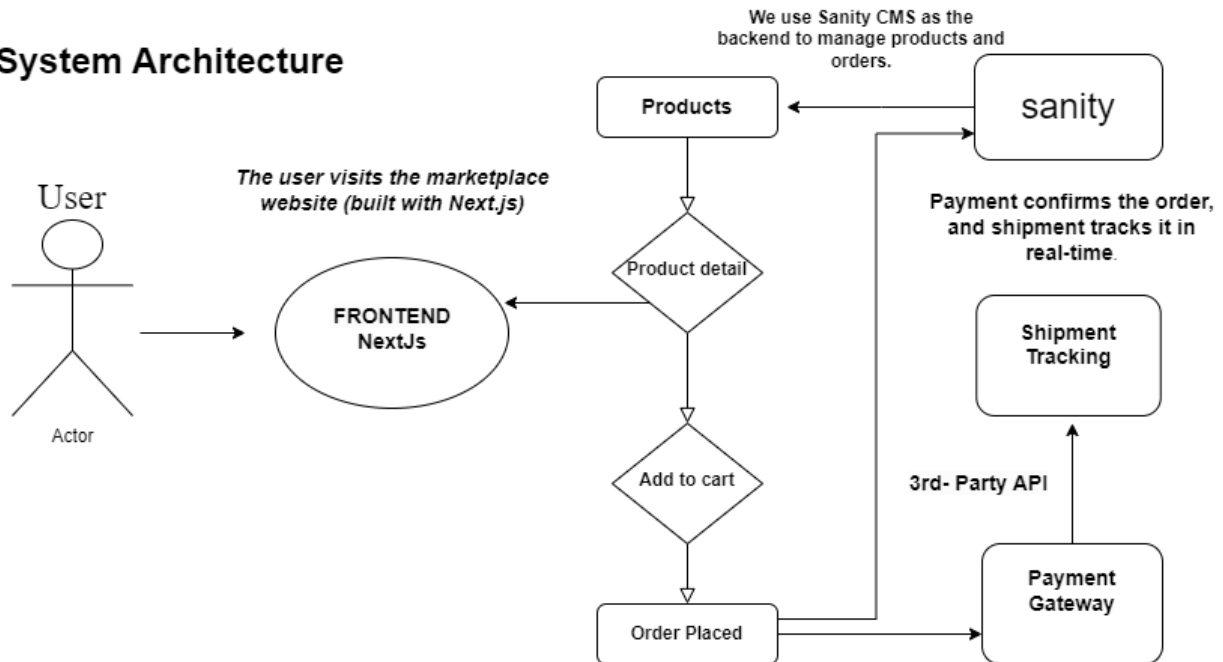
Frontend (Next.js): The website where users browse products.

Sanity CMS: Stores product data and sends it to the website.

Third-Party APIs: Handle payments and shipping, connected to the backend.

Here's How My Website Works:

System Architecture



Here, everything is connected: the frontend links to the backend, where Sanity CMS manages product data, while third-party APIs handle payments and shipping, ensuring a smooth experience from browsing to checkout.

Planning Before Building the Website

Before building the website, it's important to plan ahead. On Day 1 and Day 2, we focused on the technical setup, defining how the website will work, including the integration of the backend, frontend, and third-party APIs for smooth functionality.

DAY 3 - API INTEGRATION AND DATA MIGRATION

INTRODUCTION:

On Day 3, the focus is on integrating APIs and migrating data into Sanity CMS to set up the marketplace backend.

API Integration Report - InteriorDesigningWebsite

An API allows us to integrate third-party services into our project, making it easier to use external features and data.

Using the Provided API in Our Project

Template 6 - Sir Fahad Khan and Sir Hamza Alvi

API: <https://template6-six.vercel.app/api/products>

I have Template 6, and I am using the provided API to integrate data into Sanity for our project.

API Integration Process

- Set up Sanity in your project.
- Create a product file in "schemaTypes" and write the product schema.
- Write a script and add the environment to your project.
- Open Sanity Studio at <https://localhost:3000/studio/>.
- Check the data is imported correctly.

