**Day 5 - Testing and Backend Refinement - Furniro**

**Introduction:**

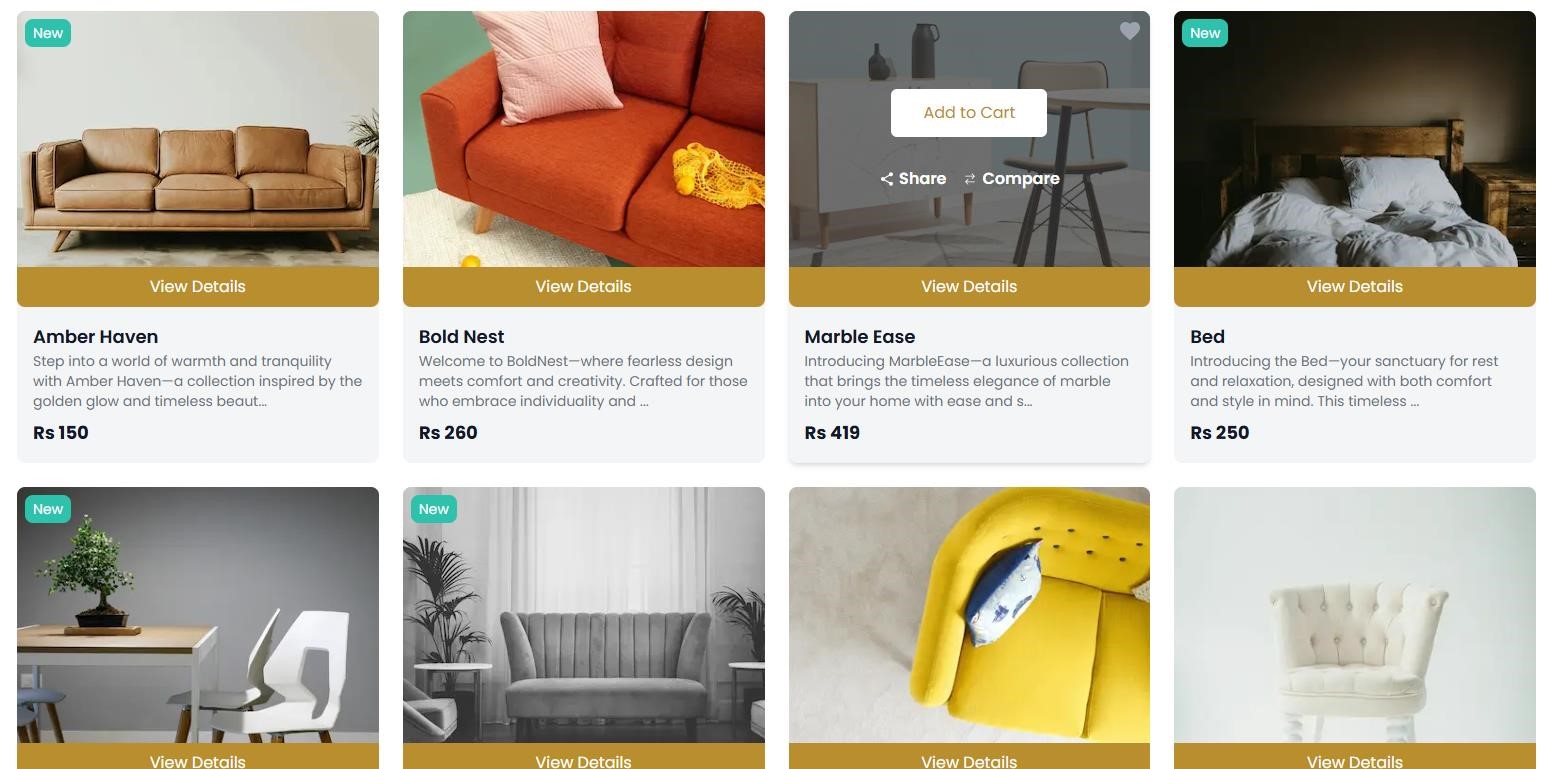
Day 5 of our hackathon focuses on testing, error handling, and backend integration refinement for the **Furniro** marketplace. In this phase, we aim to ensure that the marketplace functions seamlessly by validating core features, handling errors gracefully, and optimizing the platform for real-world deployment.

