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

Key Test Cases (with error message handling)

1. Validate Product Listing Page.
2. Validate Dynamic Product Detail Page.
3. Validate Add to Cart Page Product Addition.
4. Validate Search Bar Result Page.
5. Validate Wishlist Page Product Addition.
6. Validate Comparison Page Product Addition.

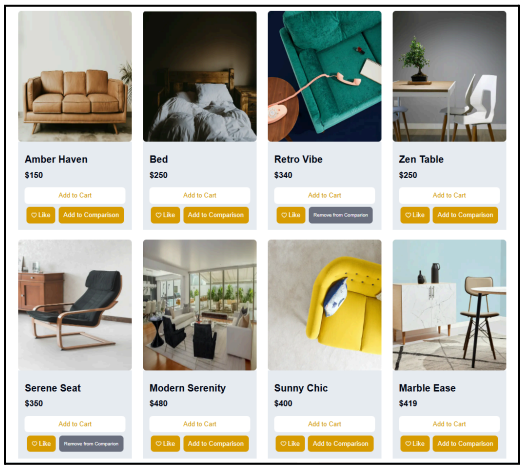
1.Validate Product Listing Page.

- **Description:** Ensure the product listing page loads all products correctly. If no product is available, an error message should display.
- **Test Case Code Snippet:**

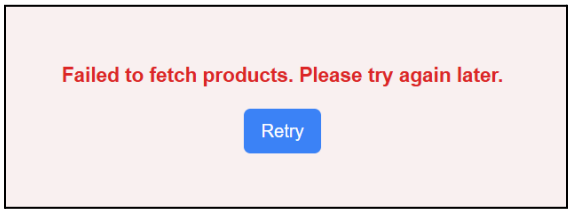
```
useEffect(() => {
  async function fetchProductData() {
    try {
      setIsLoading(true);
      setErrorMessage("");

      const productFetchData: Product[] = await client.fetch(allProduct);
      if (productFetchData.length === 0) {
        setErrorMessage("No products found.");
      } else {
        setProducts(productFetchData);
      }
    } catch (error: any) {
      if (error.name === "TypeError") {
        setErrorMessage(
          "Network error. Please check your internet connection."
        );
      } else {
        setErrorMessage("Failed to fetch products. Please try again later.");
      }
    } finally {
      setIsLoading(false);
    }
  }

  fetchProductData();
}, []);
```

```
if (errorMessage) {
  return (
    <div className="flex justify-center items-center min-h-screen bg-[#f9f0f0]">
      <div className="text-center">
        <p className="text-red-600 text-lg font-semibold">{errorMessage}</p>
        <button
          className="mt-4 px-4 py-2 bg-blue-500 text-white rounded-md hover:bg-blue-600"
          onClick={() => window.location.reload()}
        >
          Retry
        </button>
      </div>
    </div>
  );
}
```

- **Output:**



No error display correctly



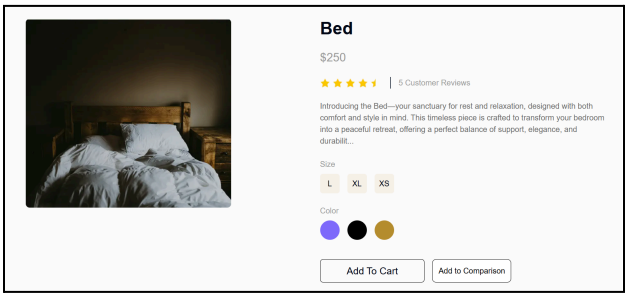
Disconnect Internet Show Error

2.Validate Dynamic Product Detail Page.

