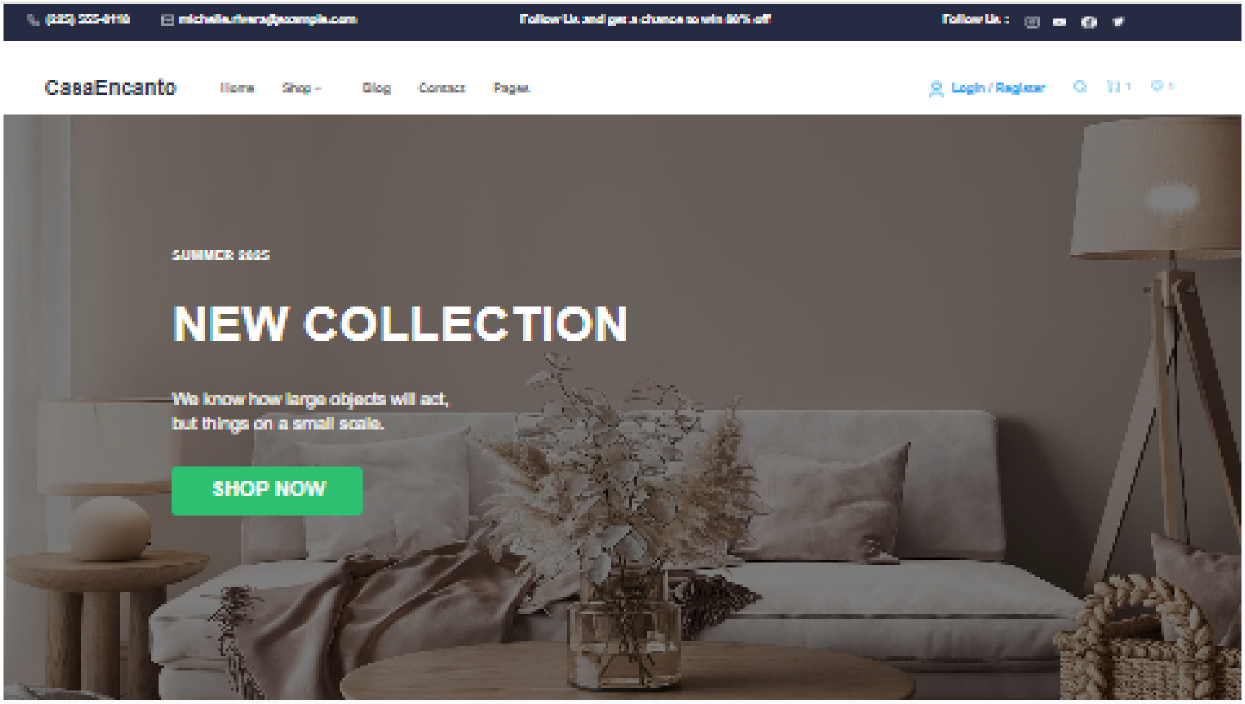


We know how large objects will act,
but things on a small scale.