The tasks we completed during this phase include:

1. **Functional Testing**: Verifying that all core features like product listing, search, and cart operations work as expected.
2. **Error Handling**: Implementing try-catch blocks and fallback UI elements to handle errors and provide clear user feedback.
3. **Performance Testing**: Analyzing and optimizing the website's performance to ensure quick load times and smooth interactions.
4. **Cross-Browser and Device Testing**: Ensuring the marketplace works consistently across different browsers and devices.
5. **Security Testing**: Securing sensitive data and protecting the platform against common vulnerabilities.
6. **User Acceptance Testing (UAT)**: Simulating real-world user interactions to validate the marketplace’s usability.
7. **Functional Testing:**

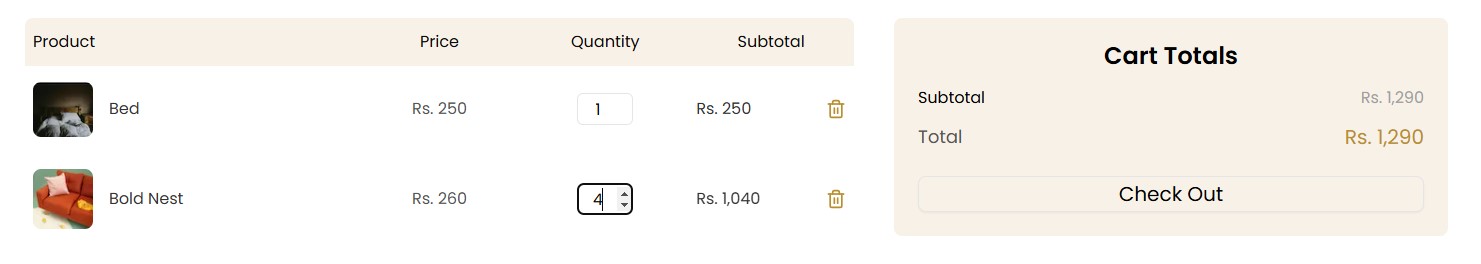
**Product Listing:**

* + **Test Case**: Ensure that the product listing page displays products correctly from the API.
  + **Steps**:
    1. Navigate to the product listing page.
    2. Verify that the product grid is populated with products fetched from the API.
    3. Ensure that each product displays the correct image, title, and price.
  + **Expected Result**: Products should be displayed in a grid layout with their respective images, titles, and prices.
  + **Actual Result**: Products were correctly displayed with the right information.
  + **Screenshot**:



**Cart Operations:**

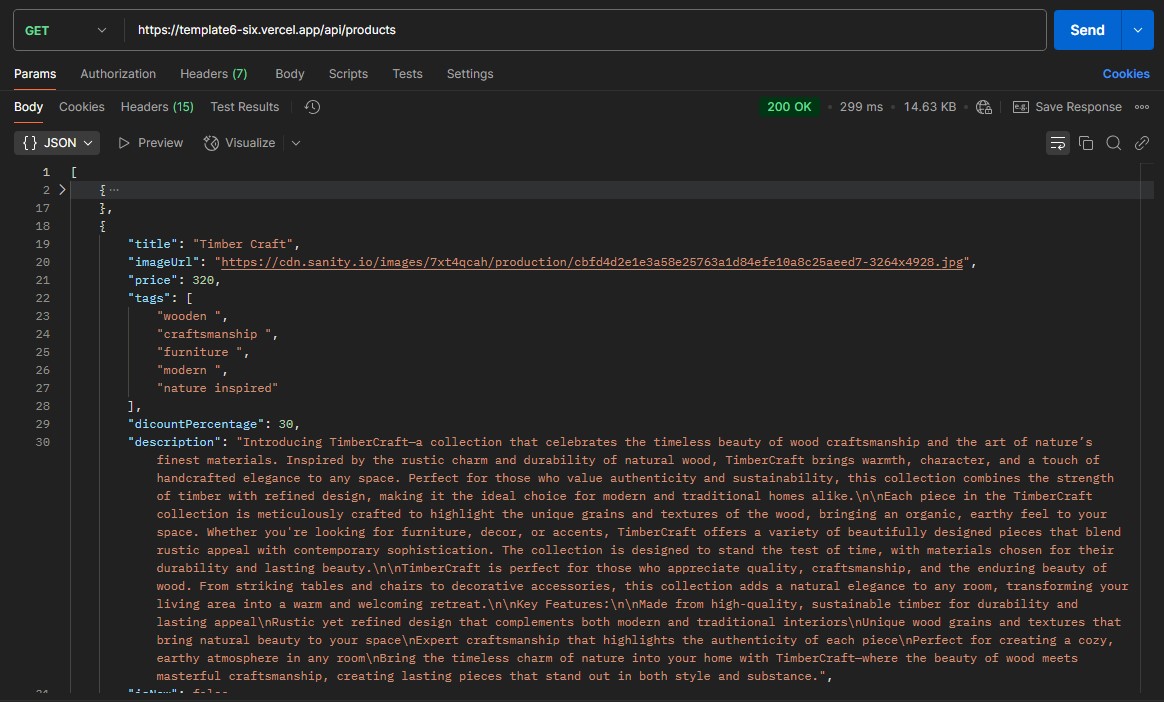
* + **Test Case**: Validate the cart's ability to add, update, and remove items.
  + **Steps**:
    1. Add products to the cart.
    2. Update the quantity of items in the cart.
    3. Remove items from the cart.
  + **Expected Result**: Items are correctly added, updated, and removed from the cart.
  + **Actual Result**: Cart functionality works correctly. **Screenshot:**