- **Description:** Verify the product detail page dynamically displays product information. If the product ID is invalid, show an error message.
- **Test CaseCode Snippet:**

```
useEffect(() => {
  const fetchProductData = async () => {
    try {
      const resolvedParams = await props.params;
      const id = resolvedParams.id;
      if (!id) {
        setErrorMessage("Product ID is missing.");
        return;
      }
      const productFetchData: Product[] = await client.fetch(
        groq`*[_type == "product" && _id == $id]{
          _id,
          title,
          isNew,
          description,
          discountPercentage,
          price,
          productImage,
          tags
        }`,
        { id }
      );
      if (productFetchData.length > 0) {
        setProduct(productFetchData[0]);
      } else {
        setErrorMessage("No product found for the given ID.");
      }
    } catch (error: any) {
      // Check if it's a network-related error
      if (error.message && error.message.includes("NetworkError")) {
        setErrorMessage(
          "Network error. Please check your internet connection."
        );
      } else {
        setErrorMessage(
          "Error fetching product data. Please try again later."
        );
      }
    }
  };

  fetchProductData();
}, [props.params]); // Trigger the effect when params change
```

- **Output:**



Product display correct

Disconnect Query

3.Validate Add to Cart Page Product Addition.

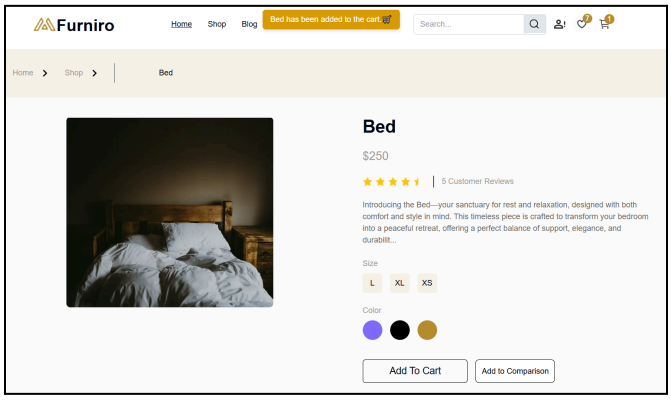
- **Description:** Check if a product is added to the cart properly. Ensure a notification is displayed, and the product is correctly added to the cart and Display Correctly.
- **Test Case Code Snippet:**

```
// Notification Handler
const triggerNotification = (message: string) => {
  setNotification(message);
  setTimeout(() => setNotification(null), 3000); // Hide after 3 seconds
};

// Add to Cart Handler
const handleAddToCart = (product: Product) => {
  dispatch(addToCart(product));
  triggerNotification(`${product.title} has been added to the cart.🛒`);
};
```

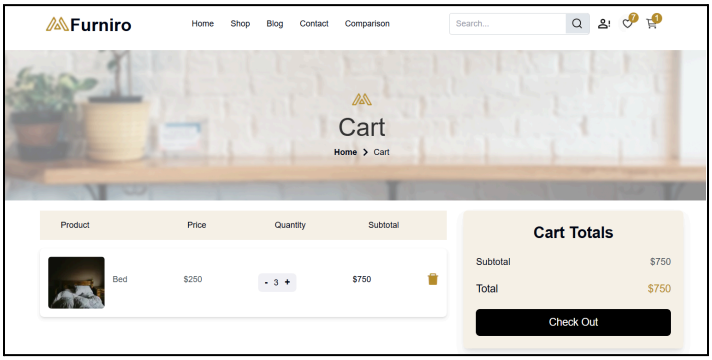
Notification & Add to cart button Handler