Day 4 - Dynamic Frontend Components - CasaEncanto

21-01-2025

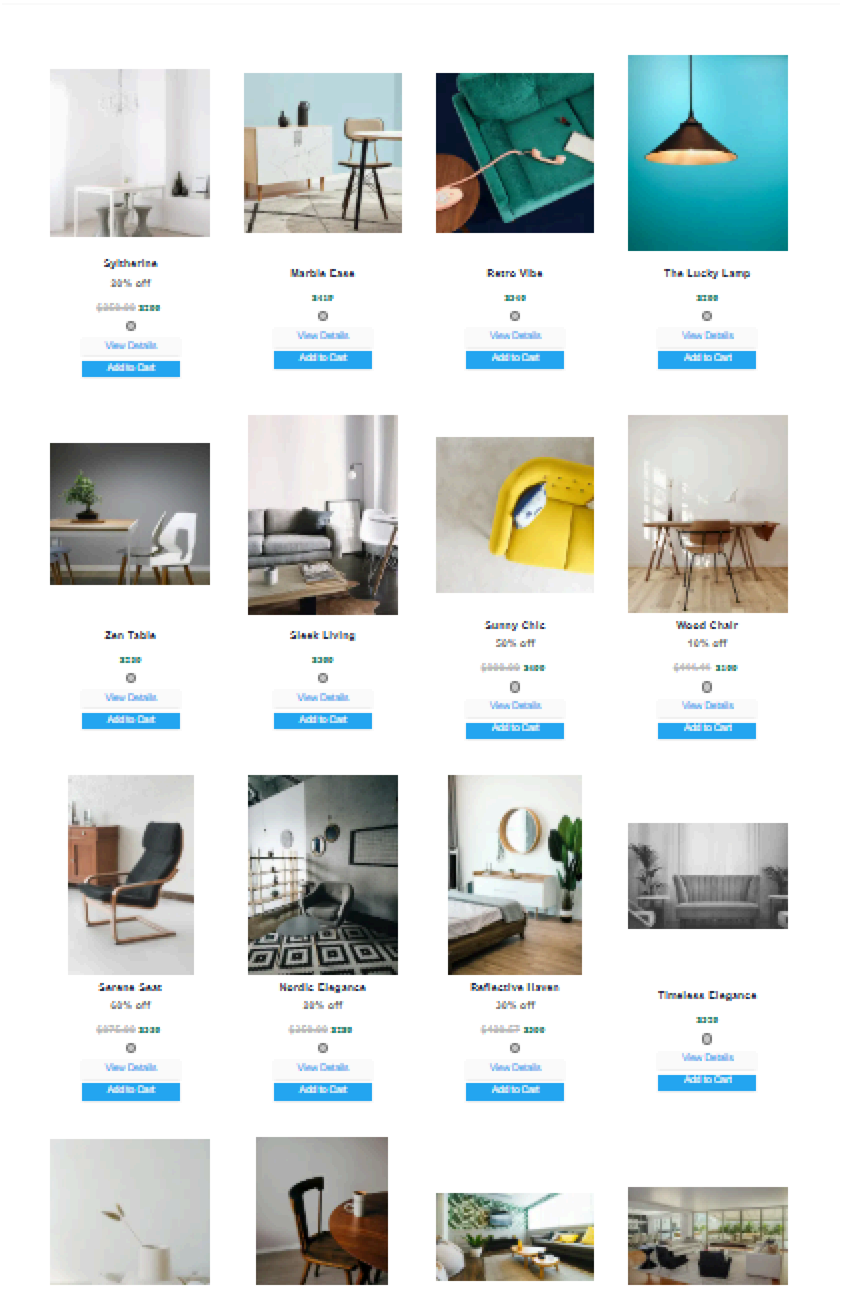
Laiba Khan

Overview

This document outlines the progress made on Day 4 of the marketplace development, focusing on dynamic frontend components. It includes functional deliverables such as product listing pages, product detail pages, category filters, search functionality, and pagination. Additionally, it provides key code snippets, API integration scripts, and a technical report summarizing the development process.

1. Functional Deliverables:

- The product listing page with dynamic data.
- Individual product detail pages with accurate routing and data rendering.
- Working category filters, search bar, and pagination.
- Any additional features implemented, such as related products or user profile components.