**API Testing with Postman:**

* + **Test Case**: Test API responses for fetching product data.
  + **Steps**:
    1. Use **Postman** to send a GET request to the product API endpoint.
    2. Verify that the correct data is returned with status 200.
  + **Expected Result**: API returns a successful response with the correct product data.
  + **Actual Result**: API response as expected.
  + **Screenshot**: [Insert screenshot here]

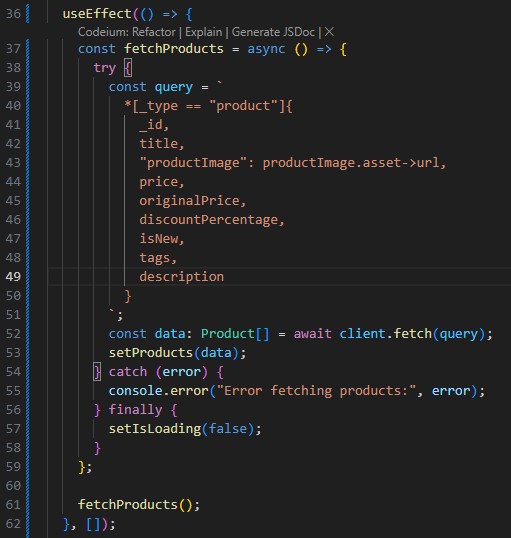


1. **Error Handling:** 
   * **Try-Catch Example**:

In the following code, we use a **try-catch** block inside a useEffect to fetch product data from the API. If an error occurs during the data fetch, it’s caught and logged, ensuring the application doesn’t crash.

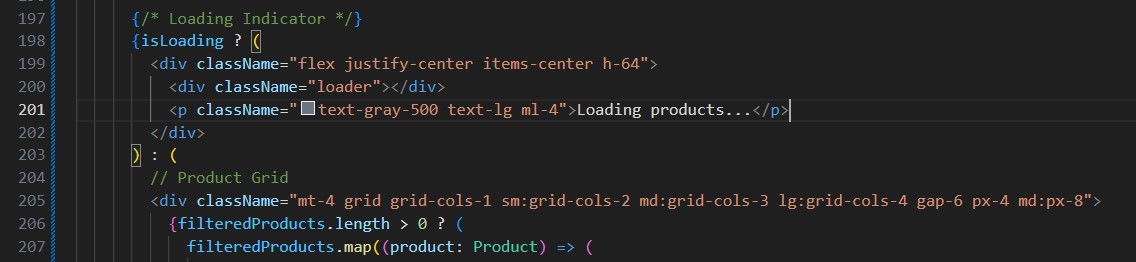
**Explanation**:

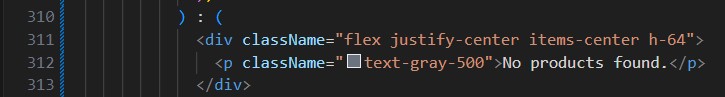
* + The **try** block attempts to fetch the product data using a query to **Sanity CMS**.
  + The **catch** block logs any error (e.g., network failure, API issue) and ensures the application continues running smoothly without crashing.
  + The **finally** block ensures that the loading state (setIsLoading(false)) is always reset, whether the fetch is successful or not.