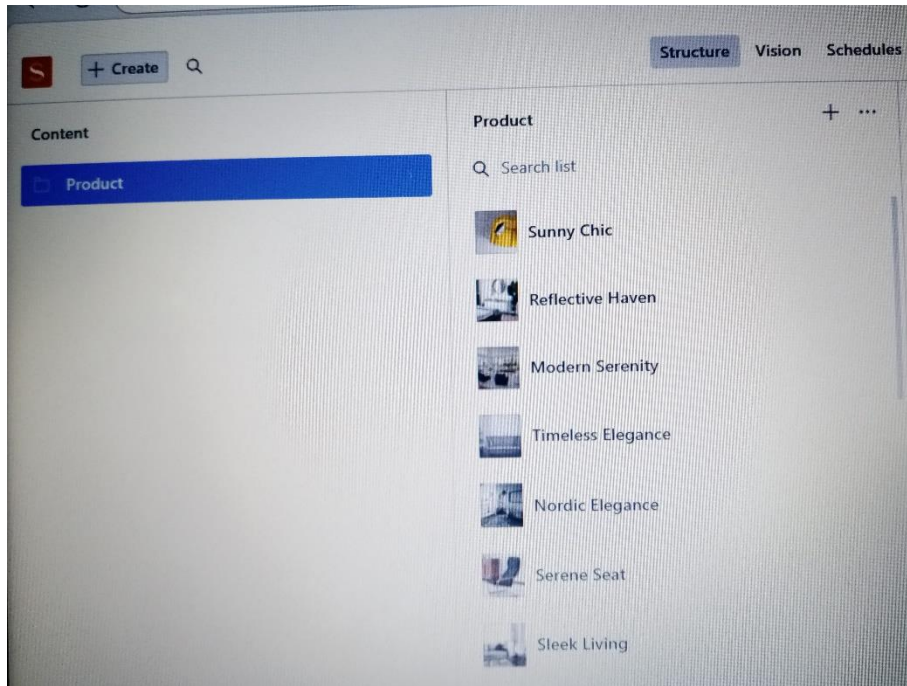
Why We Use Sanity

We use Sanity as our backend to store and manage information, like product details. It helps us easily add, update, and save data, making everything work smoothly on our website.

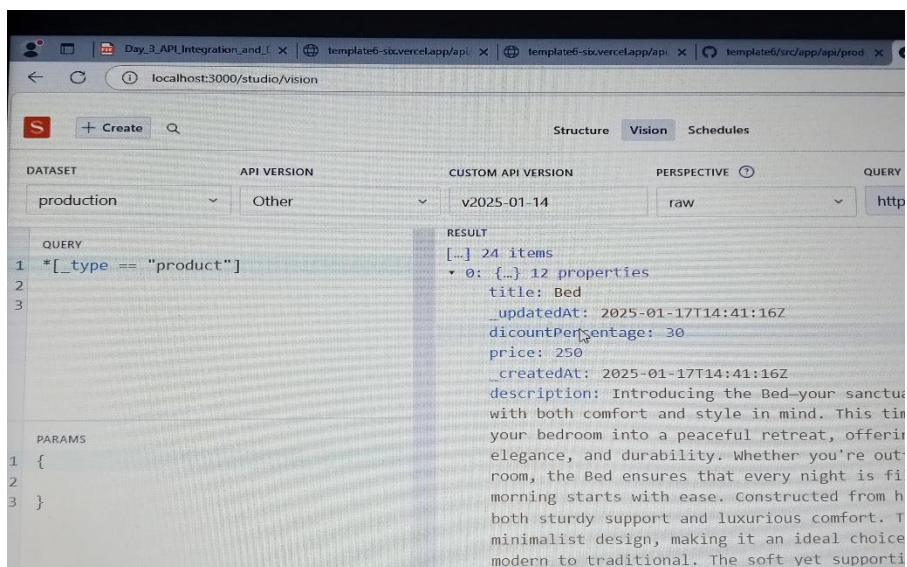
Main purpose of the Day 3 is that we used provided api and integrated its data into Sanity. Then, we fetched the data from Sanity and displayed it on the Frontend.

