Day-4

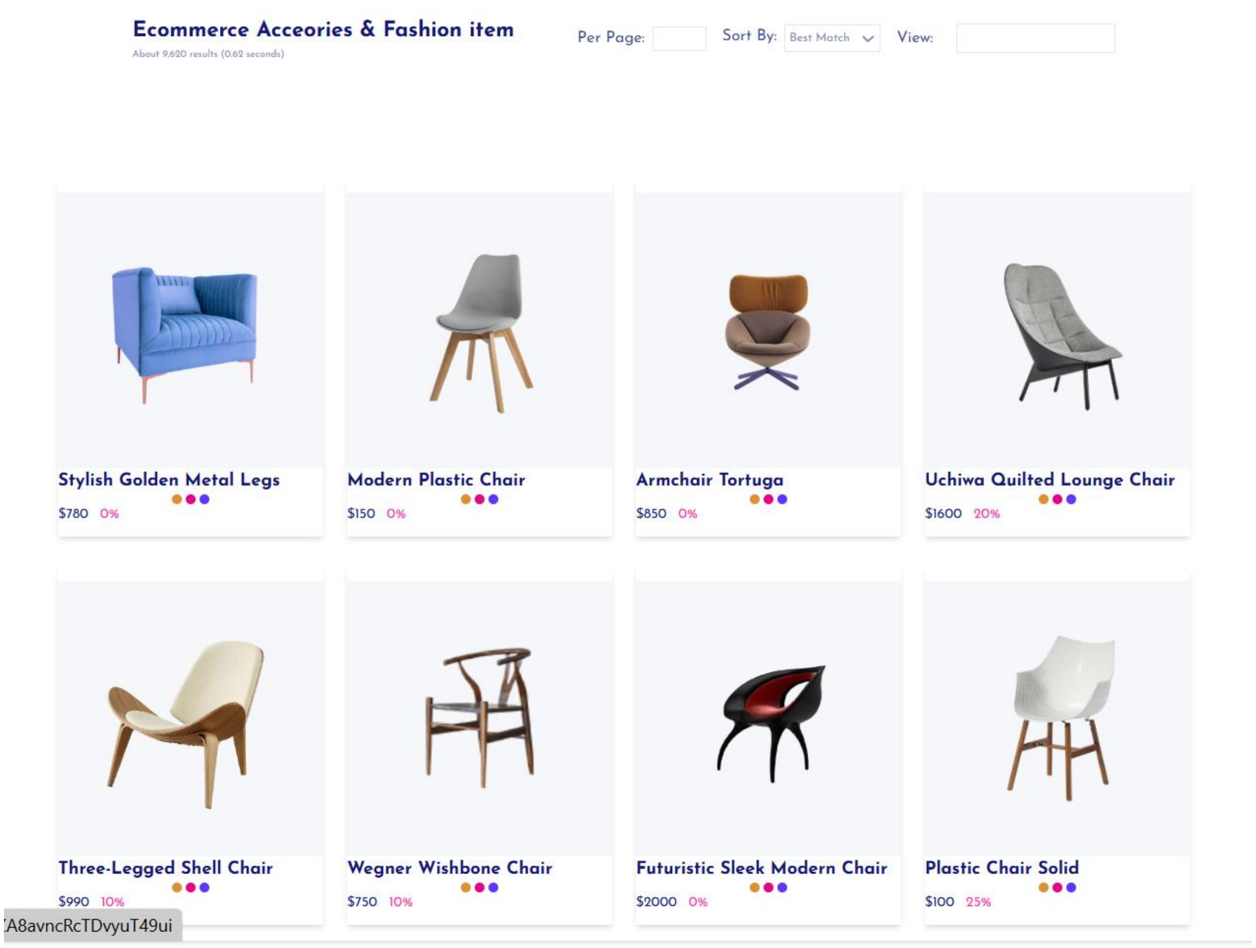
Building Dynamic Frontend Components

Table of Contents

With Functionality

1. Product Listing Component

Dynamically renders product data in a grid layout for better visual representation.


Ecommerce Acceories & Fashion item

About 9,620 results (0.62 seconds)

Per Page: Sort By: Best Match View:

Stylish Golden Metal Legs
$780 0%

Modern Plastic Chair
$150 0%

Armchair Tortuga
$850 0%

Uchiwa Quilted Lounge Chair
$1600 20%

Three-Legged Shell Chair
$990 10%

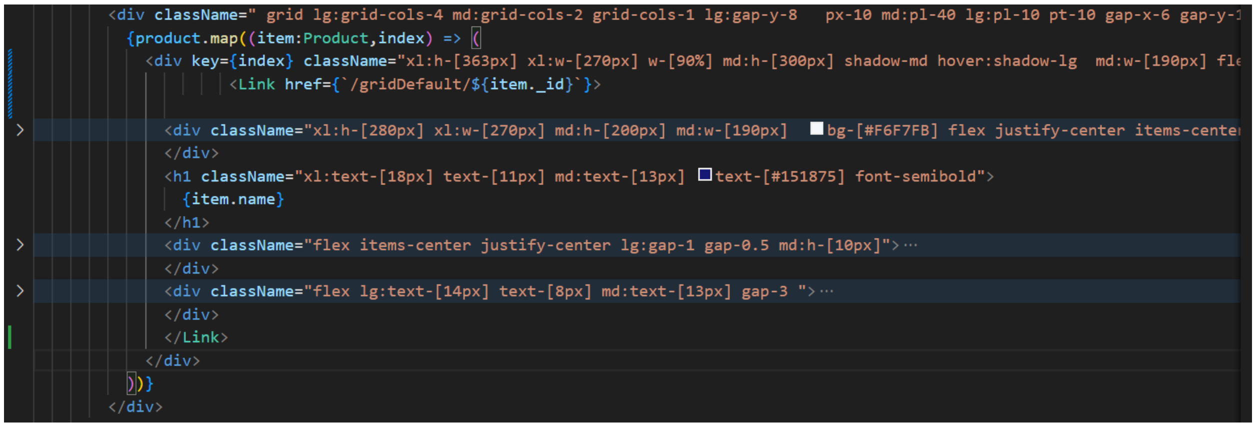
Wegner Wishbone Chair
$750 10%

Futuristic Sleek Modern Chair
$2000 0%

Plastic Chair Solid
$100 25%

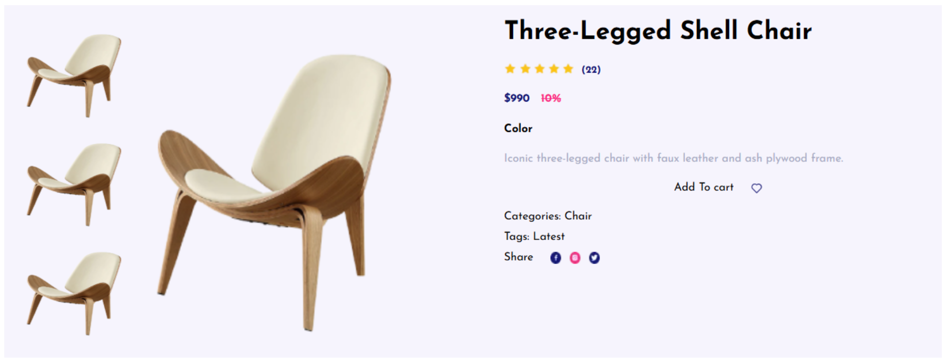
:A8avncRcTDvyuT49ui

Code Snippet:


```
<div className=" grid lg:grid-cols-4 md:grid-cols-2 grid-cols-1 lg:gap-y-8   px-10 md:pl-40 lg:pl-10 pt-10 gap-x-6 gap-y-1
 {product.map((item:Product,index) => (
  <div key={index} className="xl:h-[363px] xl:w-[270px] w-[90%] md:h-[300px] shadow-md hover:shadow-lg  md:w-[190px] fle
       <Link href={`/gridDefault/${item._id}`}>

   <div className="xl:h-[280px] xl:w-[270px] md:h-[200px] md:w-[190px]  bg-[#F6F7FB] flex justify-center items-cente
   </div>
   <h1 className="xl:text-[18px] text-[11px] md:text-[13px] text-[#151875] font-semibold">
    {item.name}
   </h1>
   <div className="flex items-center justify-center lg:gap-1 gap-0.5 md:h-[10px]">⋯
   </div>
   <div className="flex lg:text-[14px] text-[8px] md:text-[13px] gap-3 ">⋯
   </div>
   </Link>
  </div>
 ))}
</div>
```

