|  |
| --- |
| **Day 5: Testing, Error Handling, and Backend** |
| **Integration Refinement** |

# Achievements:

1. **Shipment API Integration:** Successfully integrated the **shipment API** to allow users to view real-time shipment data.
   * Ensured smooth API calls and proper error handling to provide a seamless user experience.
   * Displayed shipment tracking details dynamically, allowing users to check shipment progress effortlessly.

## Dynamic Routing:

* + Implemented dynamic routing for the product detail pages.
  + Users can now click on a product and navigate to a detailed view, displaying all relevant information such as: o Product name o Description o Price
    - Images
    - Availability status

## Sorting Functionality:

* + Added sorting buttons for various categories:
    - **Category**: Sort by product categories (e.g., furniture type).
    - **Material**: Sort based on the product material (e.g., wood, metal).
    - **Color**: Sort by available color options.
  + Sorting functionality allows users to filter products according to their preferences.

## Search Bar Functionality:

* + Implemented a responsive search bar to enable users to search for products by: o Name
    - Description
    - Tags
  + Search results update dynamically to match the input, providing a user-friendly experience.

## Improved Responsiveness:

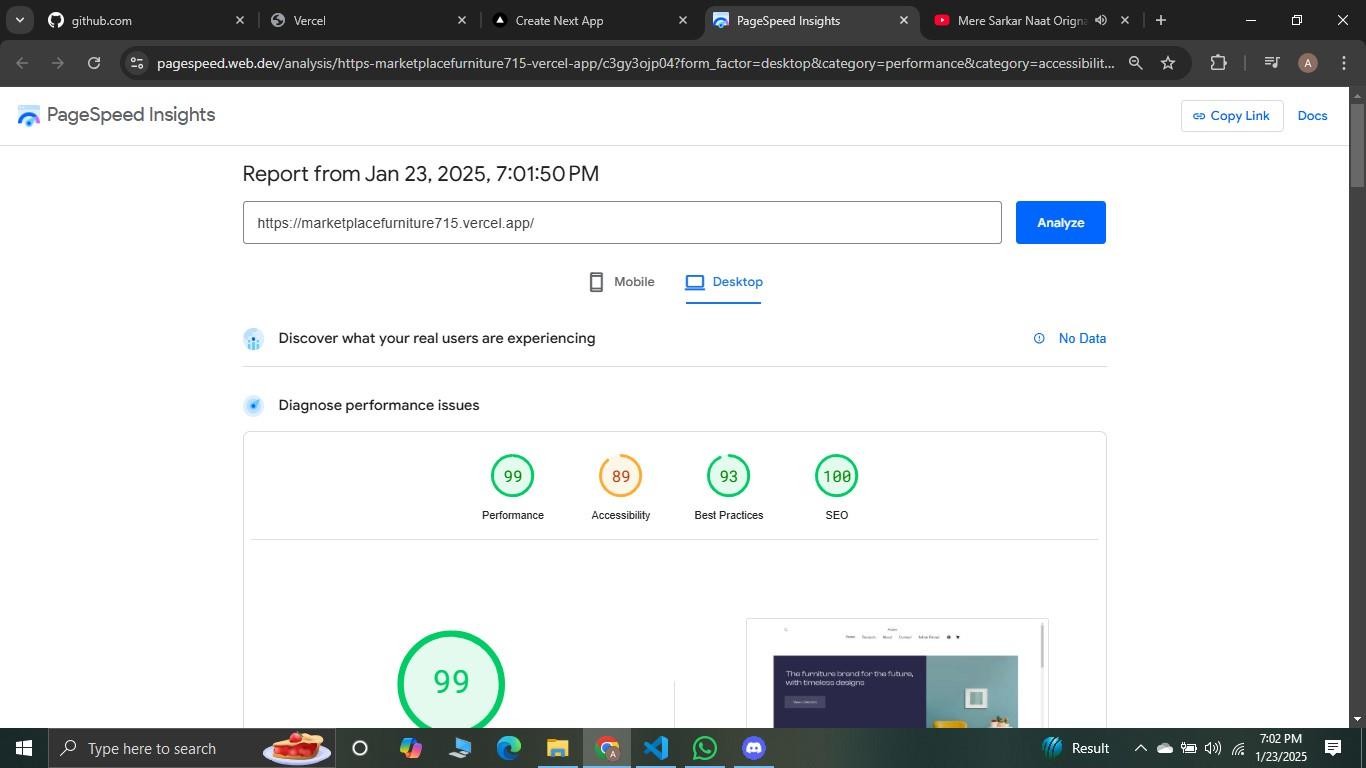
* + Enhanced responsiveness across the application to ensure compatibility with:
    - Desktop o Tablet o Mobile devices
  + All pages adapt seamlessly to different screen sizes without compromising design quality.