Snippets

1. Data Successfully Integrated into Sanity



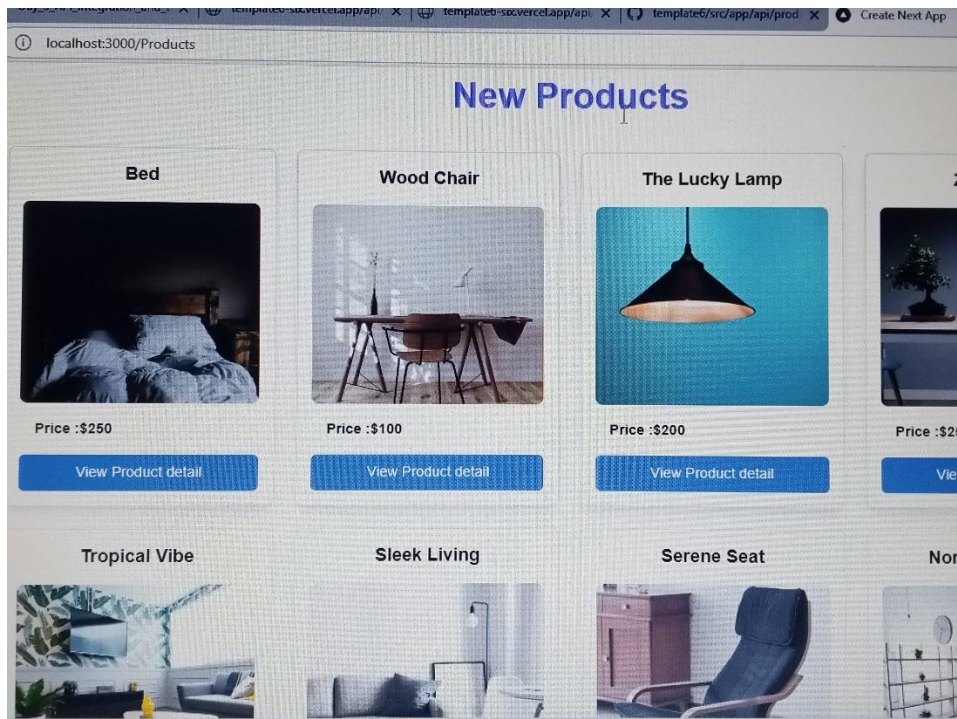
2. Using GROQ Query to Verify Data in Sanity Studio



3. Fetching Data with GROQ Query

```
1  `* [_type == "product" && _id == $id] {  
2    _id,  
3    title,  
4    price,  
5    description,  
6    tags,  
7    "imageUrl": productImage.asset->url,  
8  }`;
```

4. Display Data on Frontend



Script to Fetch and Display Data from Sanity

```
1 import { createClient } from "sanity/client";
2
3 const client = createClient({
4   projectId: "7u32tbp0",
5   dataset: "production",
6   useCdn: true,
7   apiVersion: "2025-01-13",
8   token:
9     "skiy9Klpe7mpoiYhukg7x8LyABh8LkzYEHvD0ZAJCxbpTAgUpz1fe13CaayyXkQYSLLx0BZj1ic2prQaWfCCrJ03BMXwJ5VhbcUIdolhLPnF9l6xoczfXpkySfPH3xNSGhQGUpdeCa7fQx6E5pQ23Kl0TXdWTmuq1g8a2zngU36rX",
10 });
11
12 async function uploadImageToSanity(imageUrl) {
13   try {
14     console.log('Uploading image: ${imageUrl}');
15
16     const response = await fetch(imageUrl);
17     if (!response.ok) {
18       throw new Error('Failed to fetch image: ${imageUrl}');
19     }
20
21     const buffer = await response.arrayBuffer();
22     const bufferImage = Buffer.from(buffer);
23
24     const asset = await client.assets.upload("image", bufferImage, {
25       filename: imageUrl.split("/").pop(),
26     });
27
28     console.log('Image uploaded successfully: ${asset._id}');
29     return asset._id;
30   } catch (error) {
31     console.error('Failed to upload image:', imageUrl, error);
32     return null;
33   }
34 }
35
36 async function uploadProduct(product) {
37   try {
38     const imageId = await uploadImageToSanity(product.imageUrl);
39
40     if (imageId) {
41       const document = {
42         _type: 'product',
43         title: product.title,
44         price: product.price,
45         productImage: {
46           _type: 'image',
47           asset: {
48             _ref: imageId,
49           },
50         },
51         tags: product.tags,
52         discountPercentage: product.discountPercentage, // Typo in field name: discountPercentage -> discountPercentage
53         description: product.description,
54         isNew: product.isNew,
55       };
56
57       const createdProduct = await client.create(document);
58       console.log(
59         `Product ${product.title} uploaded successfully:`,
60         createdProduct
61       );
62     } else {
63       console.log(
64         `Product ${product.title} skipped due to image upload failure.`
65       );
66     }
67   } catch (error) {
68     console.error('Error uploading product:', error);
69   }
70 }
71
72 async function importProducts() {
73   try {
74     const response = await fetch(
75       "https://template6-six.vercel.app/api/products"
76     );
77
78     if (!response.ok) {
79       throw new Error('HTTP error! Status: ${response.status}');
80     }
81
82     const products = await response.json();
83
84     for (const product of products) {
85       await uploadProduct(product);
86     }
87   } catch (error) {
88     console.error('Error fetching products:', error);
89   }
90 }
91
92 importProducts();
93
```