2. Product Detail Component

Displays detailed information about a specific product, dynamically rendering content.


Three-Legged Shell Chair

★★★★★ (22)

$990 10%

Color

Iconic three-legged chair with faux leather and ash plywood frame.

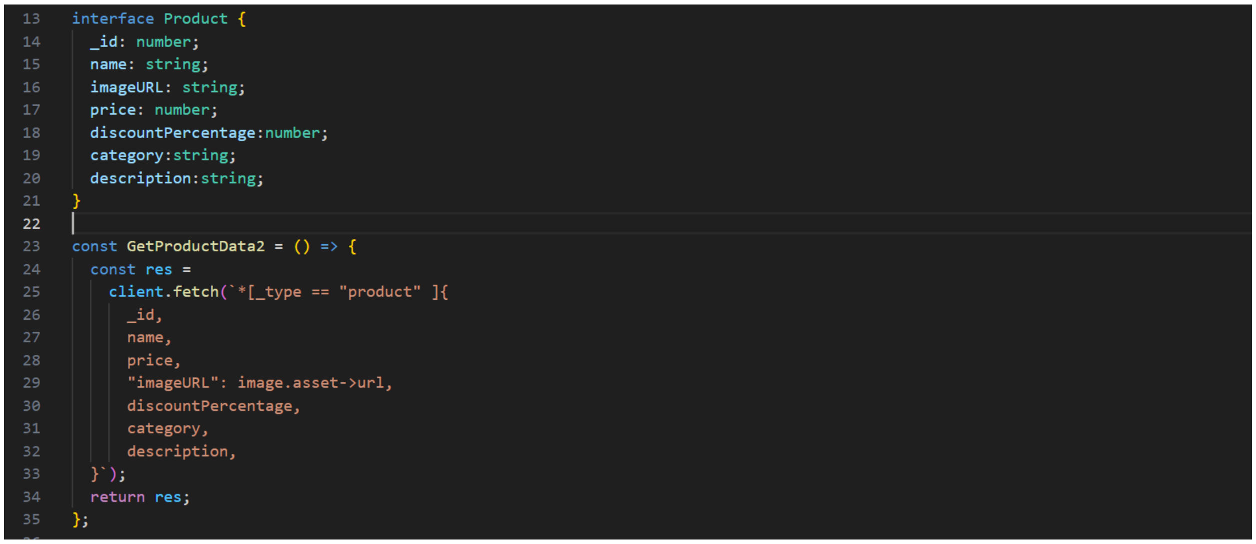
Add To cart

Categories: Chair

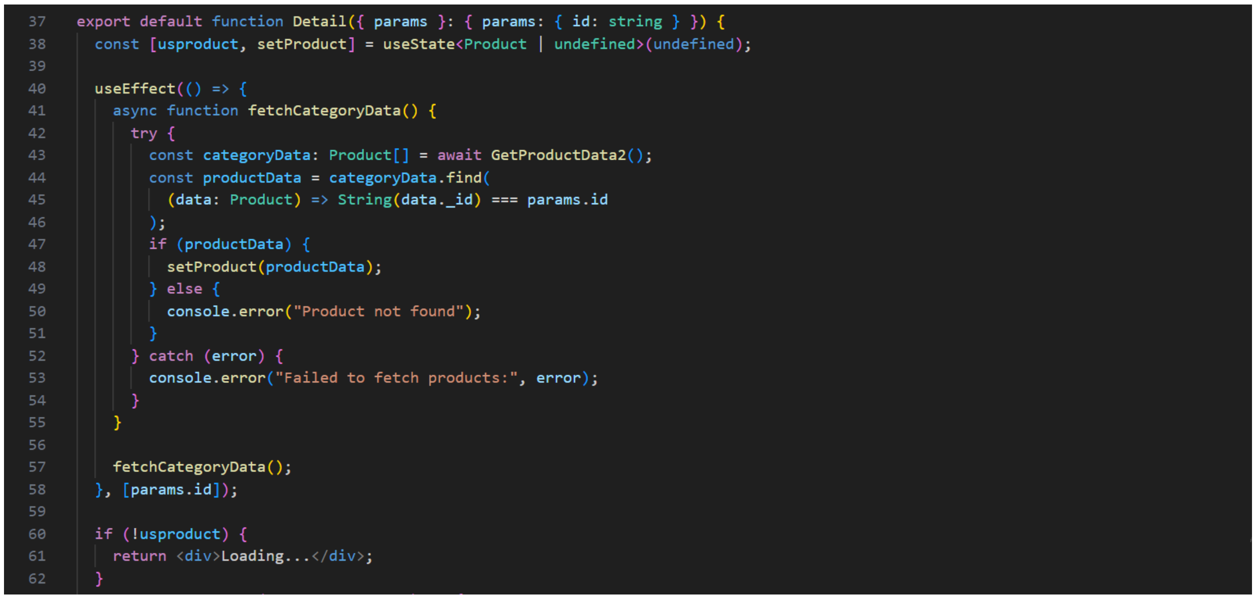
Tags: Latest

Share

Code Snippet:

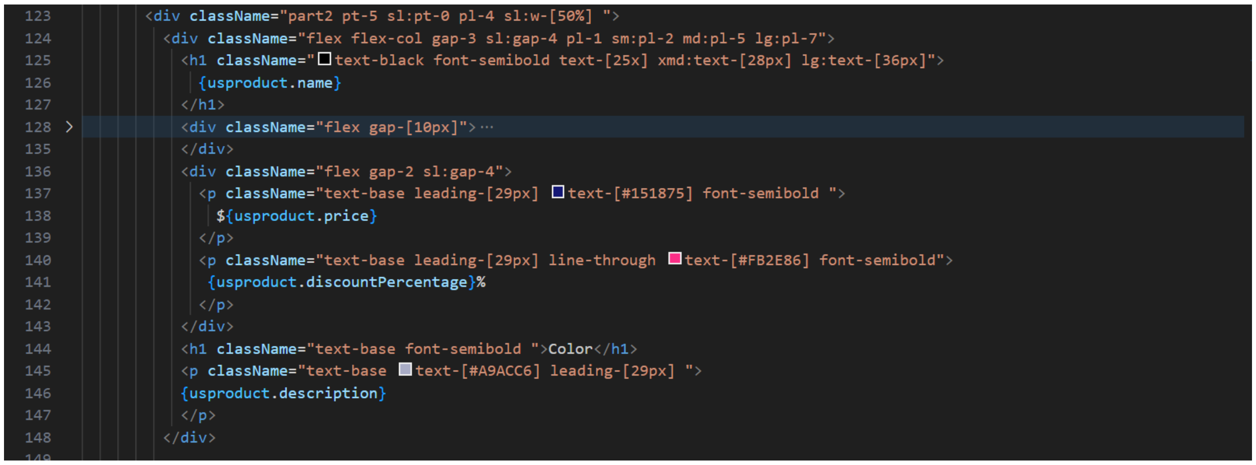

```
interface Product {
  _id: number;
  name: string;
  imageURL: string;
  price: number;
  discountPercentage:number;
  category:string;
  description:string;
}

const GetProductData2 = () => {
  const res =
    client.fetch(`*[_type == "product" ]{
      _id,
      name,
      price,
      "imageURL": image.asset->url,
      discountPercentage,
      category,
      description,
  }`);
  return res;
};
```

