Day 5: Testing and Backend Refinement

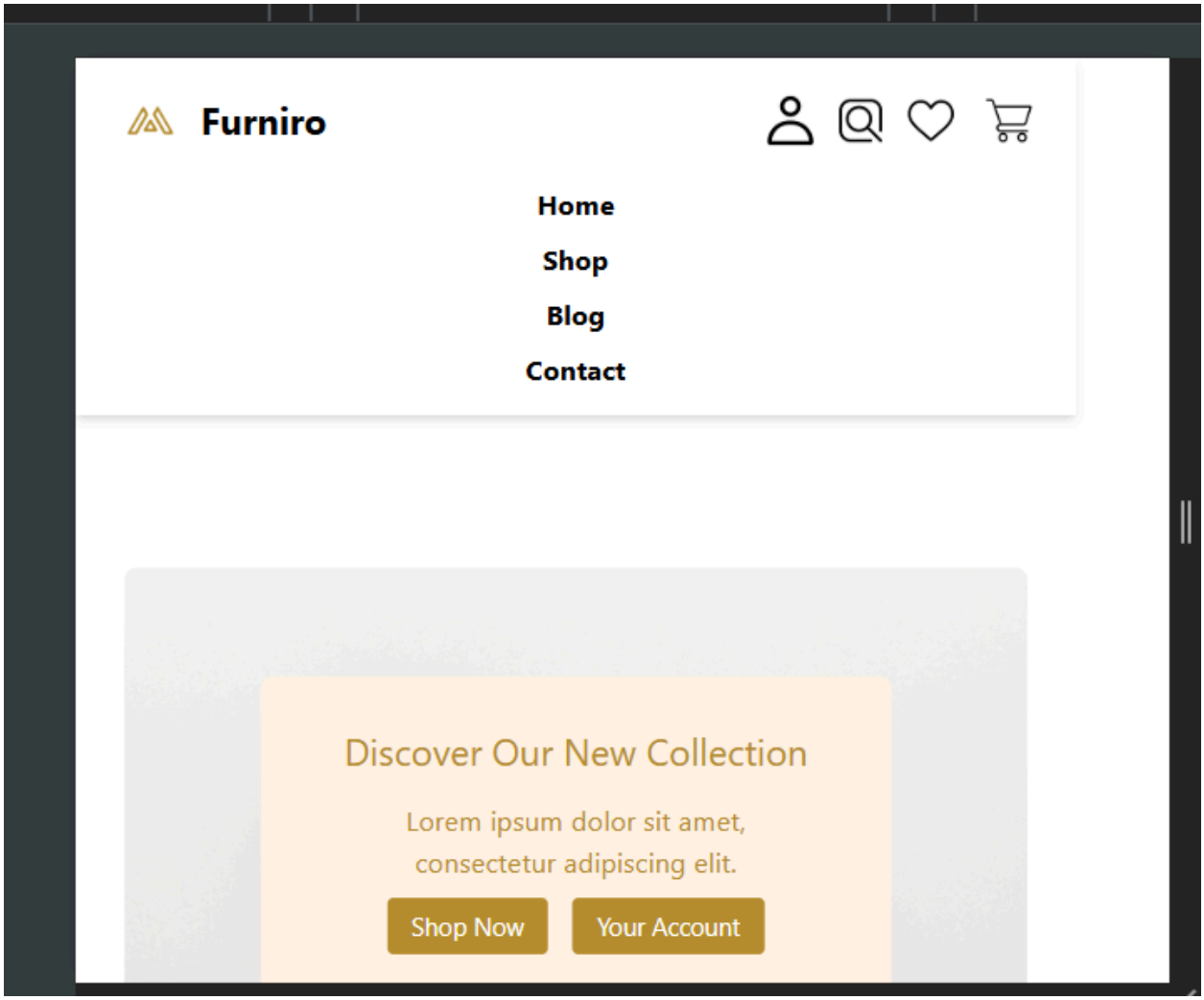
[General E-commerce Furniture Marketplace]

---

1. Responsive Testing

The website was tested for responsiveness across various environments:

- **Chrome** on a desktop
- **Large-size laptops**
- **Android devices**
- **Apple devices**



Results:

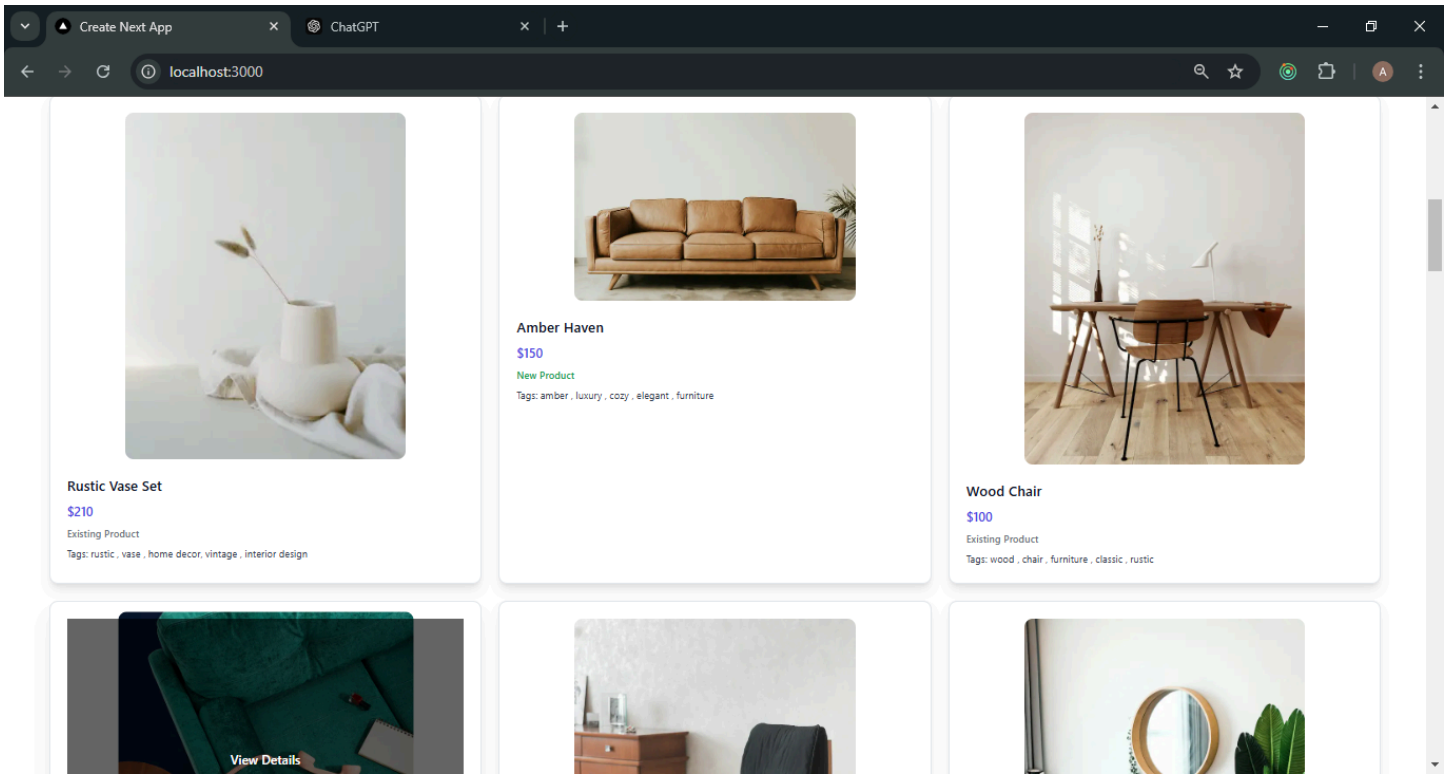
The website successfully passed all responsive tests, ensuring a seamless user experience on all platforms.

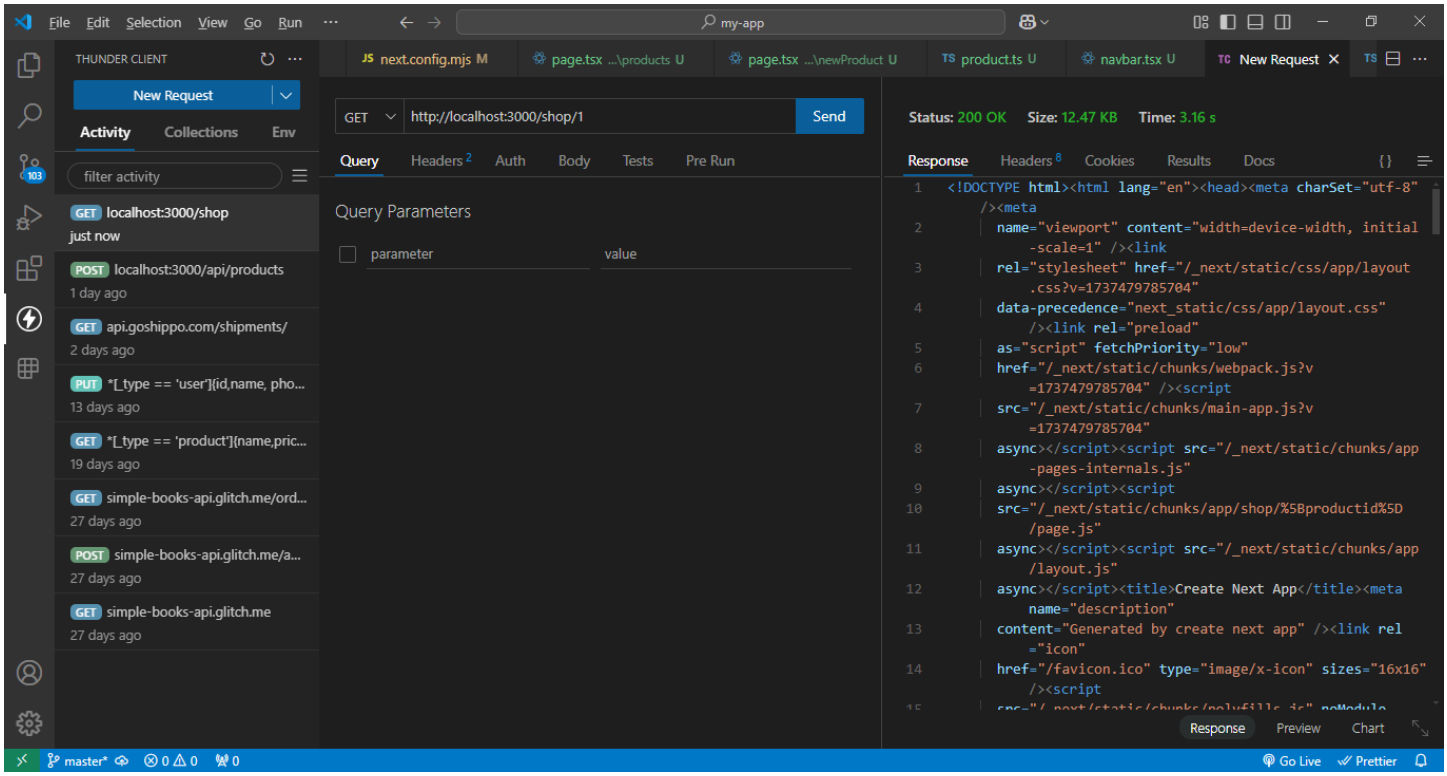
---

2. Product Page Testing

- **Integration Testing**: Products fetched from the API were tested using **Thunder Client**.