Schema:

A schema defines the structure of the data in Sanity. , making it easier to manage and display consistent content.

For Product:

- **title:** product name
- **description:** A description of the product.
- **productImage:** An image field to upload a product image.
- **tag:** A tag or category for the product.
- **price:** product price.
- **discountPercentage:** The discount percentage applied to the product.
- **is New:** A boolean field to indicate if the product is new.

Code :

```
1  import { defineType } from "sanity"
2
3  export const product = defineType({
4    name: "product",
5    title: "Product",
6    type: "document",
7    fields: [
8      {
9        name: "title",
10       title: "Title",
11       validation: (rule) => rule.required(),
12       type: "string"
13     },
14     {
15       name: "description",
16       type: "text",
17       validation: (rule) => rule.required(),
18       title: "Description",
19     },
20     {
21       name: "productImage",
22       type: "image",
23       validation: (rule) => rule.required(),
24       title: "Product Image"
25     },
26     {
27       name: "price",
28       type: "number",
29       validation: (rule) => rule.required(),
30       title: "Price",
31     },
32     {
33       name: "tags",
34       type: "array",
35       title: "Tags",
36       of: [{ type: "string" }]
37     },
38     {
39       name: "dicountPercentage",
40       type: "number",
41       title: "Discount Percentage",
42     },
43     {
44       name: "isNew",
45       type: "boolean",
46       title: "New Badge",
47     }
48   ]
49 })
```

DAY4: BUILDING DYNAMIC FRONTEND COMPONENTS FOR YOUR MARKETPLACE:

INTRODUCTION:

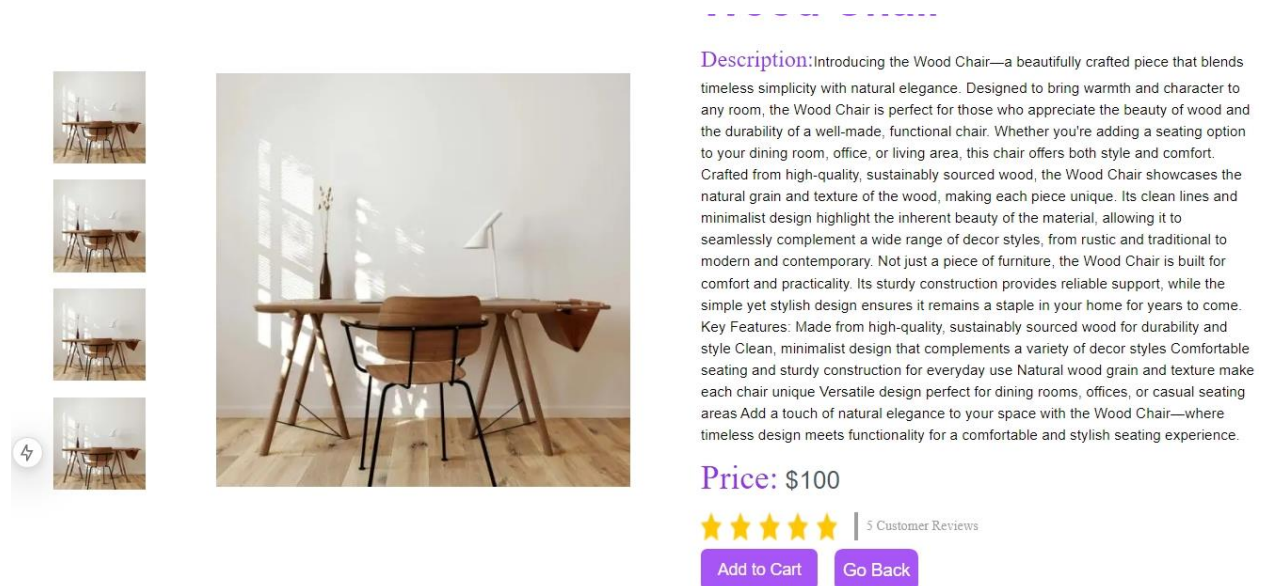
Day4, I created dynamic frontend components to show product details. Each product will have its own page, and I'll use dynamic routes to make the website more flexible and responsive.