This Detail component fetches product details based on the params.id when the component loads. It displays a loading message until the product data is successfully fetched and set in the usproduct state.



```
export default function Detail({ params }: { params: { id: string } }) {
  const [usproduct, setProduct] = useState<Product | undefined>(undefined);

  useEffect(() => {
    async function fetchCategoryData() {
      try {
        const categoryData: Product[] = await GetProductData2();
        const productData = categoryData.find(
          (data: Product) => String(data._id) === params.id
        );
        if (productData) {
          setProduct(productData);
        } else {
          console.error("Product not found");
        }
      } catch (error) {
        console.error("Failed to fetch products:", error);
      }
    }

    fetchCategoryData();
  }, [params.id]);

  if (!usproduct) {
    return <div>Loading...</div>;
  }
```

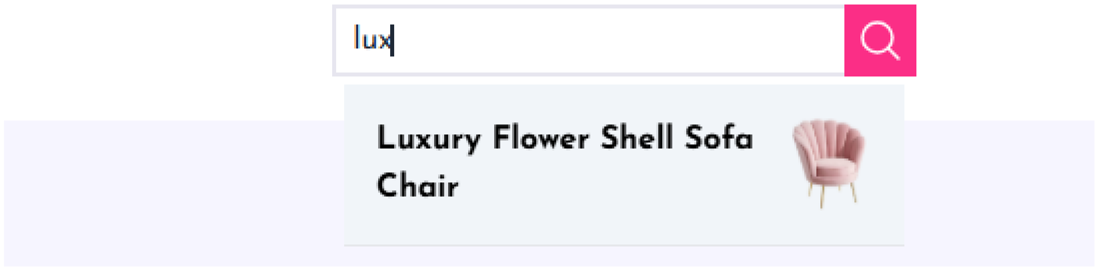
Use {usProduct} at appropriate places to bind data effectively.



```
<div className="part2 pt-5 sl:pt-0 pl-4 sl:w-[50%] ">
  <div className="flex flex-col gap-3 sl:gap-4 pl-1 sm:pl-2 md:pl-5 lg:pl-7">
    <h1 className="text-black font-semibold text-[25x] xmd:text-[28px] lg:text-[36px]">
      {usproduct.name}
    </h1>
    <div className="flex gap-[10px]">…
    </div>
    <div className="flex gap-2 sl:gap-4">
      <p className="text-base leading-[29px] text-[#151875] font-semibold ">
        ${usproduct.price}
      </p>
      <p className="text-base leading-[29px] line-through text-[#FB2E86] font-semibold">
       {usproduct.discountPercentage}%
      </p>
    </div>
    <h1 className="text-base font-semibold ">Color</h1>
    <p className="text-base text-[#A9ACC6] leading-[29px] ">
    {usproduct.description}
    </p>
  </div>
```

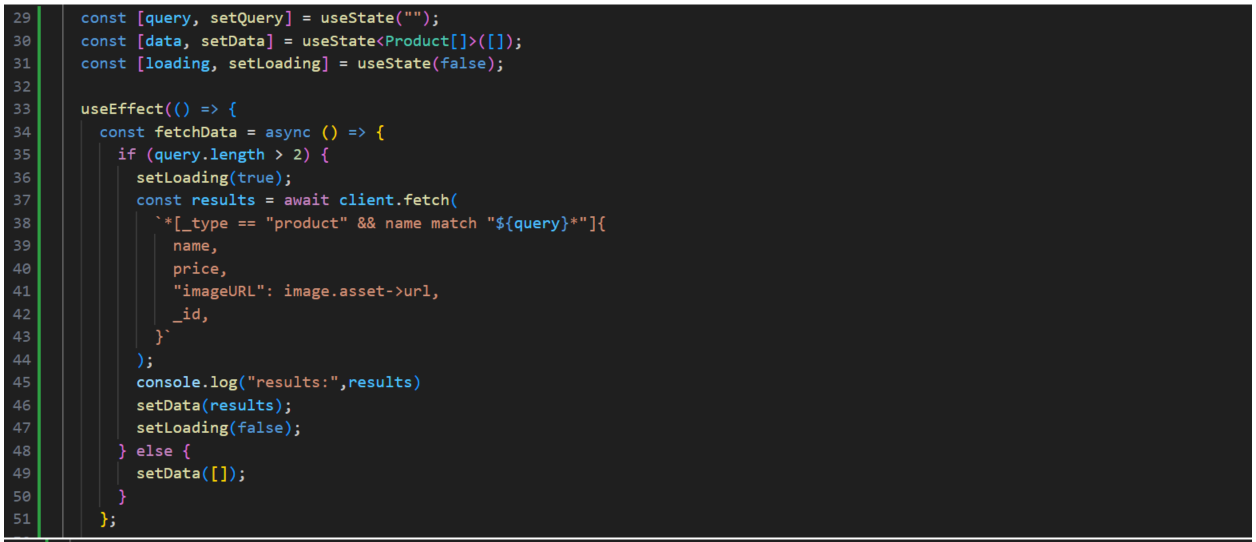
3. Search Bar

Implements search functionality to filter products by **tags**, enabling users to find relevant items quickly.

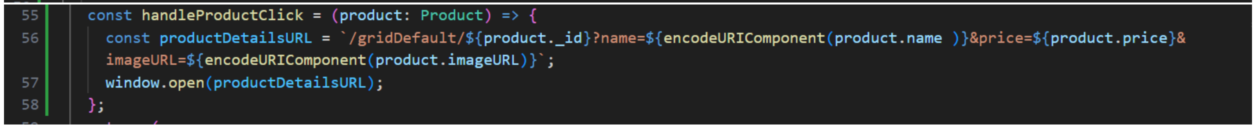


Code Snippet:

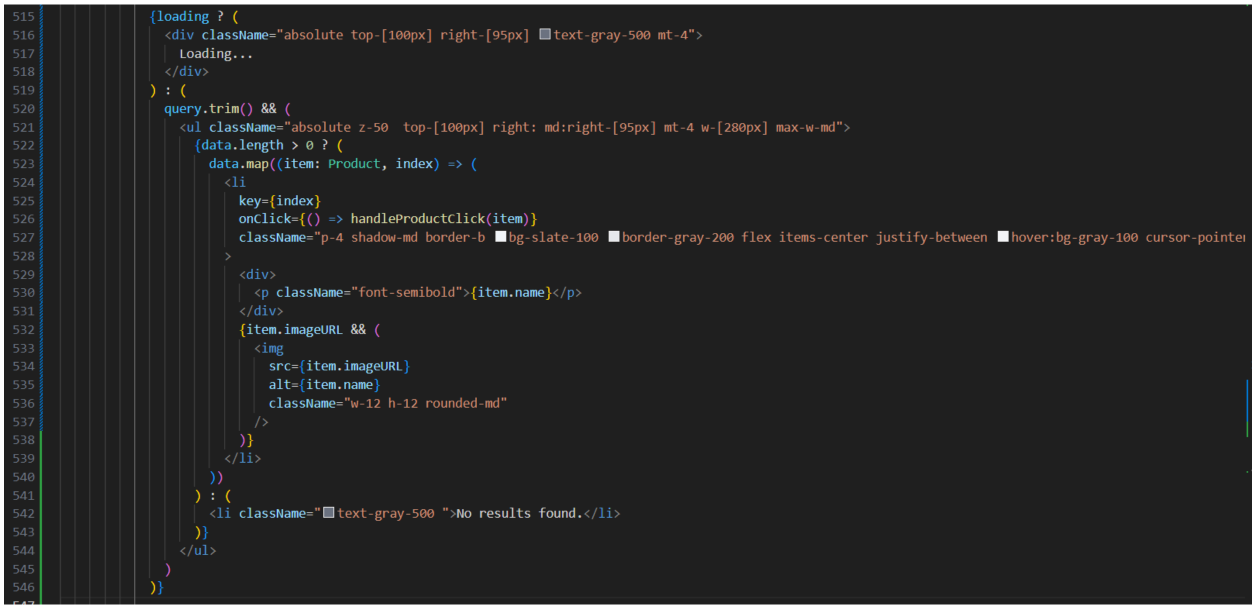
This code manages a search bar that fetches product data from a database when the query is more than 2 characters long. It updates the data state with matching results and displays a loading indicator during the fetch. Clicking on a product opens its detail page in a new tab with pre-filled information in the URL.



```
const [query, setQuery] = useState("");
const [data, setData] = useState<Product[]>([]);
const [loading, setLoading] = useState(false);

useEffect(() => {
  const fetchData = async () => {
    if (query.length > 2) {
      setLoading(true);
      const results = await client.fetch(
        `*[_type == "product" && name match "${query}*"]{
          name,
          price,
          "imageURL": image.asset->url,
          _id,
        }`
      );
      console.log("results:",results)
      setData(results);
      setLoading(false);
    } else {
      setData([]);
    }
  };
```



```
const handleProductClick = (product: Product) => {
  const productDetailsURL = `/gridDefault/${product._id}?name=${encodeURIComponent(product.name )}&price=${product.price}&
  imageURL=${encodeURIComponent(product.imageURL)}`;
  window.open(productDetailsURL);
};
```

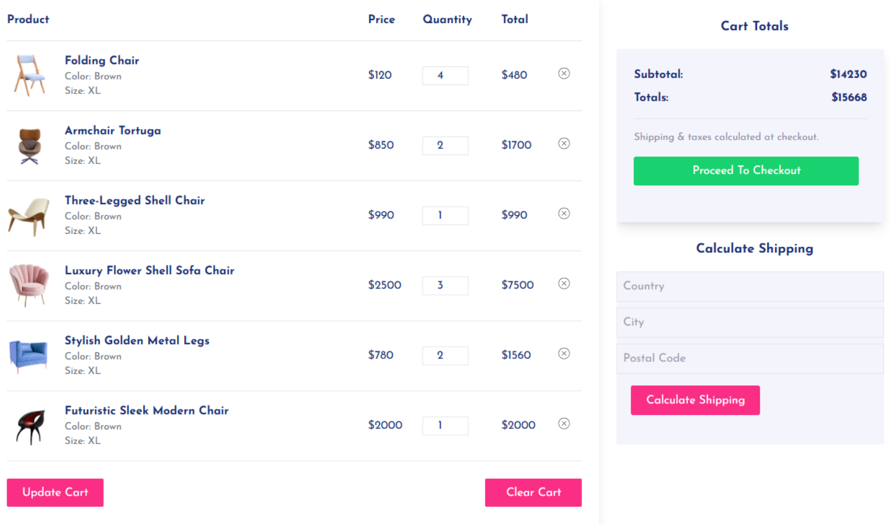


```
{loading ? (
  <div className="absolute top-[100px] right-[95px] text-gray-500 mt-4">
    Loading...
  </div>
) : (
  query.trim() && (
    <ul className="absolute z-50  top-[100px] right: md:right-[95px] mt-4 w-[280px] max-w-md">
      {data.length > 0 ? (
        data.map((item: Product, index) => (
          <li
            key={index}
            onClick={() => handleProductClick(item)}
            className="p-4 shadow-md border-b bg-slate-100 border-gray-200 flex items-center justify-between hover:bg-gray-100 cursor-pointe
          >
            <div>
              <p className="font-semibold">{item.name}</p>
            </div>
            {item.imageURL && (
              <img
                src={item.imageURL}
                alt={item.name}
                className="w-12 h-12 rounded-md"
              />
            )}
          </li>
        ))
      ) : (
        <li className="text-gray-500 ">No results found.</li>
      )}
    </ul>
  )
)}
```

4. Cart Component

Displays added items, calculates totals and subtotals, and includes the following functionalities:

- Clear Cart
- Remove Item



Product	Price	Quantity	Total	
Folding Chair Color: Brown Size: XL	$120	4	$480	⊗
Armchair Tortuga Color: Brown Size: XL	$850	2	$1700	⊗
Three-Legged Shell Chair Color: Brown Size: XL	$990	1	$990	⊗
Luxury Flower Shell Sofa Chair Color: Brown Size: XL	$2500	3	$7500	⊗
Stylish Golden Metal Legs Color: Brown Size: XL	$780	2	$1560	⊗
Futuristic Sleek Modern Chair Color: Brown Size: XL	$2000	1	$2000	⊗

Update Cart   Clear Cart

Cart Totals

Subtotal: $14230
Totals: $15668

Shipping & taxes calculated at checkout.

Proceed To Checkout

Calculate Shipping

Country
City
Postal Code

Calculate Shipping

Code Snippet:

Remove a single product:

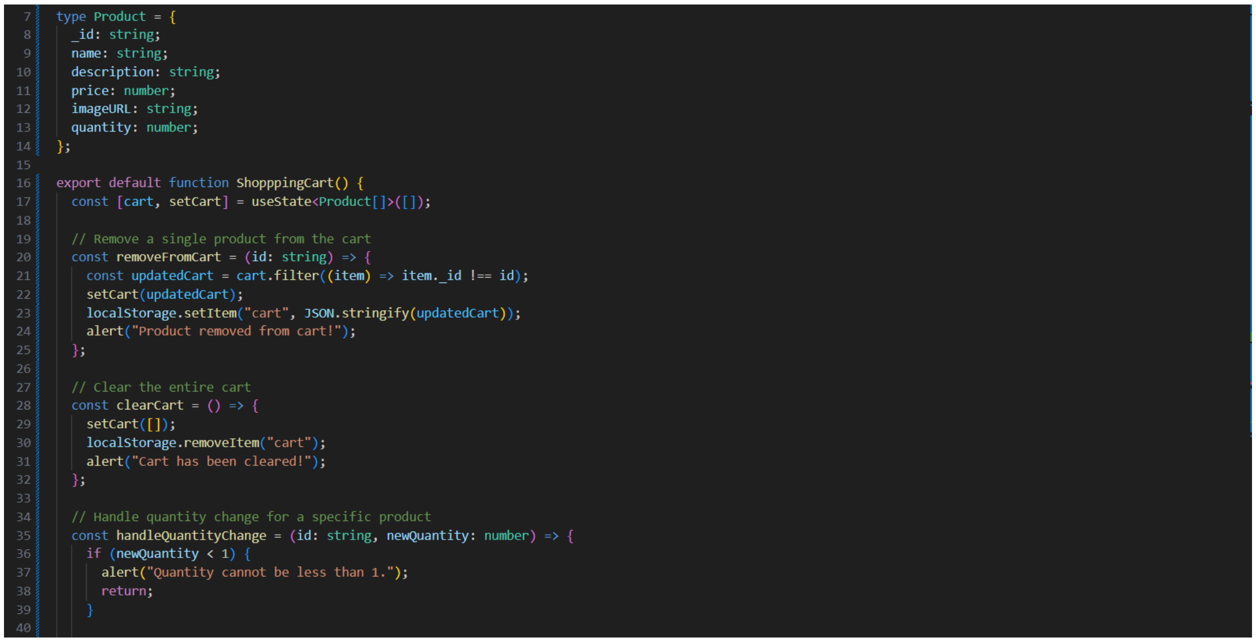
- The removeFromCart function filters out the product with the matching id from the cart, updates the state, and saves the updated cart to localStorage. It also displays an alert confirming the removal.

Clear the cart:

- The clearCart function clears the entire cart state and removes it from localStorage, displaying an alert that the cart has been cleared.

Handle quantity change:

- The handleQuantityChange function updates the quantity of a specific product in the cart. If the new quantity is less than 1, it shows an alert to prevent invalid values.



```
type Product = {
  _id: string;
  name: string;
  description: string;
  price: number;
  imageURL: string;
  quantity: number;
};

export default function ShopppingCart() {
  const [cart, setCart] = useState<Product[]>([]);

  // Remove a single product from the cart
  const removeFromCart = (id: string) => {
    const updatedCart = cart.filter((item) => item._id !== id);
    setCart(updatedCart);
    localStorage.setItem("cart", JSON.stringify(updatedCart));
    alert("Product removed from cart!");
  };

  // Clear the entire cart
  const clearCart = () => {
    setCart([]);
    localStorage.removeItem("cart");
    alert("Cart has been cleared!");
  };

  // Handle quantity change for a specific product
  const handleQuantityChange = (id: string, newQuantity: number) => {
    if (newQuantity < 1) {
      alert("Quantity cannot be less than 1.");
      return;
    }

```

Update product quantity:

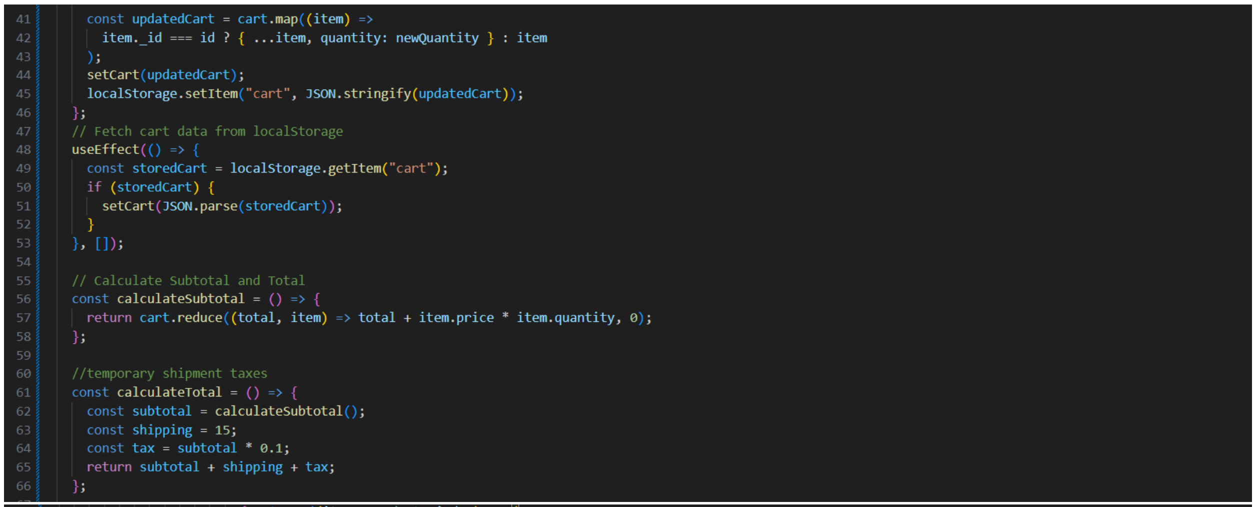
- The updatedCart variable adjusts the quantity of a specific product in the cart using the map method. If the product's id matches, it updates its quantity; otherwise, it leaves the product unchanged. The updated cart is saved to the state and localStorage.

Fetch cart data:

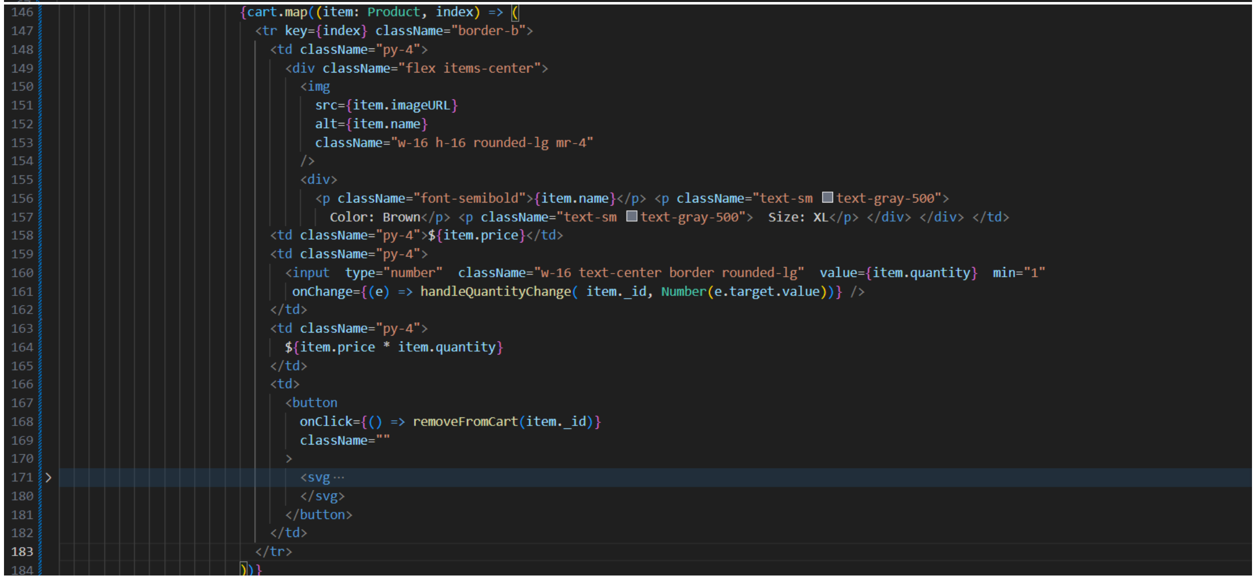
- The useEffect hook retrieves the cart data from localStorage on the initial render and updates the cart state if data exists.

Calculate subtotal:

- The calculateSubtotal function computes the subtotal by summing up the product of price and quantity for all items in the cart.