**Fallback UI Example:**

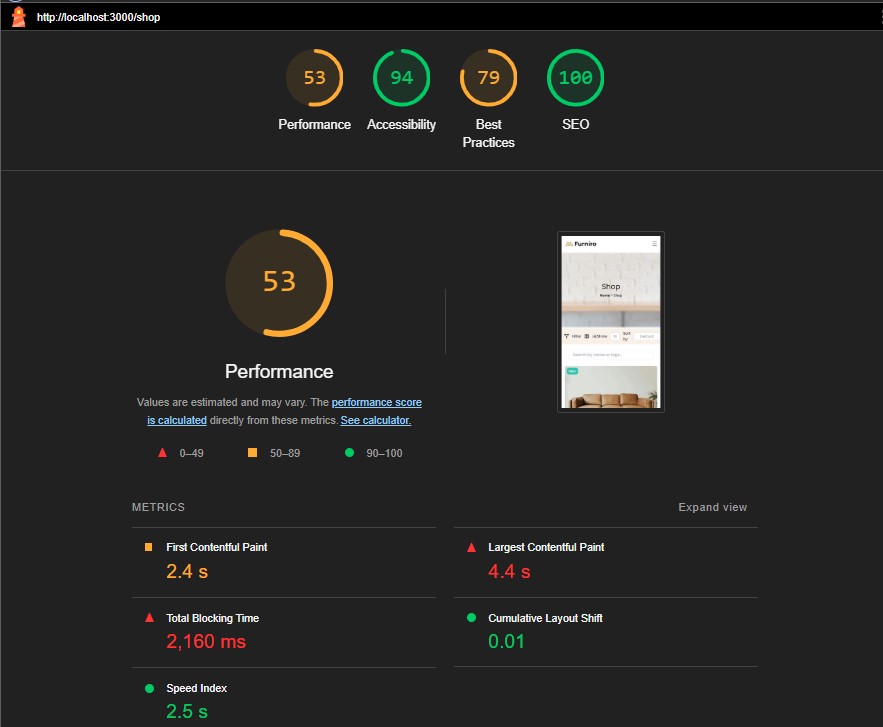
When products are unavailable or there’s an error fetching them, we show an alternative message such as "No items found" or a loading spinner.

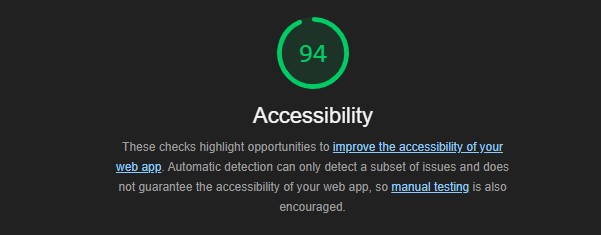




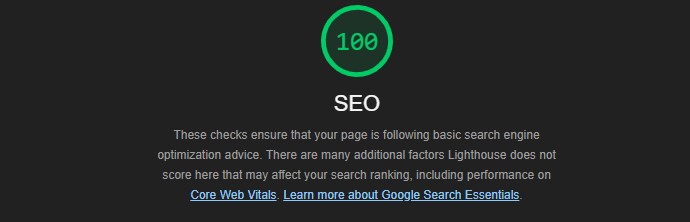
**3.Performance Testing:**

Description: The performance of the website was tested using tools like Lighthouse.









1. **Cross-Browser and Device Testing:**

**Description:**

We tested the marketplace across various browsers and devices to ensure consistent performance and responsiveness.

**Key Points:**

* + Tested on popular browsers: Chrome, Firefox, Safari, and Edge.
  + Verified responsiveness on desktop, tablet, and mobile devices.
  + Ensured no layout issues or broken features across different screen sizes.