Product Detail Page:

On the product detail page, the product's name, price, and image will be displayed, giving users all the information they need about the product.

Dynamic Route Setup

To create a dynamic route, we make a folder with square brackets `[]`, like `[id]`. This allows each product to have its own page based on its unique ID



Description: Introducing the Wood Chair—a beautifully crafted piece that blends timeless simplicity with natural elegance. Designed to bring warmth and character to any room, the Wood Chair is perfect for those who appreciate the beauty of wood and the durability of a well-made, functional chair. Whether you're adding a seating option to your dining room, office, or living area, this chair offers both style and comfort. Crafted from high-quality, sustainably sourced wood, the Wood Chair showcases the natural grain and texture of the wood, making each piece unique. Its clean lines and minimalist design highlight the inherent beauty of the material, allowing it to seamlessly complement a wide range of decor styles, from rustic and traditional to modern and contemporary. Not just a piece of furniture, the Wood Chair is built for comfort and practicality. Its sturdy construction provides reliable support, while the simple yet stylish design ensures it remains a staple in your home for years to come. Key Features: Made from high-quality, sustainably sourced wood for durability and style Clean, minimalist design that complements a variety of decor styles Comfortable seating and sturdy construction for everyday use Natural wood grain and texture make each chair unique Versatile design perfect for dining rooms, offices, or casual seating areas Add a touch of natural elegance to your space with the Wood Chair—where timeless design meets functionality for a comfortable and stylish seating experience.

Price: \$100

★★★★★ | 5 Customer Reviews

[Add to Cart](#) [Go Back](#)

DAY 5: TESTING, ERROR HANDLING, AND BACKEND INTEGRATION REFINEMENT

INTRODUCTION:

On Day 5, I test the website to ensure all functions and APIs work properly, check its responsiveness across devices, and make sure it runs smoothly for real-world use.

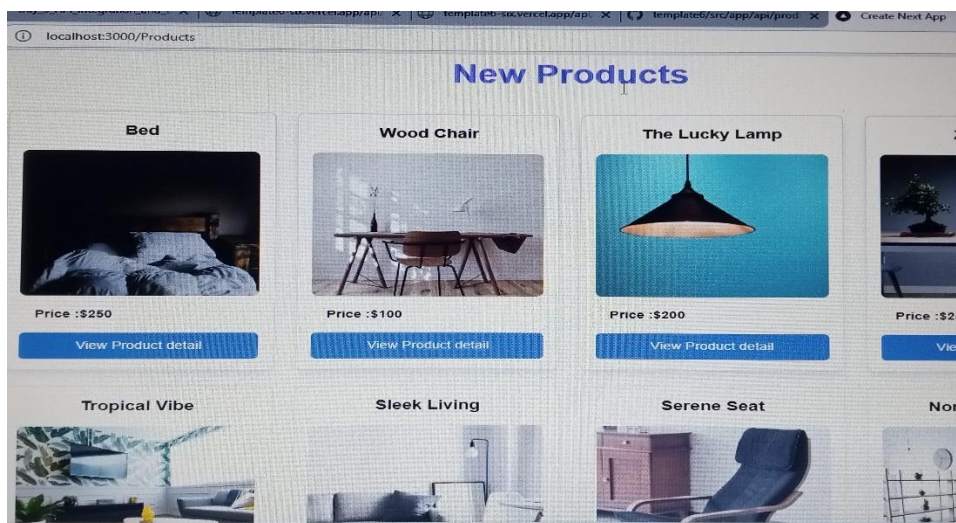
Here are the key steps I take to ensure my website is working properly:

1. Functional Testing

I check that all features, like product display and adding items to the cart, work correctly. User can easily add products to their cart.