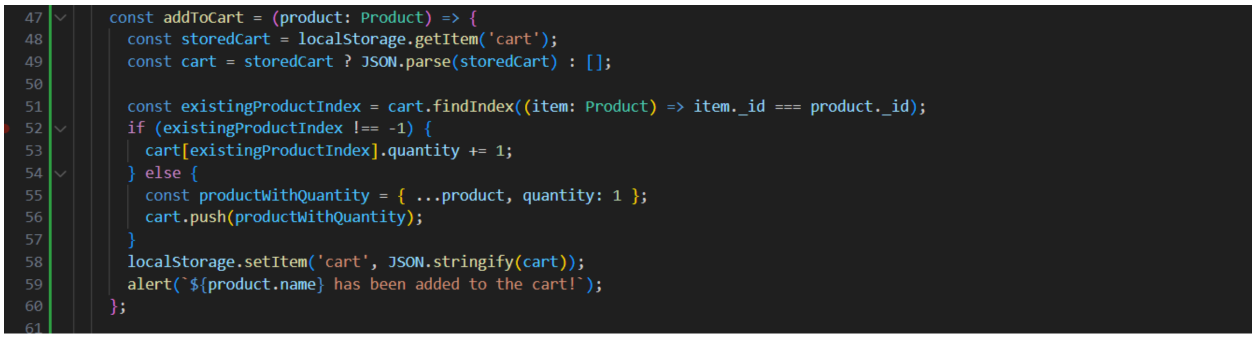
```
    const updatedCart = cart.map((item) =>
      item._id === id ? { ...item, quantity: newQuantity } : item
    );
    setCart(updatedCart);
    localStorage.setItem("cart", JSON.stringify(updatedCart));
  };
  // Fetch cart data from localStorage
  useEffect(() => {
    const storedCart = localStorage.getItem("cart");
    if (storedCart) {
      setCart(JSON.parse(storedCart));
    }
  }, []);

  // Calculate Subtotal and Total
  const calculateSubtotal = () => {
    return cart.reduce((total, item) => total + item.price * item.quantity, 0);
  };

  //temporary shipment taxes
  const calculateTotal = () => {
    const subtotal = calculateSubtotal();
    const shipping = 15;
    const tax = subtotal * 0.1;
    return subtotal + shipping + tax;
  };
```



```
                {cart.map((item: Product, index) => (
                  <tr key={index} className="border-b">
                    <td className="py-4">
                      <div className="flex items-center">
                        <img
                          src={item.imageURL}
                          alt={item.name}
                          className="w-16 h-16 rounded-lg mr-4"
                        />
                        <div>
                          <p className="font-semibold">{item.name}</p> <p className="text-sm text-gray-500">
                            Color: Brown</p> <p className="text-sm text-gray-500">  Size: XL</p> </div> </div> </td>
                    <td className="py-4">${item.price}</td>
                    <td className="py-4">
                      <input  type="number"  className="w-16 text-center border rounded-lg"  value={item.quantity}  min="1"
                       onChange={(e) => handleQuantityChange( item._id, Number(e.target.value))} />
                    </td>
                    <td className="py-4">
                      ${item.price * item.quantity}
                    </td>
                    <td>
                      <button
                        onClick={() => removeFromCart(item._id)}
                        className=""
                      >
                        <svg···
                        </svg>
                      </button>
                    </td>
                  </tr>
                ))}
```

Here is the code where we can add product in the cart

This function adds a product to the cart; if the product already exists, it increases its quantity, and if not, it creates a new entry with quantity 1.



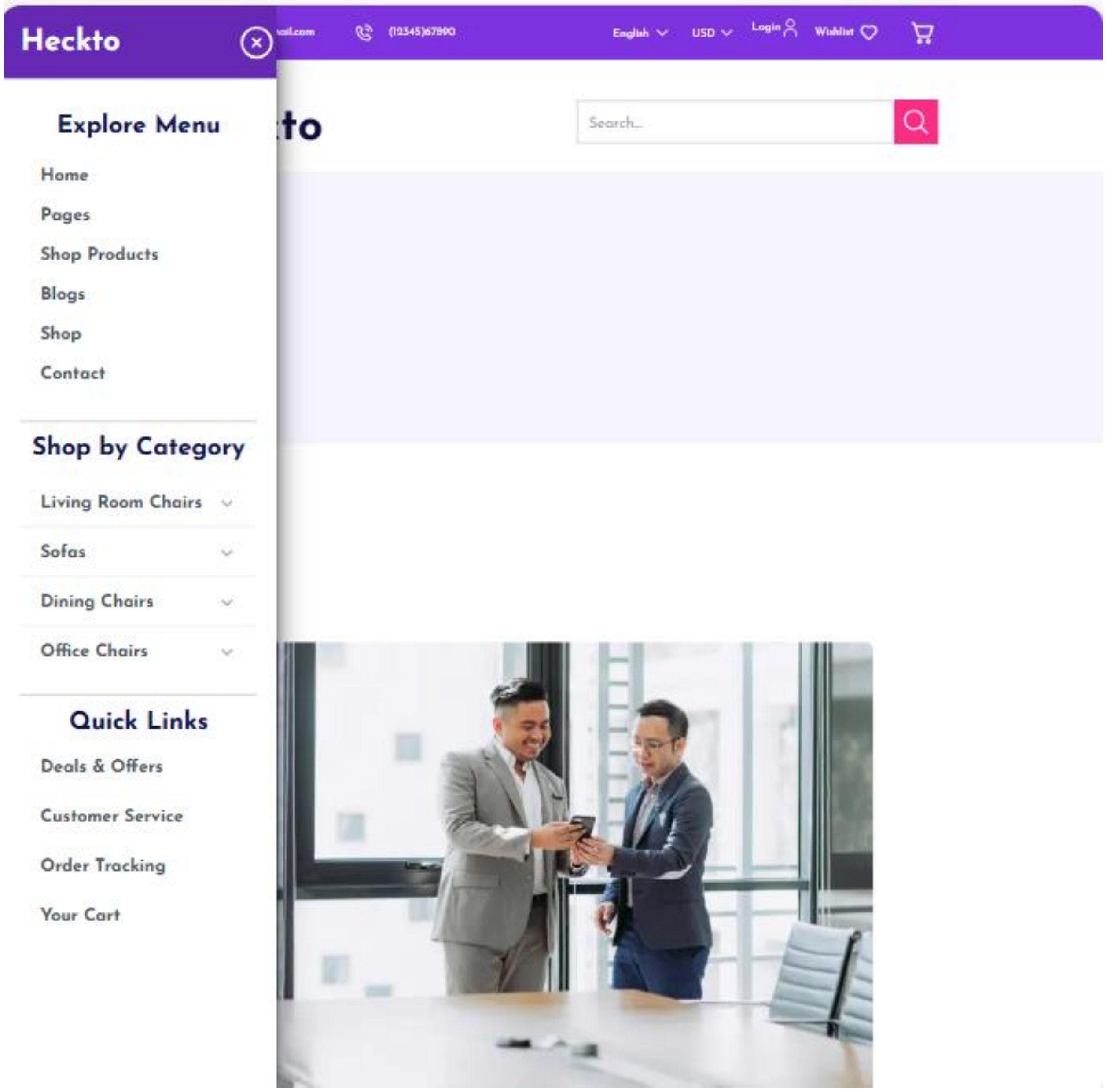
```
  const addToCart = (product: Product) => {
    const storedCart = localStorage.getItem('cart');
    const cart = storedCart ? JSON.parse(storedCart) : [];

    const existingProductIndex = cart.findIndex((item: Product) => item._id === product._id);
    if (existingProductIndex !== -1) {
      cart[existingProductIndex].quantity += 1;
    } else {
      const productWithQuantity = { ...product, quantity: 1 };
      cart.push(productWithQuantity);
    }
    localStorage.setItem('cart', JSON.stringify(cart));
    alert(`${product.name} has been added to the cart!`);
  };
```