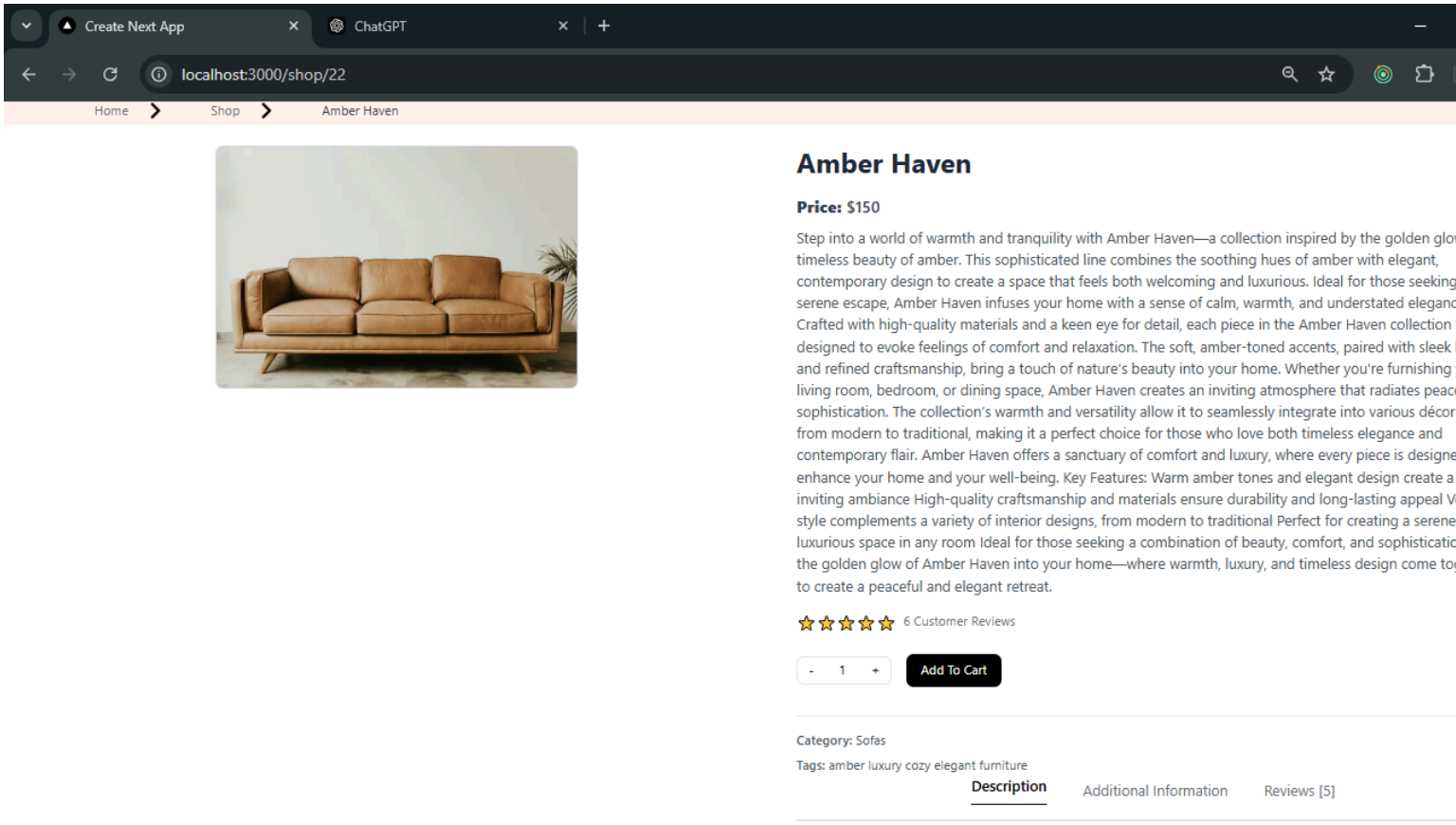
- **Data Fetching**: Verified that product data is accurately retrieved and displayed on the product listing page.
- **Responsiveness**: The product page layout adapts well to different screen sizes.