- **Output:**

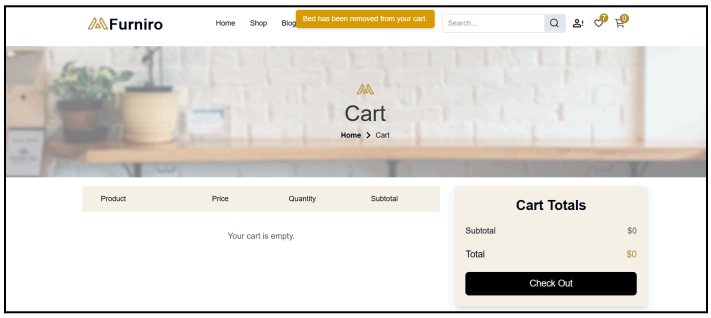
Add product successfully



The product is displayed correctly on the Cart page



Remove product successfully

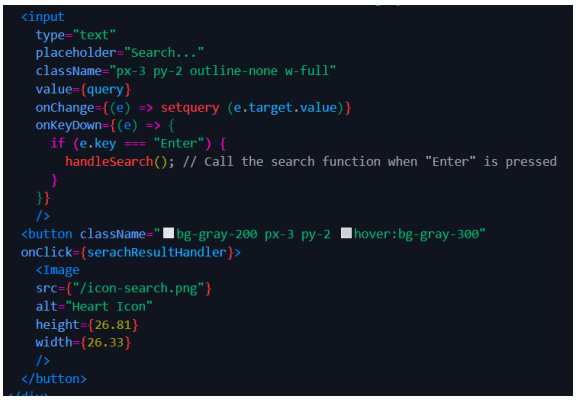


4.Validate Search Bar Result Page

- **Description:** Test if the search bar retrieves results accurately. If no results are found, show a message.
- **Test Case Code Snippet::**

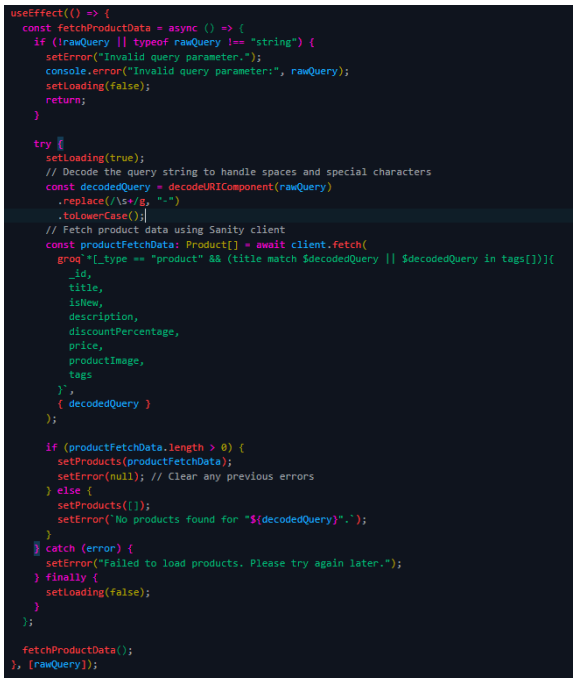
Header code for search bar

```
const [query , setquery] = useState<string>("")


const router = useRouter();
const serachResultHandler = () =>{
  router.push(`/search/${query}`)
}


const handleSearch = () => {
  if (query.trim() === "") {
    console.error("Search query is empty");
    return;
  }
};
```



```
<input
  type="text"
  placeholder="Search..."
  className="px-3 py-2 outline-none w-full"
  value={query}
  onChange={(e) => setquery (e.target.value)}
  onKeyDown={(e) => {
    if (e.key === "Enter") {
      handleSearch(); // Call the search function when "Enter" is pressed
    }
  }}
  />
<button className="bg-gray-200 px-3 py-2 hover:bg-gray-300"
onClick={serachResultHandler}>
  <Image
  src={"/icon-search.png"}
  alt="Heart Icon"
  height={26.81}
  width={26.33}
  />
</button>
```