## Performance Optimization:

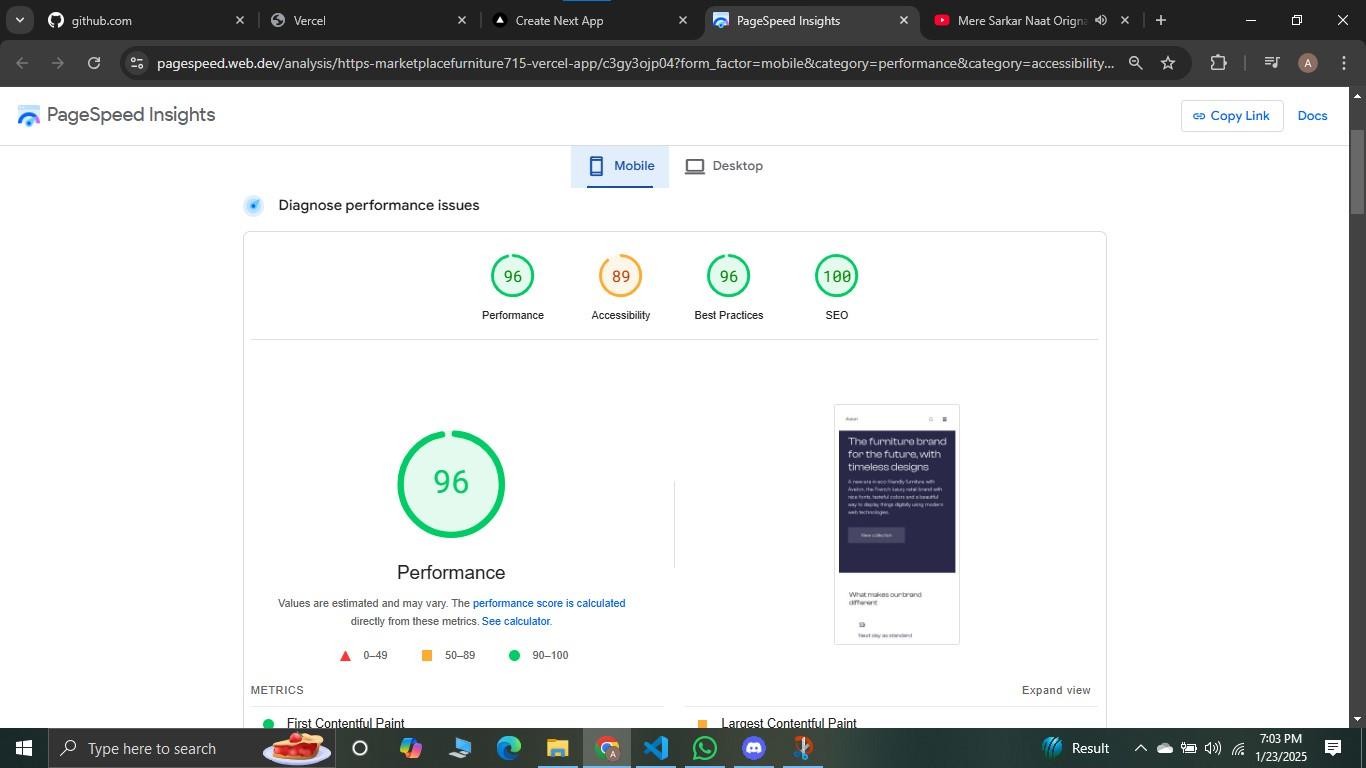
* + Conducted performance analysis using **Google Lighthouse**.
  + Achieved the following results:
    - **Performance**: 99 (Desktop), 96 (Mobile) o **Accessibility**: 89
    - **Best Practices**: 93 (Desktop), 96 (Mobile) o **SEO**: 100