3. Product Detail Page Testing

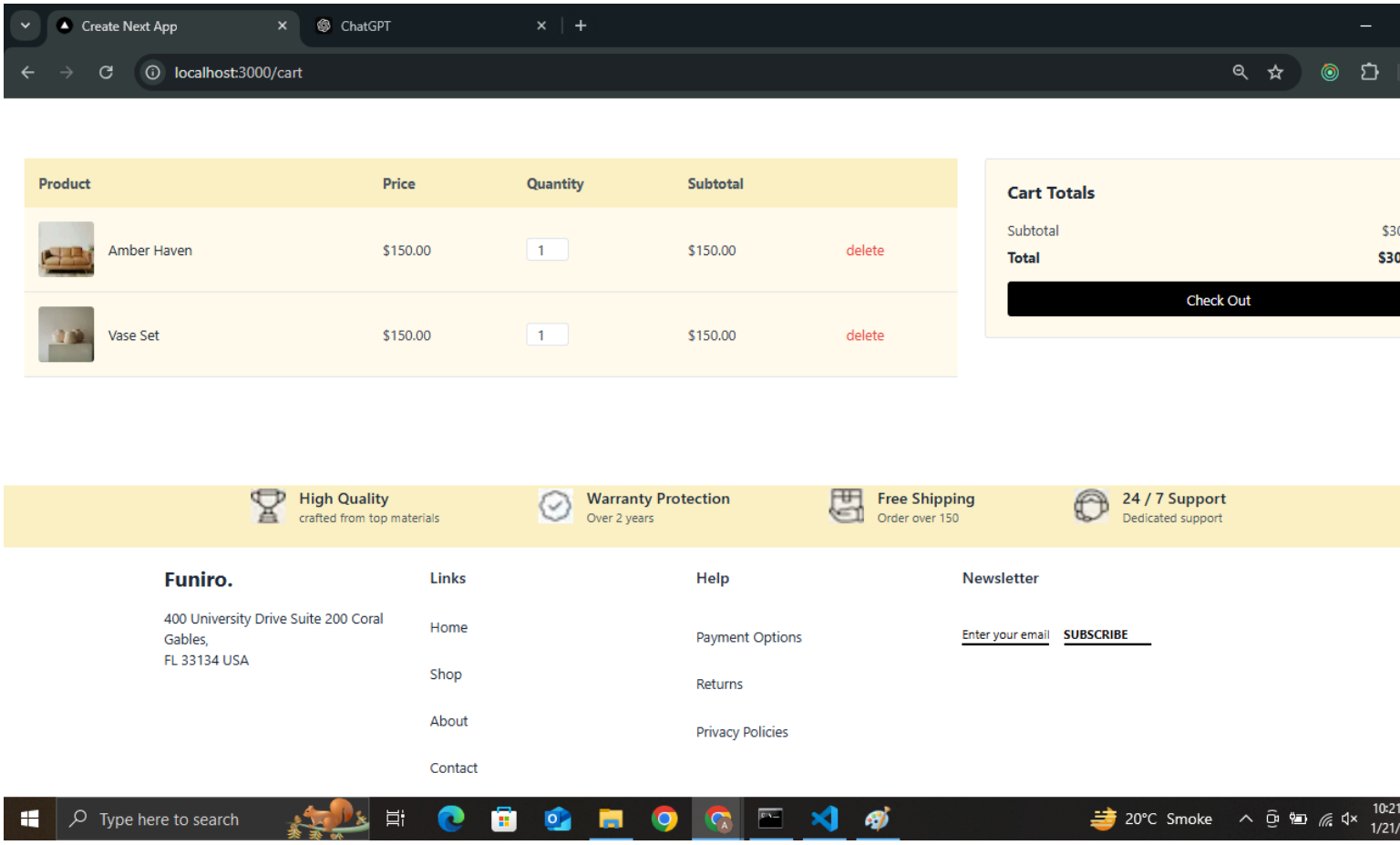
- **Dynamic Routing**: Successfully tested dynamic routing from the product listing page to the product detail page.
- **API Testing**: Endpoints for product IDs were validated using **Thunder Client**, ensuring correct data retrieval for individual products.



---

4. Cart Page Functionality

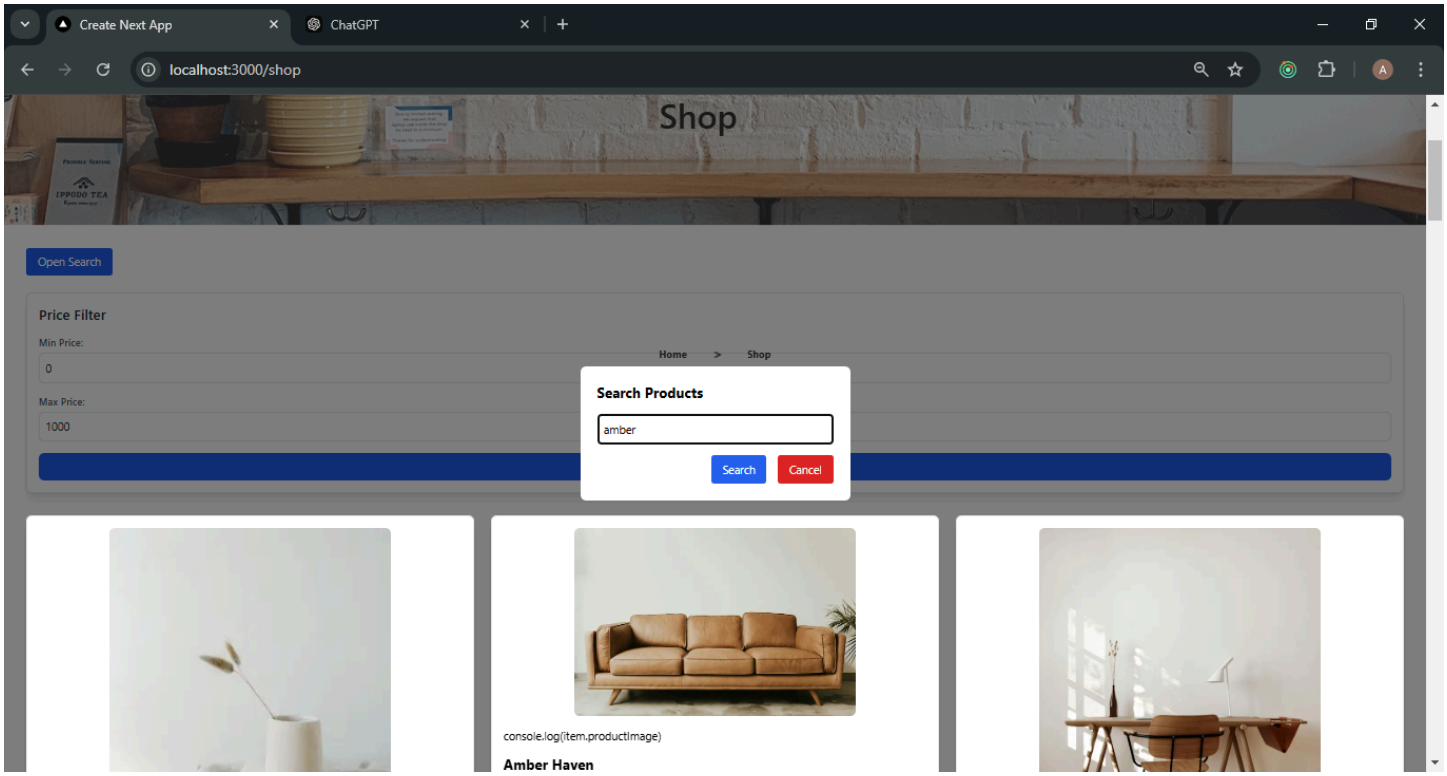
- Users can:
  - Add and remove products from the cart.
  - View the updated total price dynamically.
- **Result**: All functionalities are working as expected.