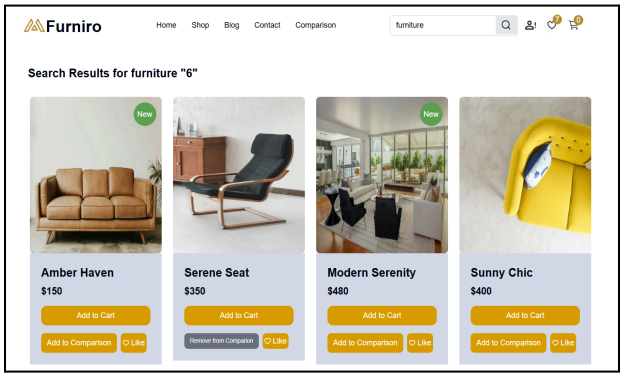
Fetching search result



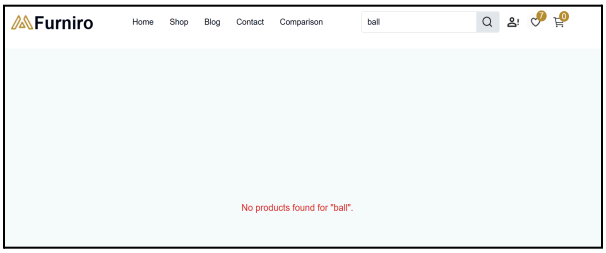
```
useEffect(() => {
  const fetchProductData = async () => {
    if (!rawQuery || typeof rawQuery !== "string") {
      setError("Invalid query parameter.");
      console.error("Invalid query parameter:", rawQuery);
      setLoading(false);
      return;
    }

    try {
      setLoading(true);
      // Decode the query string to handle spaces and special characters
      const decodedQuery = decodeURIComponent(rawQuery)
        .replace(/\s+/g, "-")
        .toLowerCase();
      // Fetch product data using Sanity client
      const productFetchData: Product[] = await client.fetch(
        groq`*[_type == "product" && (title match $decodedQuery || $decodedQuery in tags[])]{
          _id,
          title,
          isNew,
          description,
          discountPercentage,
          price,
          productImage,
          tags
        }`,
        { decodedQuery }
      );

      if (productFetchData.length > 0) {
        setProducts(productFetchData);
        setError(null); // Clear any previous errors
      } else {
        setProducts([]);
        setError(`No products found for "${decodedQuery}".`);
      }
    } catch (error) {
      setError("Failed to load products. Please try again later.");
    } finally {
      setLoading(false);
    }
  };

  fetchProductData();
}, [rawQuery]);
```

- **Output:**

Product display correctly



search product which has not in sanity



5.Validate Wishlist Page Product Addition.

Description: Ensure products are added to the wishlist correctly, display a notification when a product is added, and verify that the product is displayed properly on the Wishlist page.

Test Case Code Snippet:

Handle add to wishlist & notification

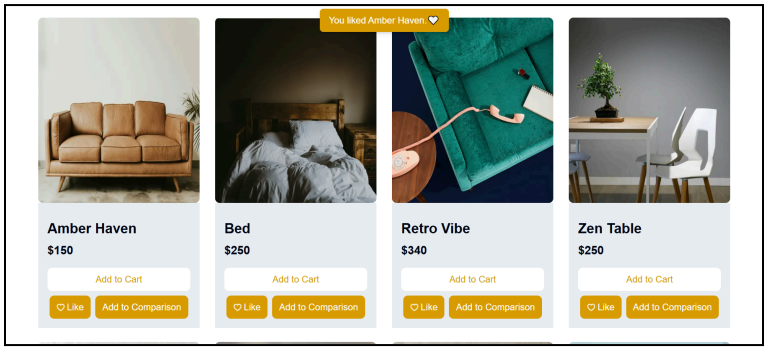
```
// Wishlist Product Handler
const handleLikeProduct = (product: Product) => {
  dispatch(likeProduct(product));
  triggerNotification(`You liked ${product.title}.❤`);
};
```