## Lighthouse Report Screenshots:

* + **Desktop Report:**

****

* + **Mobile Report:**



## Error Handling:

* + Added robust error handling for API calls to ensure the application gracefully manages:
    - API failures o Network issues o Invalid data
  + Displayed user-friendly error messages when necessary.
  + Implemented a fallback UI for scenarios where data cannot be fetched, such as:
    - **Loading State**: Displays a loading indicator while data is being fetched.
    - **Error State**: Shows a fallback message like "No products found" or "Failed to load data" if the API call fails or no products are available.

## UI Refinements:

* + Polished the overall UI to align with the project’s Pinterest-inspired theme.
  + Used consistent color schemes, typography, and spacing to enhance visual appeal.

## CVS Documentation:

The following table summarizes the test cases executed during this testing cycle:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case Description** | **Test Steps** | **Expected Result** | **Actual Result** | **Status** | **Sev Le** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TC001 | Validate product listing page | Open product page > Verify products | Products displayed correctly | Products displayed correctly | Passed | Low |
|  | Test API error handling | Disconnect API  > Refresh page | Show fallback UI with error message | Error message shown |  |  |
| TC002 |  | Passed | Me |
| TC003 | Check cart functionality | Add product to cart > Verify cart contents | Cart updates with added product | Cart updates as expected | Passed | Hig |
| TC004 | Ensure responsiveness on mobile | Resize browser window  >  Check layout | Layout adjusts properly to screen size | Responsive layout working as intended | Passed | Me |
| TC005 | Test dynamic routing functionality | Navigate to dynamic product page via URL | Correct product details displayed based on URL | Product details displayed correctly based on URL  structure | Passed | Hig |

* **Testing Tools Used:**

* + **React Testing Library**: For unit testing React components. We verified the functionality and rendering of individual components to ensure their behavior aligns with the expected outcomes.
  + **Cypress**: For end-to-end testing. We simulated real user interactions to test the overall flow and integration of components within the application.

* **Areas Tested:**

* + **Dynamic Routing**: Verified the correctness of routes and ensured that the application correctly displays dynamic product pages based on the URL structure. Dynamic routing functionality is working as expected.
  + **Cart Functionality**: Ensured the cart updates dynamically when products are added, removed, or updated. The add-to-cart feature is functioning correctly.
  + **Search and Filter Functionality**: Tested the search bar and filter buttons to confirm they retrieve and display relevant products efficiently. Both features are working excellently.
  + **Shipment API**: Validated the API integration by simulating different responses (e.g., successful, failed, and delayed) and ensured the fallback UI handles errors gracefully. API fetches are handled well using try-catch blocks, with appropriate error messages such as "No products found" or "Loading..." displayed when needed.
  + **Responsiveness**: Checked responsiveness on different screen sizes and devices to guarantee a consistent user experience.
  + **Product Details Page**: Confirmed all product details are displayed correctly, including images, descriptions, and pricing. Product details are shown accurately and align with the expected design.

# Summary:

Day 5 involved significant improvements in both functionality and user experience. The integration of the shipment API, dynamic routing, and search/sorting features ensures the application meets user needs effectively. Additionally, rigorous testing and responsiveness optimizations guarantee the app performs well across all devices.