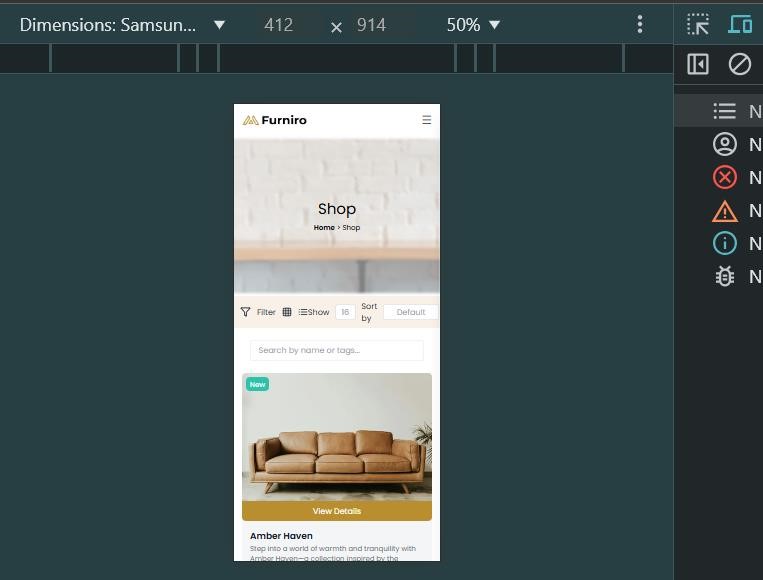
**Expected Result:**

The marketplace should function seamlessly across all browsers and devices.

**Actual Result:**

The marketplace displayed correctly with no layout or functionality issues on all tested browsers and devices.

**Screenshot:**



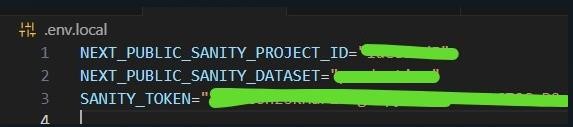
1. **Security Testing:**

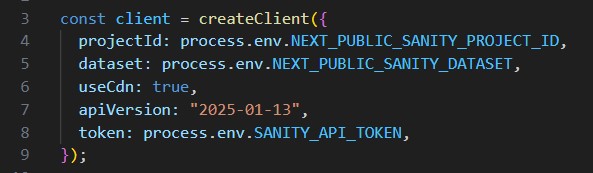
**Description:**

We focused on securing the marketplace by validating inputs, ensuring secure communication, and protecting sensitive data.

**Secure API Communication:**

* + Ensure API calls are made over HTTPS to encrypt data during transmission.
  + Store sensitive data like API keys in environment variables to prevent exposure.





1. **User Acceptance Testing (UAT):**

We tested the marketplace to ensure it meets real-world usage expectations.

**Key Points:**

* + Simulated tasks like browsing, adding to cart, and checking out.
  + Collected feedback from peers and mentors to improve usability.

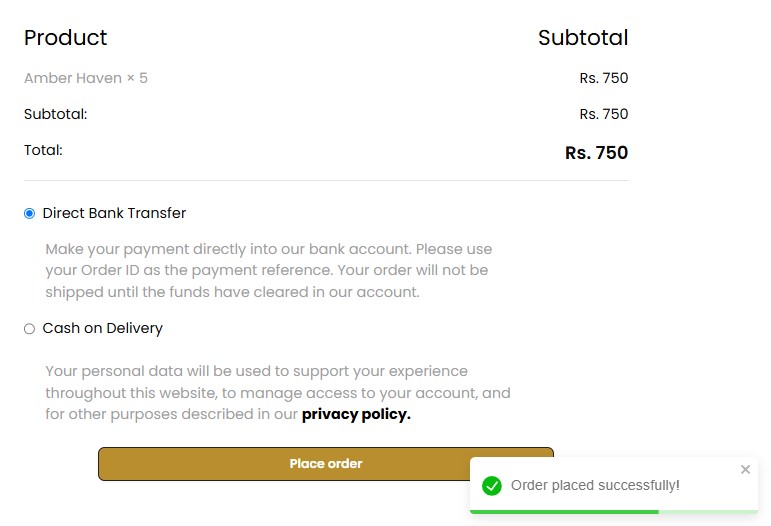
**Expected Result:**

A seamless and user-friendly experience.

**Actual Result:**

UAT completed successfully, with no major issues.

**Screenshot:**



**Final Checklist:**

**Task Status**

Functional Testing ✔

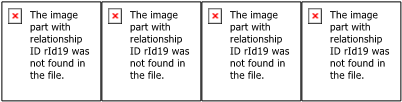
Error Handling ✔

Performance Testing ✔

Device Testing ✔

Security Testing ✔

Documentation ✔



-Made by Rimmat Khalid