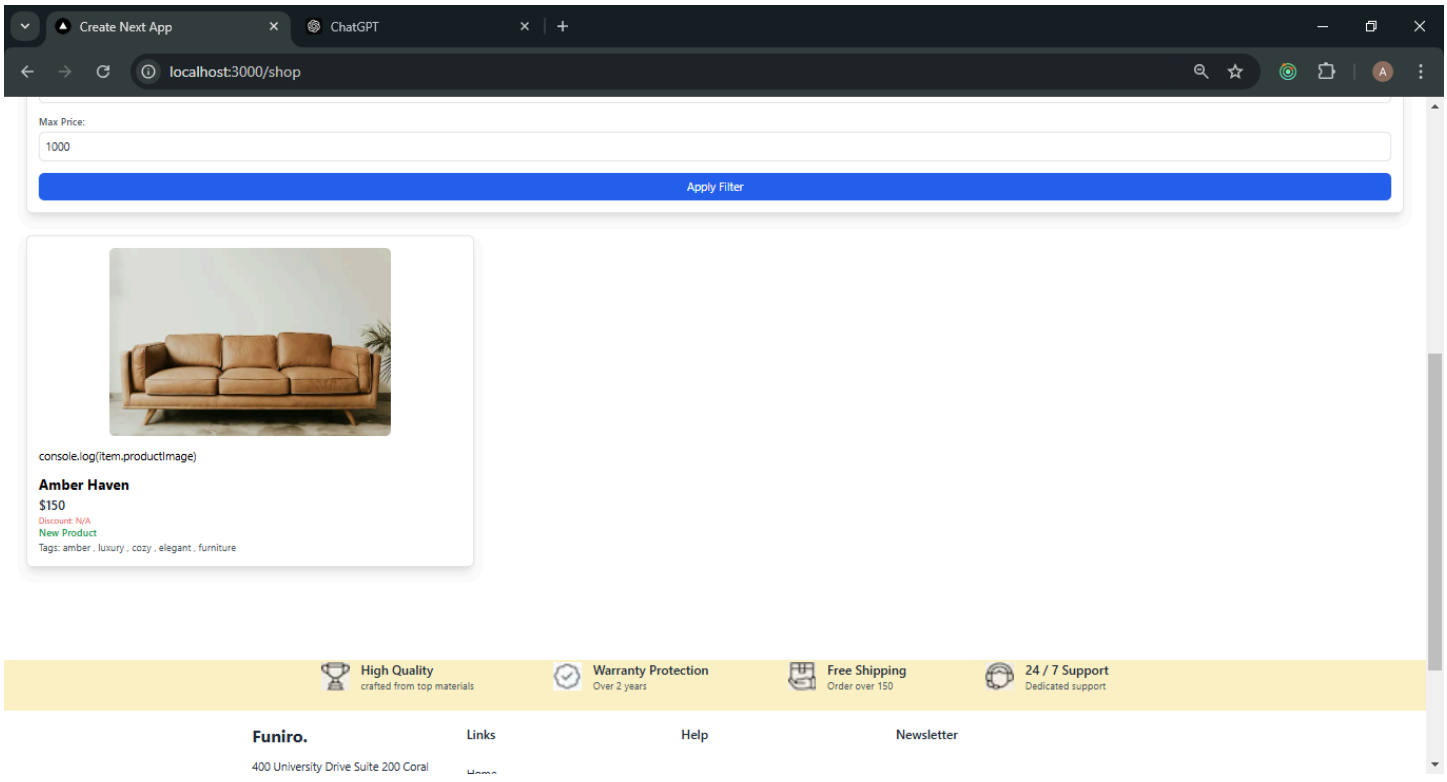


---

5. Search Bar Functionality

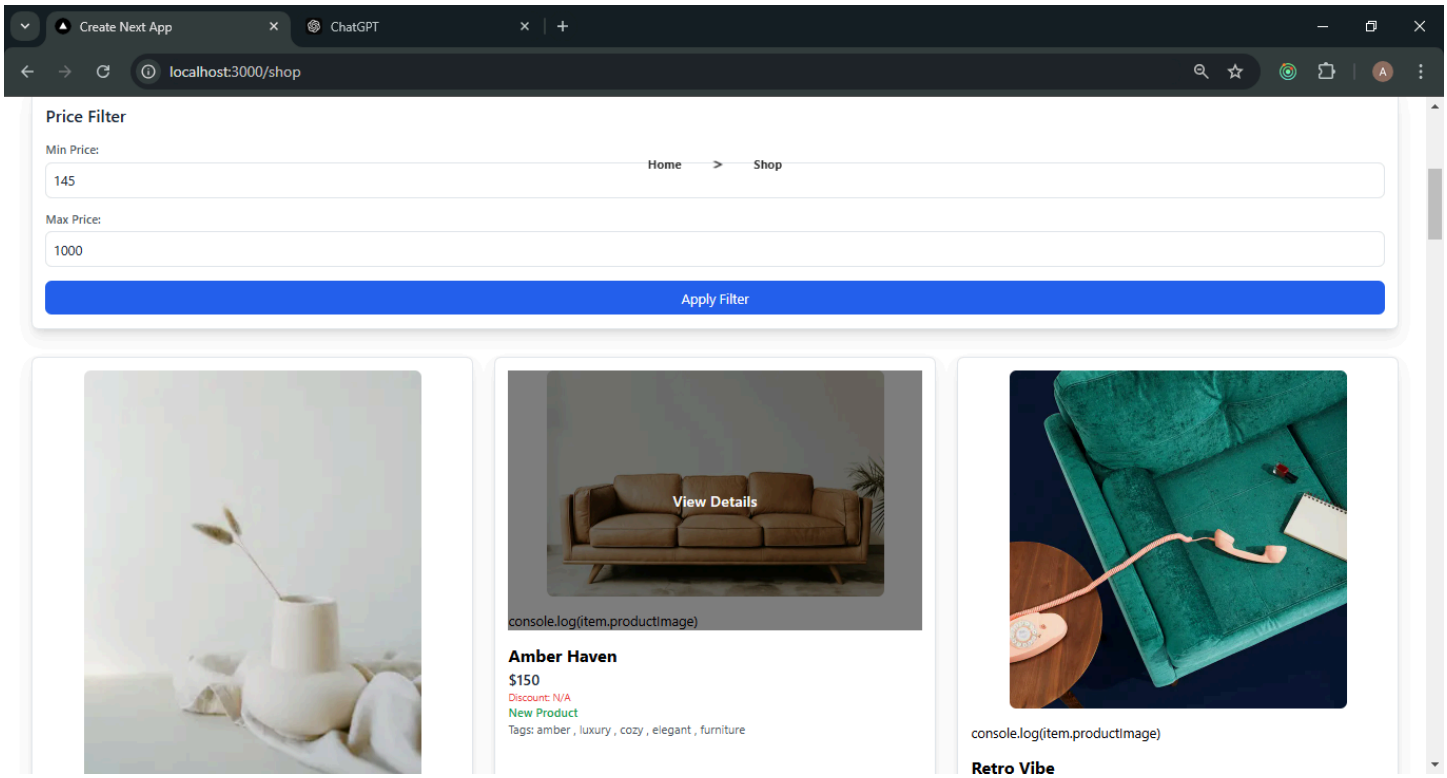
- Tested the search bar to ensure it retrieves and displays products based on:
  - **Name**
  - **Tags**
- **Result**: Products are correctly filtered and displayed on the screen.


Create Next App
ChatGPT
localhost:3000/shop
Shop
Open Search
Price Filter
Min Price:
0
Max Price:
1000
Home > Shop
Search Products
amber
Search
Cancel
console.log(item.productImage)
Amber Haven


Create Next App
ChatGPT
localhost:3000/shop
Max Price:
1000
Apply Filter
console.log(item.productImage)
Amber Haven
$150
Discount: N/A
New Product
Tags: amber , luxury , cozy , elegant , furniture
High Quality
crafted from top materials
Warranty Protection
Over 2 years
Free Shipping
Order over 150
24 / 7 Support
Dedicated support
Funiro.
400 University Drive Suite 200 Coral
Links
Help
Newsletter

6. Filter Functionality

- Tested the price filter to ensure it accurately narrows results based on the selected price range.
- **Result**: Filtering functionality works as expected.