Sylthérine
20% off
[illegible] [illegible]
View Details
Add to Cart

Marble Case
[illegible]
View Details
Add to Cart

Retro Vibe
[illegible]
View Details
Add to Cart

The Lucky Lamp
[illegible]
View Details
Add to Cart

Zen Table
[illegible]
View Details
Add to Cart

Sleek Living
[illegible]
View Details
Add to Cart

Sunny Chic
50% off
[illegible] [illegible]
View Details
Add to Cart

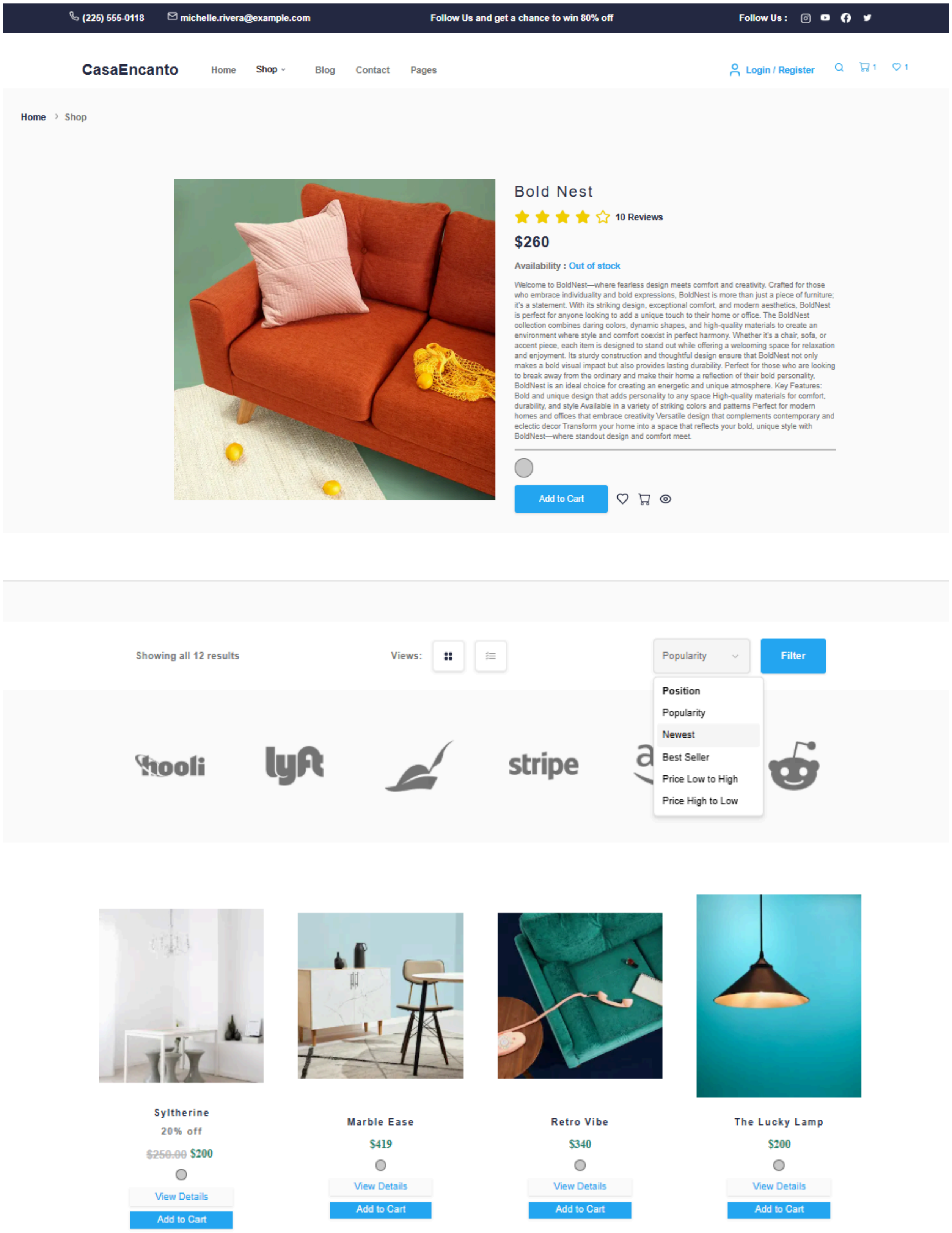
Wood Chair
10% off
[illegible] [illegible]
View Details
Add to Cart

