

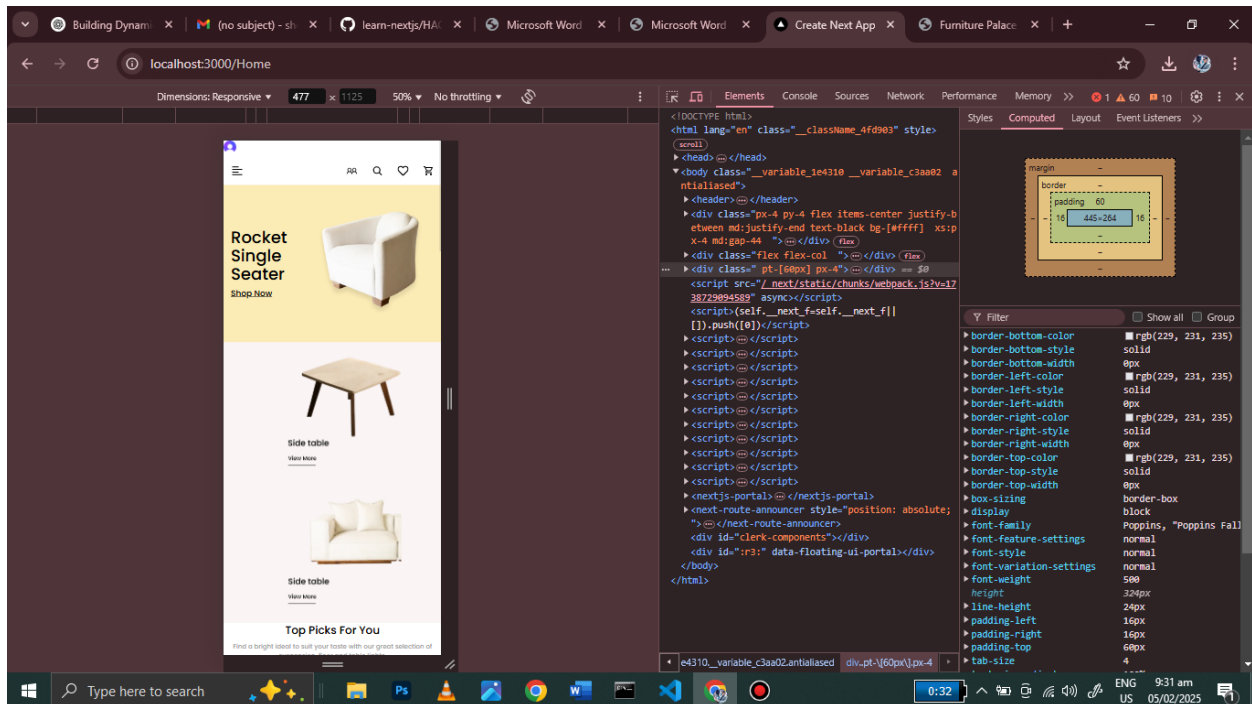
# Furniture Palace

Day 05

Responsive test:

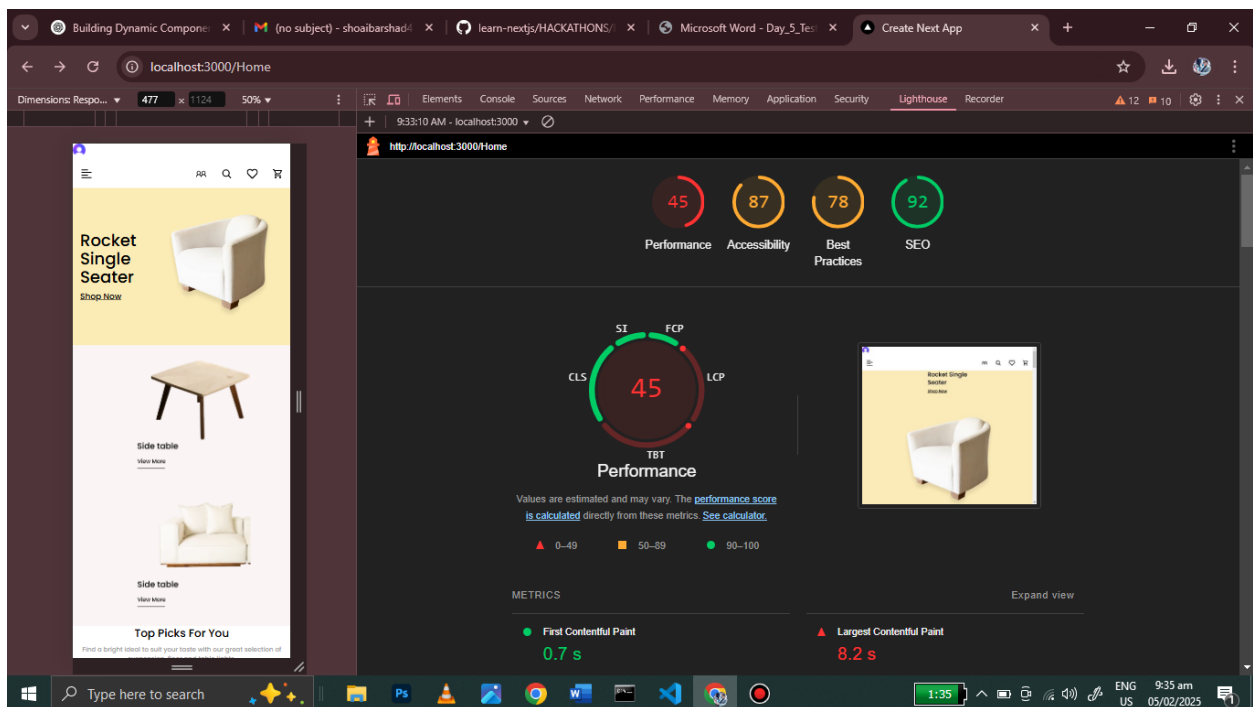
use tailwind css for responsive and styling, fast and smooth approach.