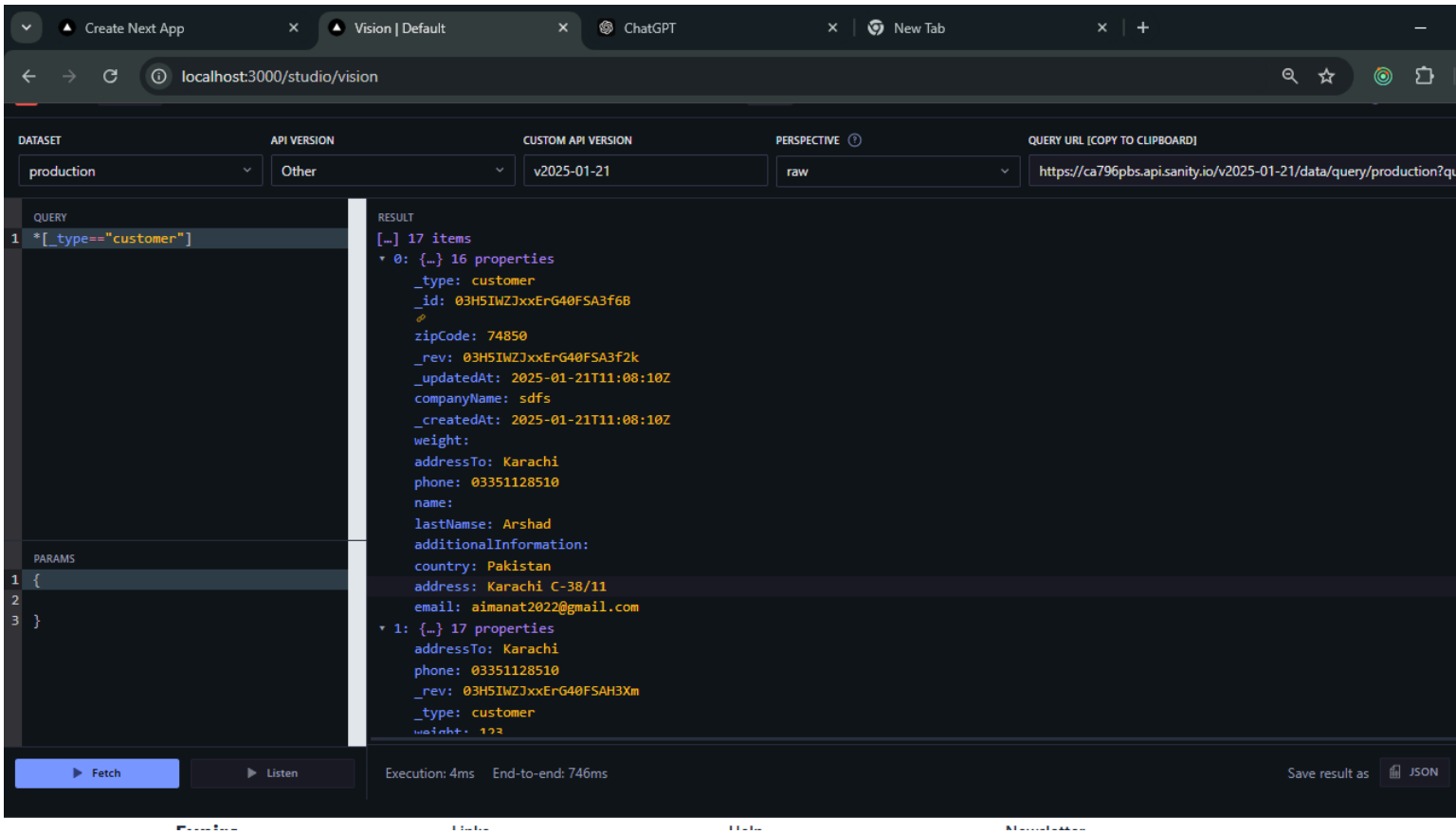


7. Shipment and Checkout Process Testing

- Verified by:
  - Entering customer and shipment details into the database.
  - Retrieving shipment data and a tracking number successfully.

8. User Data API Integration Testing

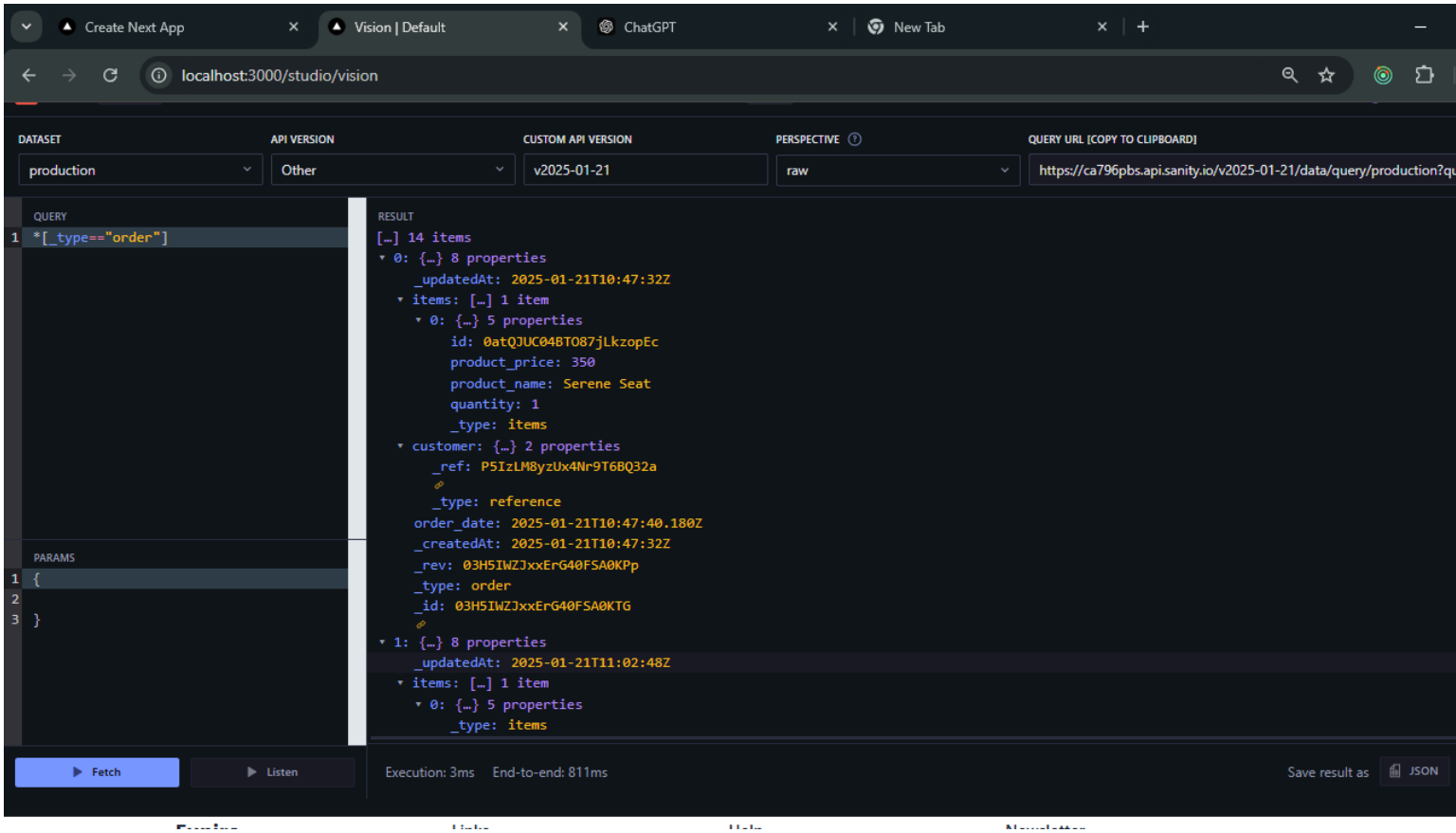
- User creation in **Sanity** was tested using **Thunder Client**.
- Queries verified user data creation and retrieval.
- **Result**: All user data APIs function as intended.



9. Order Data API Integration Testing

- Similar to user data, order creation in **Sanity** was tested via **Thunder Client**.