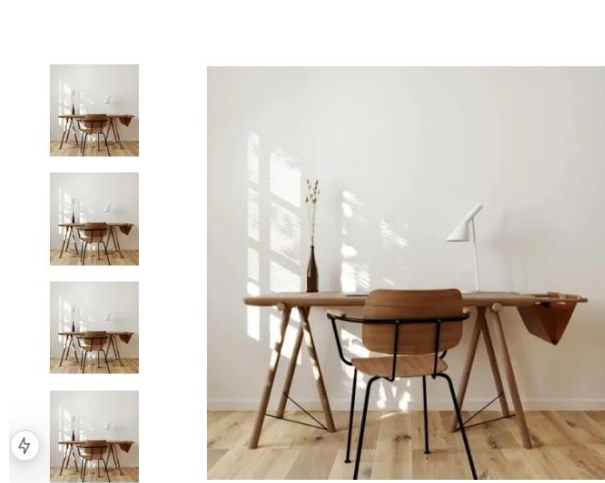
Product

No issue detected in product display, all products are shown perfectly on the website.



Product display

No issue with product details; users can easily view and access all information for each product.



Description: Introducing the Wood Chair—a beautifully crafted piece that blends timeless simplicity with natural elegance. Designed to bring warmth and character to any room, the Wood Chair is perfect for those who appreciate the beauty of wood and the durability of a well-made, functional chair. Whether you're adding a seating option to your dining room, office, or living area, this chair offers both style and comfort. Crafted from high-quality, sustainably sourced wood, the Wood Chair showcases the natural grain and texture of the wood, making each piece unique. Its clean lines and minimalist design highlight the inherent beauty of the material, allowing it to seamlessly complement a wide range of decor styles, from rustic and traditional to modern and contemporary. Not just a piece of furniture, the Wood Chair is built for comfort and practicality. Its sturdy construction provides reliable support, while the simple yet stylish design ensures it remains a staple in your home for years to come.

Key Features: Made from high-quality, sustainably sourced wood for durability and style
Clean, minimalist design that complements a variety of decor styles
Comfortable seating and sturdy construction for everyday use
Natural wood grain and texture make each chair unique
Versatile design perfect for dining rooms, offices, or casual seating areas
Add a touch of natural elegance to your space with the Wood Chair—where timeless design meets functionality for a comfortable and stylish seating experience.

Price: \$100




★★★★★ | 5 Customer Reviews

[Add to Cart](#) [Go Back](#)

Product Add to Cart:

No issue with adding products to the cart; users can easily add items to their cart without any problems.

Your Cart

	<p>Wood Chair</p> <p>Price: \$100 Quantity: 1</p>	Remove
	<p>Sleek Living</p> <p>Price: \$300 Quantity: 1</p>	Remove
	<p>Tropical Vibe</p> <p>Price: \$550 Quantity: 1</p>	Remove

Total: \$950.00

[Proceed to Checkout](#)