Display Wishlist Product

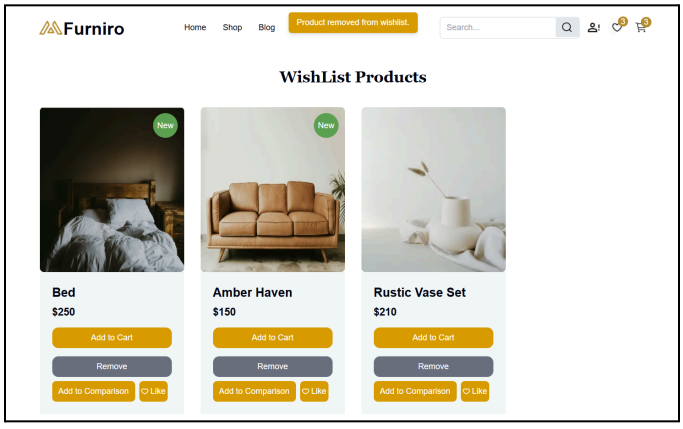
```jsx
{/* No Liked Products */}
{like.length === 0 ? (
  <div className="text-center text-gray-500">
    <p>No liked products yet!</p>
    <Link href="/shop">
      <p className="text-blue-500 underline">Browse products</p>
    </Link>
  </div>
) : (
  <div className="grid max-w-[1220px] mx-auto grid-cols-1 sm:grid-cols-2
    {like.map((element: Product) => (
      <div
        className="relative w-[250px] xs:w-[280px] mx-auto group"
        key={element._id}
      >
        {/* Image Section with Link */}
        <Link href={`/shop/${element._id}`}>···
        </Link>

        {/* Product Info Section */}
        <div className="bg-[#F4F5F7] p-4 sm:p-6">···
        </div>
      </div>
    ))}
  </div>
)}
```

Output:

Show Notification on Adding Product to Like Page



Product display correctly also remove smoothly



6.Validate Comparison Page Product Addition.

Description: Check if products are added to the comparison list. If the product is already in the comparison list, display a message.

Test Case Code Snippet:

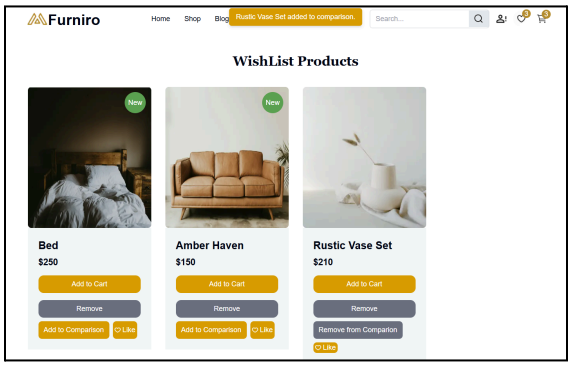
Add to comparison all function & notification handler

```
const handleAddToComparison = (product: Product) => {
  if (comparison.length >= 2) {
    triggerNotification("You can only compare up to 2 products."); // Show a warning
  } else if (comparison.some((item) => item._id === product._id)) {
    triggerNotification(`${product.title} is already in comparison.`);
  } else {
    dispatch(addToComparison(product));
    triggerNotification(`${product.title} added to comparison.`);
  }
};

const handleRemoveFromComparison = (productId: string) => {
  // Remove product from the comparison array
  dispatch(removeFromComparison(productId));
};
```

Handle Add and Remove Product on Comparison Page

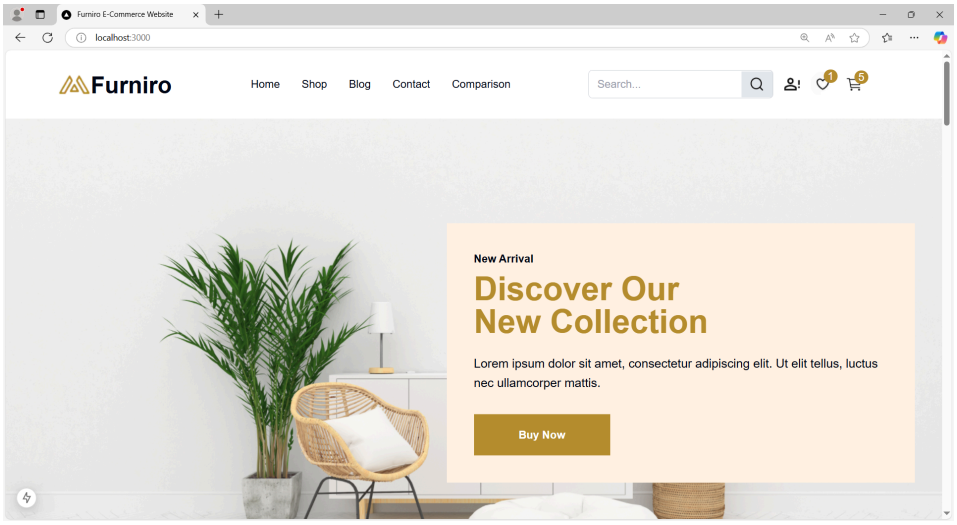
```
{/* Compare */}
<button
  onClick={() => {
    comparison.some((item) => item._id === element._id)
      ? handleRemoveFromComparison(element._id) // Call the remove function if already in comparison
      : handleAddToComparison(element); // Call the add function if not in comparison
  }}
  className={`py-2 px-3 rounded-lg ${
    comparison.some((item) => item._id === element._id)
      ? " bg-gray-500  text-white " // Styling for "Remove from Comparison"
      : " bg-[#D89E00]  text-white" // Styling for "Add to Comparison"
  }`}
>
  {comparison.some((item) => item._id === element._id)
    ? "Remove from Comparion"
    : "Add to Comparison"}{" "}
</button>
```