5. Header and Footer Component

Provides a structured navigation experience with the following features:

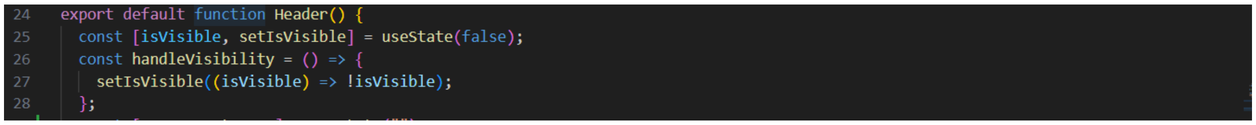
- **Hamburger Menu:** For responsive navigation
- **Dropdown Navigation:** For organized menu options.
- **Search Bar:** Integrated for quick product searches.

⇨ Hamburger Menu:



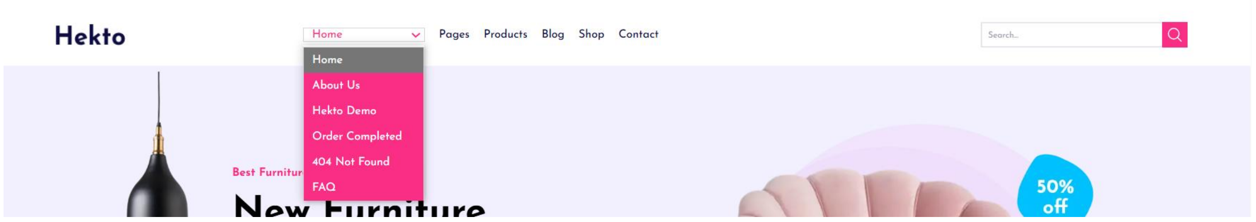
Code Snippet:

This function manages the visibility of an element. It toggles the isVisible state between true and false whenever handleVisibility is called.



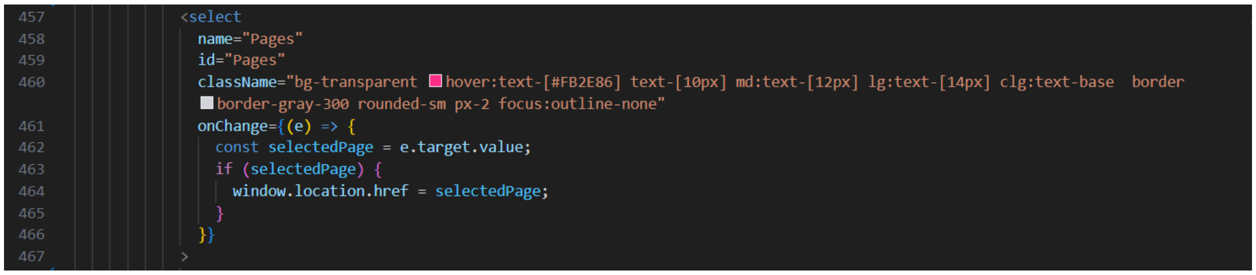
```
export default function Header() {
  const [isVisible, setIsVisible] = useState(false);
  const handleVisibility = () => {
    setIsVisible((isVisible) => !isVisible);
  };
```

Dropdown Navigation:



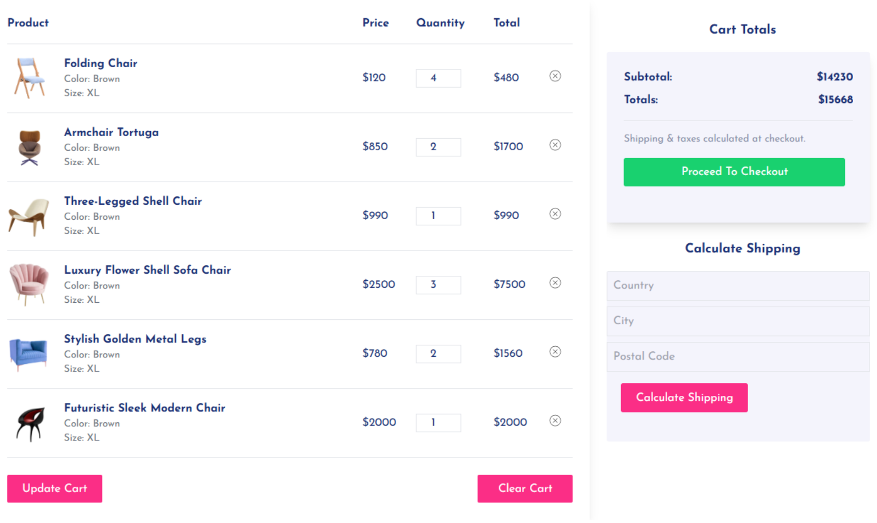
Code Snippet:

This code renders a dropdown (<select>) element. When a user selects an option, the onChange event handler changes the current page to the selected one by updating window.location.href.



```
<select
  name="Pages"
  id="Pages"
  className="bg-transparent hover:text-[#FB2E86] text-[10px] md:text-[12px] lg:text-[14px] clg:text-base  border
  border-gray-300 rounded-sm px-2 focus:outline-none"
  onChange={(e) => {
    const selectedPage = e.target.value;
    if (selectedPage) {
      window.location.href = selectedPage;
    }
  }}
>
```

6.Checkout Flow Component:



Product	Price	Quantity	Total
Folding Chair Color: Brown Size: XL	$120	4	$480
Armchair Tortuga Color: Brown Size: XL	$850	2	$1700
Three-Legged Shell Chair Color: Brown Size: XL	$990	1	$990
Luxury Flower Shell Sofa Chair Color: Brown Size: XL	$2500	3	$7500
Stylish Golden Metal Legs Color: Brown Size: XL	$780	2	$1560
Futuristic Sleek Modern Chair Color: Brown Size: XL	$2000	1	$2000