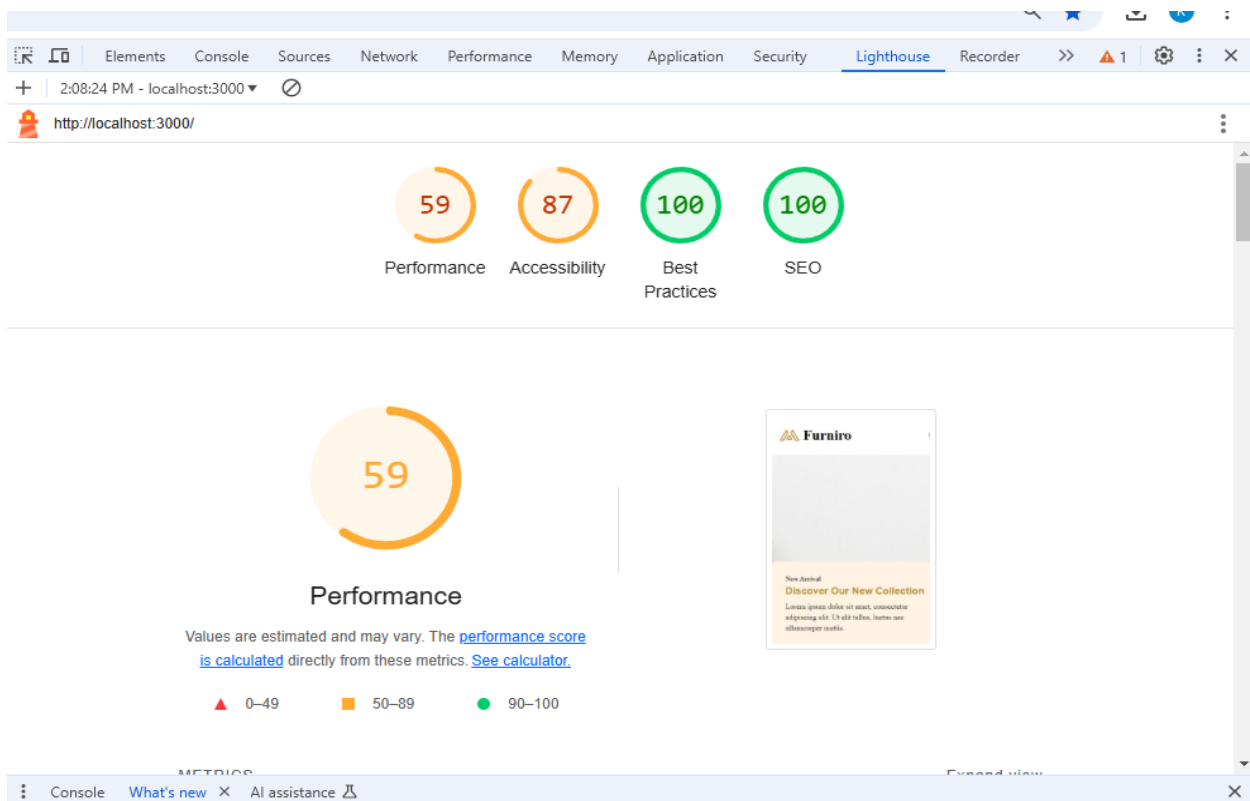
2. Error Handling

I add simple error messages, like "Product not found," to help users understand when something goes wrong.

“Product not found”

3. Performance Testing

I use the Lighthouse tool to analyze my website's speed and performance.



PERFORMANCE: 59

ACCESSIBILITY: 87

BEST PRACTICE: 100

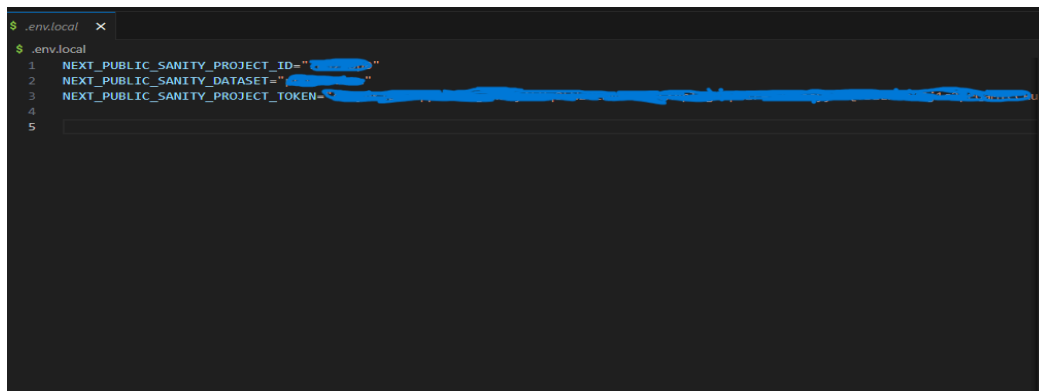
SEO: 100

4. Cross-Browser and Device Testing

I check my website's responsiveness using Chrome and test it across all devices-mobile, tablet, laptop, to ensure it looks great and works smoothly on every screen size.