Output:

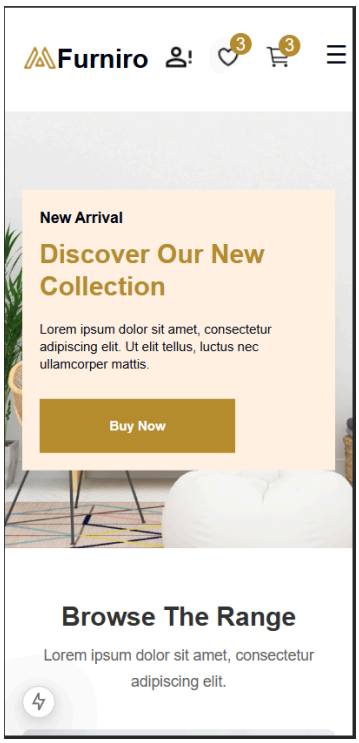


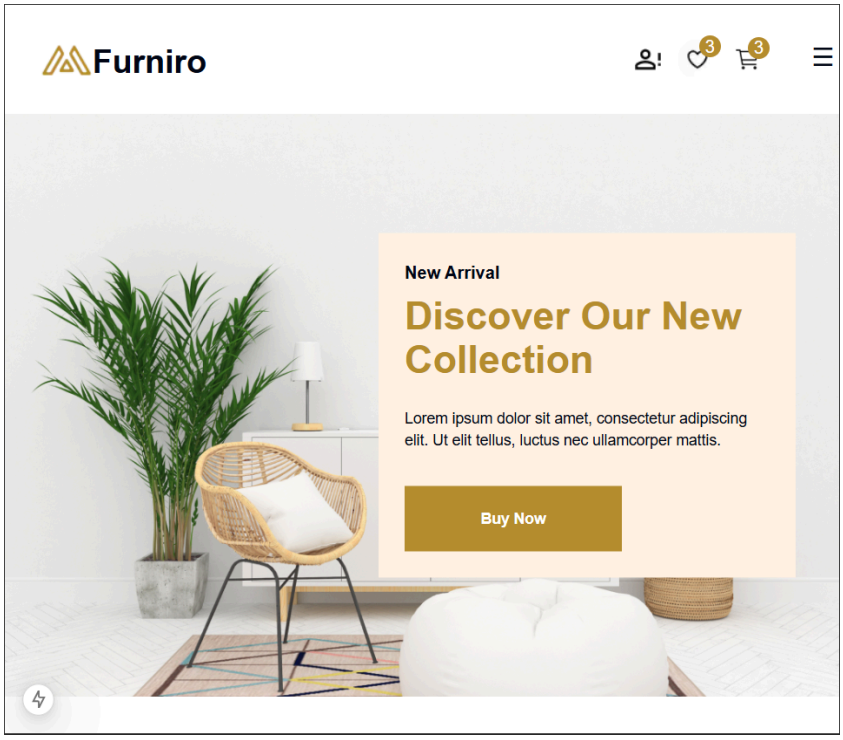
Cross Browser Checking & Responsive

Check Website on Edge Browser



Check Responsiveness on Mobile & Tablet





Website Perfomance