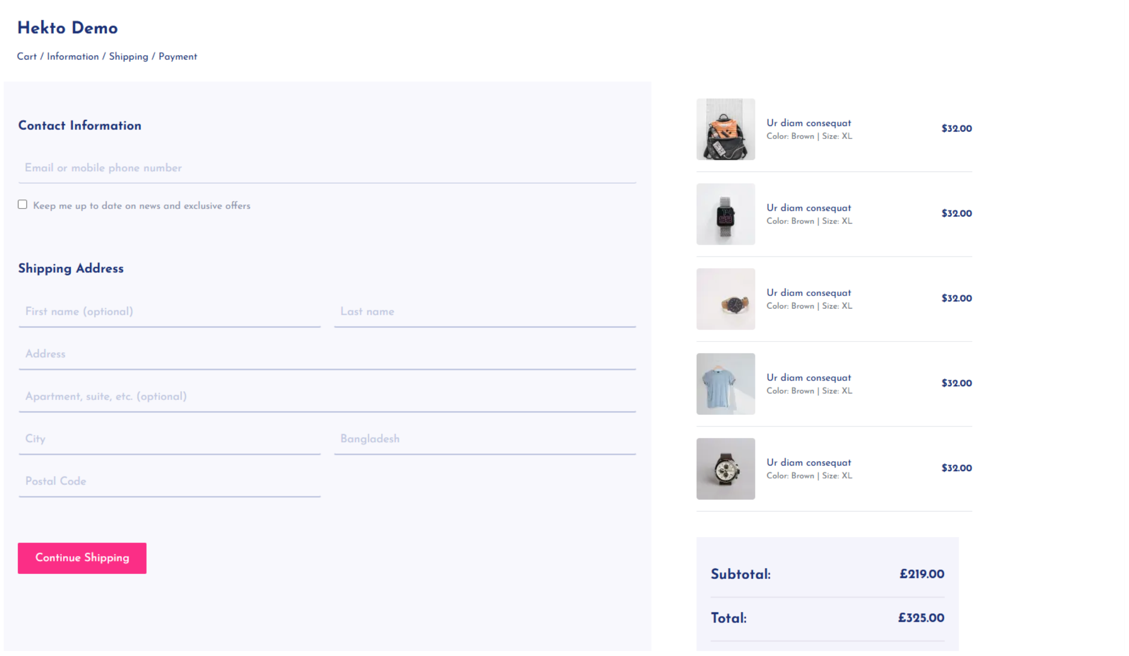
Cart Totals

Subtotal: $14230
Totals: $15668

Shipping & taxes calculated at checkout.

Calculate Shipping

User through Checjout process after clicking “proceed to checkout”.



Hekto Demo

Cart / Information / Shipping / Payment

Contact Information

Email or mobile phone number
Keep me up to date on news and exclusive offers

Shipping Address

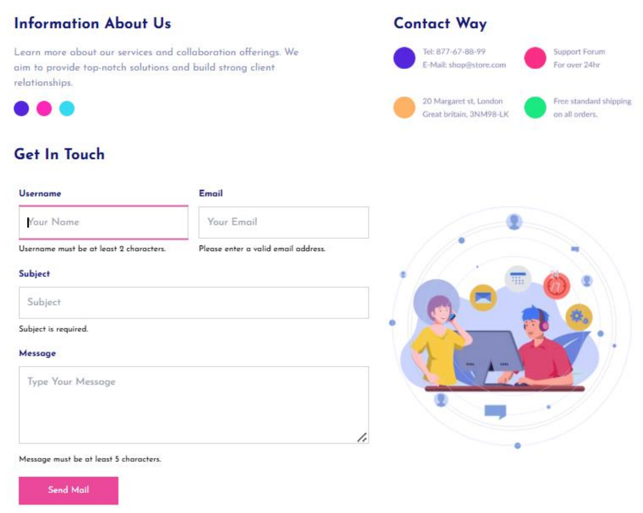
First name (optional)
Last name
Address
Apartment, suite, etc. (optional)
City
Bangladesh
Postal Code
Continue Shipping

Ur diam consequat Color: Brown | Size: XL	$32.00
Ur diam consequat Color: Brown | Size: XL	$32.00
Ur diam consequat Color: Brown | Size: XL	$32.00
Ur diam consequat Color: Brown | Size: XL	$32.00
Ur diam consequat Color: Brown | Size: XL	$32.00

Subtotal: £219.00

Total: £325.00

7.Contact Form with Validation:



Information About Us

Learn more about our services and collaboration offerings. We aim to provide top-notch solutions and build strong client relationships.

Contact Way

Tel: 877-67-88-99
E-Mail: shop@store.com

Support Forum
For over 24hr

20 Margaret st, London
Great britain, 3NM98-LK

Free standard shipping
on all orders.

Get In Touch

Username: Your Name
Username must be at least 2 characters.

Email: Your Email
Please enter a valid email address.

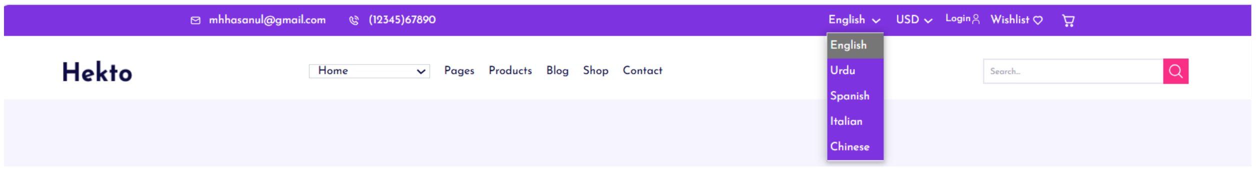
Subject: Subject
Subject is required.

Message: Type Your Message
Message must be at least 5 characters.

Without Functionality

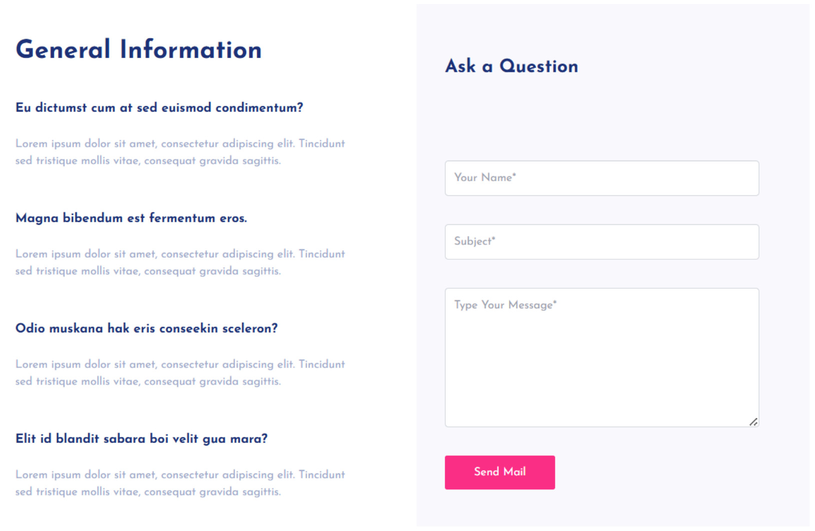
1. Multi-Language Support Component

A component to support multiple languages for better accessibility and localization.



mhhasanul@gmail.com (12345)67890

2. FAQ Component



General Information

Eu dictumst cum at sed euismod condimentum?

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Tincidunt sed tristique mollis vitae, consequat gravida sagittis.

Magna bibendum est fermentum eros.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Tincidunt sed tristique mollis vitae, consequat gravida sagittis.

Odio muskana hak eris conseekin sceleron?

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Tincidunt sed tristique mollis vitae, consequat gravida sagittis.

Elit id blandit sabara boi velit gua mara?

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Tincidunt sed tristique mollis vitae, consequat gravida sagittis.

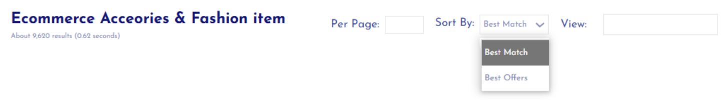
Ask a Question

Your Name*

Subject*

Type Your Message*

3.Advance Search and Filter Panel component


Ecommerce Acceories & Fashion item
About 9,620 results (0.62 seconds)
Per Page:
Sort By:
Best Match
View:
Best Match
Best Offers

Libraries Used

- **Shadcn:** For dropdowns, offering a sleek and responsive UI.
- **Shadcn Form:** Ensures robust and reliable form validation.
- **Swiper:** A library for creating smooth and interactive carousel slides.