Step 5: Security Testing

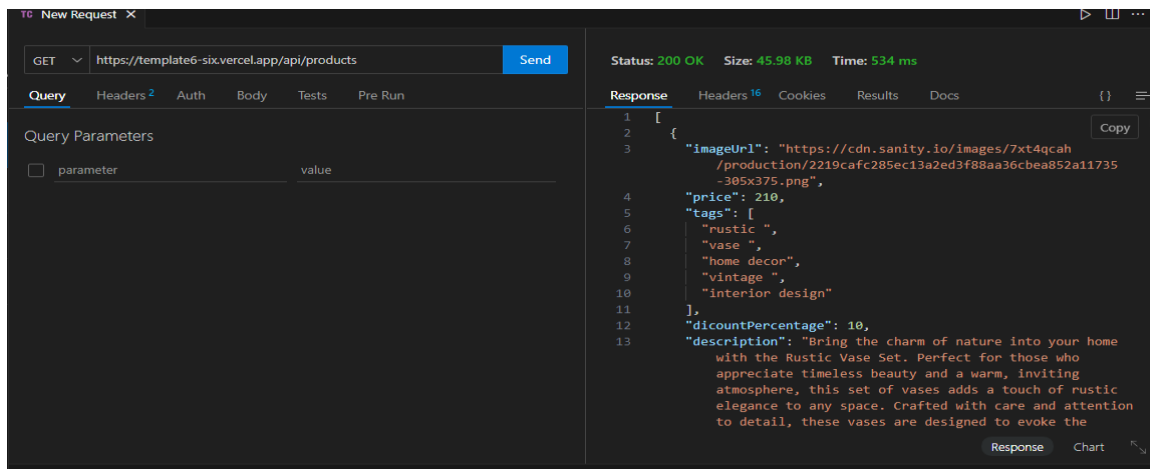
For security, I store all sensitive information, such as API keys and passwords, in the env.local file to keep it safe and secure.



```
$.env.local x
$.env.local
1 NEXT_PUBLIC_SANITY_PROJECT_ID=""
2 NEXT_PUBLIC_SANITY_DATASET=""
3 NEXT_PUBLIC_SANITY_PROJECT_TOKEN=""
4
5
```

Step 5: API Testing and Security

I use Thunder Client to check my APIs, and no issues were detected. They are working perfectly and providing the correct data.



```
T0 New Request x
GET https://template6-six.vercel.app/api/products Send
Query Headers Auth Body Tests Pre Run
Query Parameters
parameter value
Response Headers Cookies Results Docs {}
1 [
2 {
3   "imageUrl": "https://cdn.sanity.io/images/7xt4qcah/production/2219cafc285ec13a2ed3f88aa36cbea852a11735-305x375.png",
4   "price": 210,
5   "tags": [
6     "rustic ",
7     "vase ",
8     "home decor",
9     "vintage ",
10    "interior design"
11  ],
12   "discountPercentage": 10,
13   "description": "Bring the charm of nature into your home with the Rustic Vase Set. Perfect for those who appreciate timeless beauty and a warm, inviting atmosphere, this set of vases adds a touch of rustic elegance to any space. Crafted with care and attention to detail, these vases are designed to evoke the
```

Testing Report

Test Case ID	Test Case Description	Test Steps	Expected Result	Actual Result	Status	Severity Level	Assigned To	Remarks	
1	TC001	Validate product listing page	Open product page > Verify products	Products displayed correctly	Products displayed correctly	Passed	Low	-	Working properly
2	TC002	Test API error handling	Disconnect API > Refresh page	Show fallback UI with error message	Error message shown	Passed	Medium	-	Displayed correctly.
3	TC003	Check cart functionality	Add product to cart > Verify cart contents	Cart updates with added product	Cart updates as expected	Passed	High	-	Works as expected.
4	TC004	Ensure responsiveness on mobile	Resize browser window > Check layout	Layout adjusts properly to screen size	Responsive layout working as intended	Passed	Medium	-	Mobile-friendly.

The testing report shows all key features, including product listing, API error handling, cart functionality, and responsiveness, are working as expected.

Summary

Testing helped improve the website by fixing errors and making it more stable. Features like product display, cart, and APIs are working well. The site is responsive on all devices, and error messages are clearer for users. This testing gives me confidence to keep improving and adding new features.

DAY 6 - DEPLOYMENT PREPARATION AND STAGING ENVIRONMENT SETUP

On Day 6, I'm preparing my marketplace for deployment by setting up a staging environment and hosting on vercel.

Steps for Implementation:

Step 1: Hosting Platform Setup

I am using **Vercel** for quick deployment of my project.

2. Connect Repository

- Link GitHub repository to Vercel.
- Configure build settings and add deployment scripts.

Step 2: Configure Environment Variables

- Create an **.env** file.
- Add sensitive variables like API keys and tokens securely.

Upload Variables to Hosting Platform:

- Upload variables to the hosting platform.
- Use the platform's dashboard to securely add environment variables.

Step 3: Deploy to Staging

Deploy the website and make sure it works correctly without any errors.

A **staging environment** is a test version of the website where you check if everything works properly before launching it live and fix any errors. It's used to make sure the site is ready for real users.

THANK YOU

STUDENT NAME: KRISTINA

SLOT: THURSDAY MORNING