Tailwind CSS provides a **mobile-first** approach to responsive design, making it easy to create layouts that adapt to different screen sizes. It uses **responsive prefixes** to apply styles at specific breakpoints.



## Lighthouse Testing Report

**Lighthouse** is an open-source tool by Google for testing web page performance, accessibility, SEO, and best practices. Below is a structured report based on a **Lighthouse audit** performed on a Next.js marketplace.



## # Lighthouse Testing Report

**Tested URL:** `http://localhost:3000/Home`

**Test Date:** February 5, 2025

**Testing Tool:** Google Lighthouse (Chrome DevTools)

**Device:** Mobile (Responsive Mode - 477x1124)

---

## 1. Lighthouse Scores

Category	Score (Out of 100)
----------	--------------------

Performance	45
-------------	----

Accessibility	87
---------------	----

Best Practices	78
----------------	----

SEO	92
-----	----

## Error Handling:

This report documents the error handling strategies implemented in the Next.js project, including methods to catch and manage errors efficiently to improve application stability and user experience.

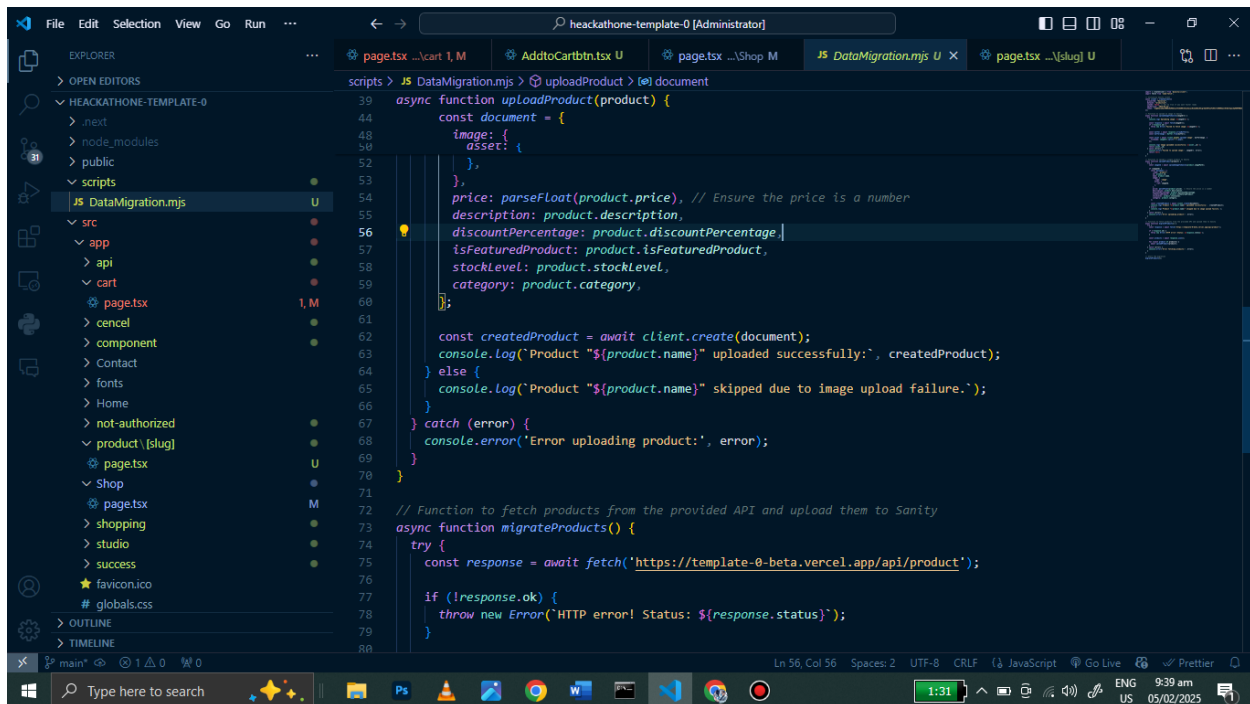
---

## 2. Identified Errors & Handling Strategies

### 2.1. API Errors (Sanity API & External APIs)

**Issue:** API calls failing due to network errors, incorrect responses, or server downtime.

**Fix:** Implemented try...catch blocks with proper error messages and fallback UI.



```
39 async function uploadProduct(product) {
40   const document = {
41     image: {
42       asset: {
43         url: product.image,
44       },
45     },
46     price: parseFloat(product.price), // Ensure the price is a number
47     description: product.description,
48     discountPercentage: product.discountPercentage,
49     isFeaturedProduct: product.isFeaturedProduct,
50     stockLevel: product.stockLevel,
51     category: product.category,
52   };
53
54   const createdProduct = await client.create(document);
55   console.log("Product "${product.name}" uploaded successfully:", createdProduct);
56 } else {
57   console.log("Product "${product.name}" skipped due to image upload failure.");
58 }
59 } catch (error) {
60   console.error("Error uploading product:", error);
61 }
62
63 // Function to fetch products from the provided API and upload them to Sanity
64 async function migrateProducts() {
65   try {
66     const response = await fetch('https://template-0-beta.vercel.app/api/product');
67
68     if (!response.ok) {
69       throw new Error(`HTTP error! Status: ${response.status}`);
70     }
71   } catch (error) {
72     console.error("Error fetching products:", error);
73   }
74 }
75
76 # globals.css
77
78
79
80
```