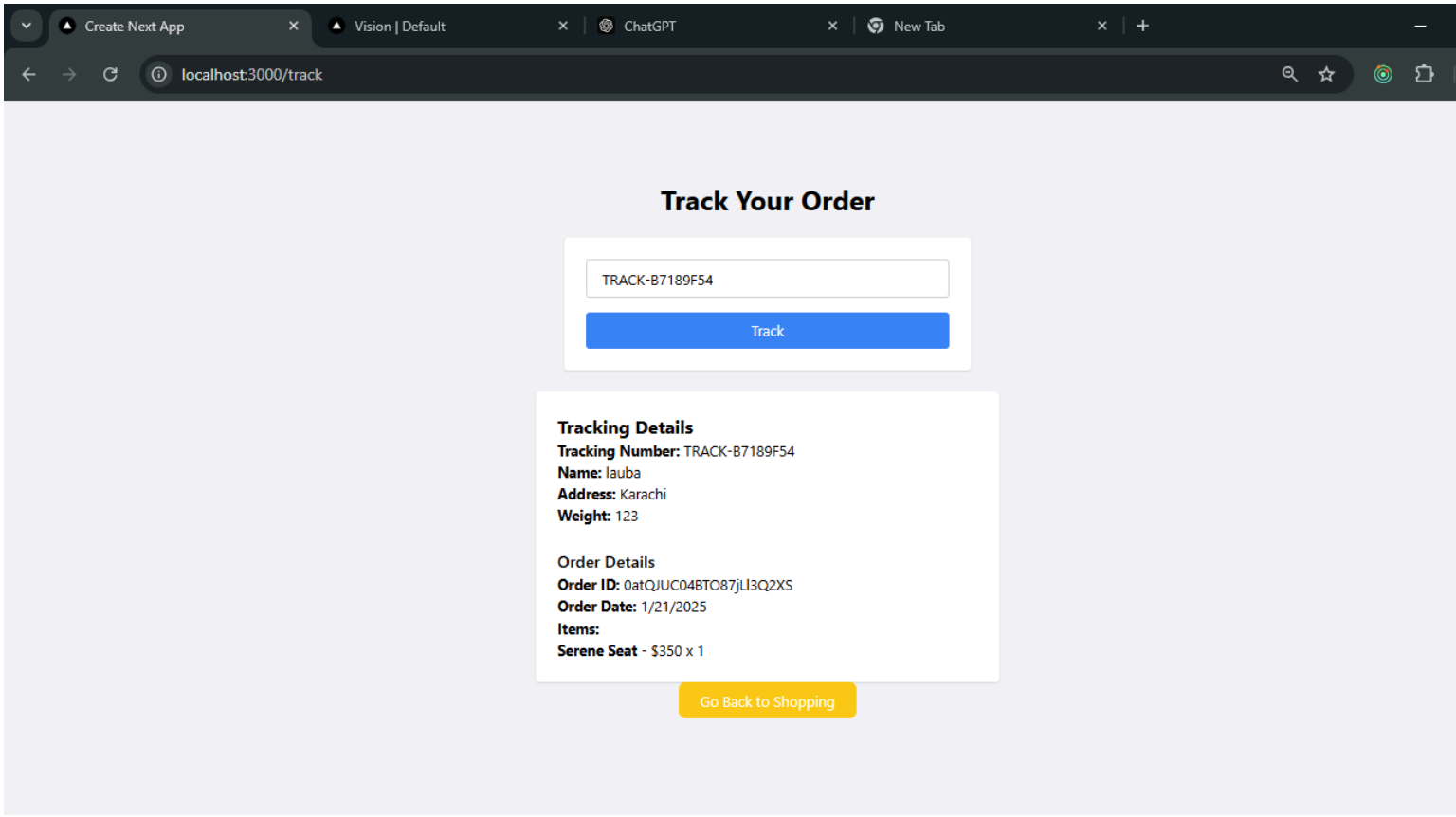
- **Result**: Order data is created and retrieved successfully.



---

10. Order Tracking (Shipo Integration)

- Data sent to **Shipo** successfully returned:
  - Tracking number
  - Tracking details
- **Error Handling**: Implemented to catch issues such as missing or invalid data. Fallback data outputs are displayed for unresolved errors.



---

11. Admin Panel Testing

Validated the admin panel for:

- User additions in shipments, which displayed correctly.
- All incoming orders were listed accurately.
- Product data appeared as expected.

Error Handling Implementations

Network Failures

- Display user-friendly error messages (e.g., "Unable to fetch data. Please check your connection.")

```
  try {
    const response = await fetch(`/api/track?trackingNumber=${trackingNumber}`);
    const data = await response.json();

    if (response.ok && data.customer) {
      setTrackingData(data); // Store fetched data (including both customer and order)
    } else {
      setError(data.error || "No data found for this tracking number.");
    }
  } catch (err) {
    setError("An error occurred while fetching the data. Please try again.");
  }
};
```

Invalid or Missing Data

- Show warnings if API returns incomplete or invalid data. Example: "Data not found. Please try again."

Unexpected Server Errors

- Fallback UI elements displayed:
  - "No products available" for empty API responses.
  - "Something went wrong. Please reload the page."

```
    <div className="mt-4">

                    <h2 className="text-xl font-bold">{item.title}</h2>

                    <p className="text-lg font-semibold
text-gray-800">${item.price}</p>

                    <p className="text-xs text-red-500">

                      Discount: {item.discountPercentage ?
`${item.discountPercentage}%` : "N/A"}

                    </p>

                    <p

                      className={`text-sm font-medium ${

                        item.isNew ? "text-green-600" : "text-gray-500"

                      }`}

                    >

                      {item.isNew ? "New Product" : "Existing Product"}

                    </p>

                    <p className="text-sm text-gray-600">

                      Tags:{" "}

                      {item.tags && item.tags.length > 0

                        ? item.tags.join(", ")
```

```
                    : "No tags available"}

                </p>

              </div>

            </div>

          ))

        ) : (

          <p>No products found matching your search or price filter.</p>

        )}
```

Permission Issues

- Display messages for unauthorized actions, such as:
  - "You do not have permission to perform this action."