Serene Seat
60% off
[illegible] [illegible]
View Details
Add to Cart

Nordic Elegance
30% off
[illegible] [illegible]
View Details
Add to Cart

Reflective Haven
30% off
[illegible] [illegible]
View Details
Add to Cart

Timeless Elegance
[illegible]
View Details
Add to Cart



Bold Nest

★★★★☆ 10 Reviews

$260

Availability : Out of stock

Welcome to BoldNest—where fearless design meets comfort and creativity. Crafted for those who embrace individuality and bold expressions, BoldNest is more than just a piece of furniture; it's a statement. With its striking design, exceptional comfort, and modern aesthetics, BoldNest is perfect for anyone looking to add a unique touch to their home or office. The BoldNest collection combines daring colors, dynamic shapes, and high-quality materials to create an environment where style and comfort coexist in perfect harmony. Whether it's a chair, sofa, or accent piece, each item is designed to stand out while offering a welcoming space for relaxation and enjoyment. Its sturdy construction and thoughtful design ensure that BoldNest not only makes a bold visual impact but also provides lasting durability. Perfect for those who are looking to break away from the ordinary and make their home a reflection of their bold personality, BoldNest is an ideal choice for creating an energetic and unique atmosphere. Key Features: Bold and unique design that adds personality to any space High-quality materials for comfort, durability, and style Available in a variety of striking colors and patterns Perfect for modern homes and offices that embrace creativity Versatile design that complements contemporary and eclectic decor Transform your home into a space that reflects your bold, unique style with BoldNest—where standout design and comfort meet.

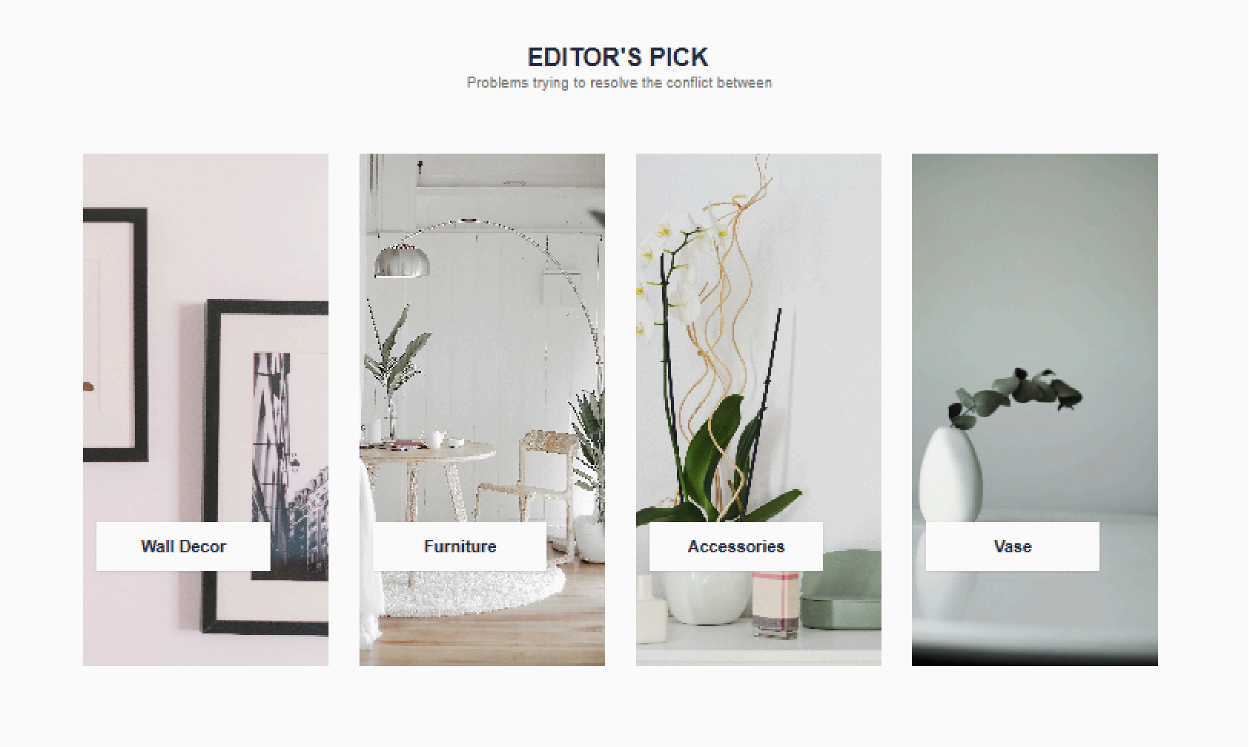
Showing all 12 results

Syltherine
20% off
~~$250.00~~ $200

Marble Ease
$419

Retro Vibe
$340

The Lucky Lamp
$200

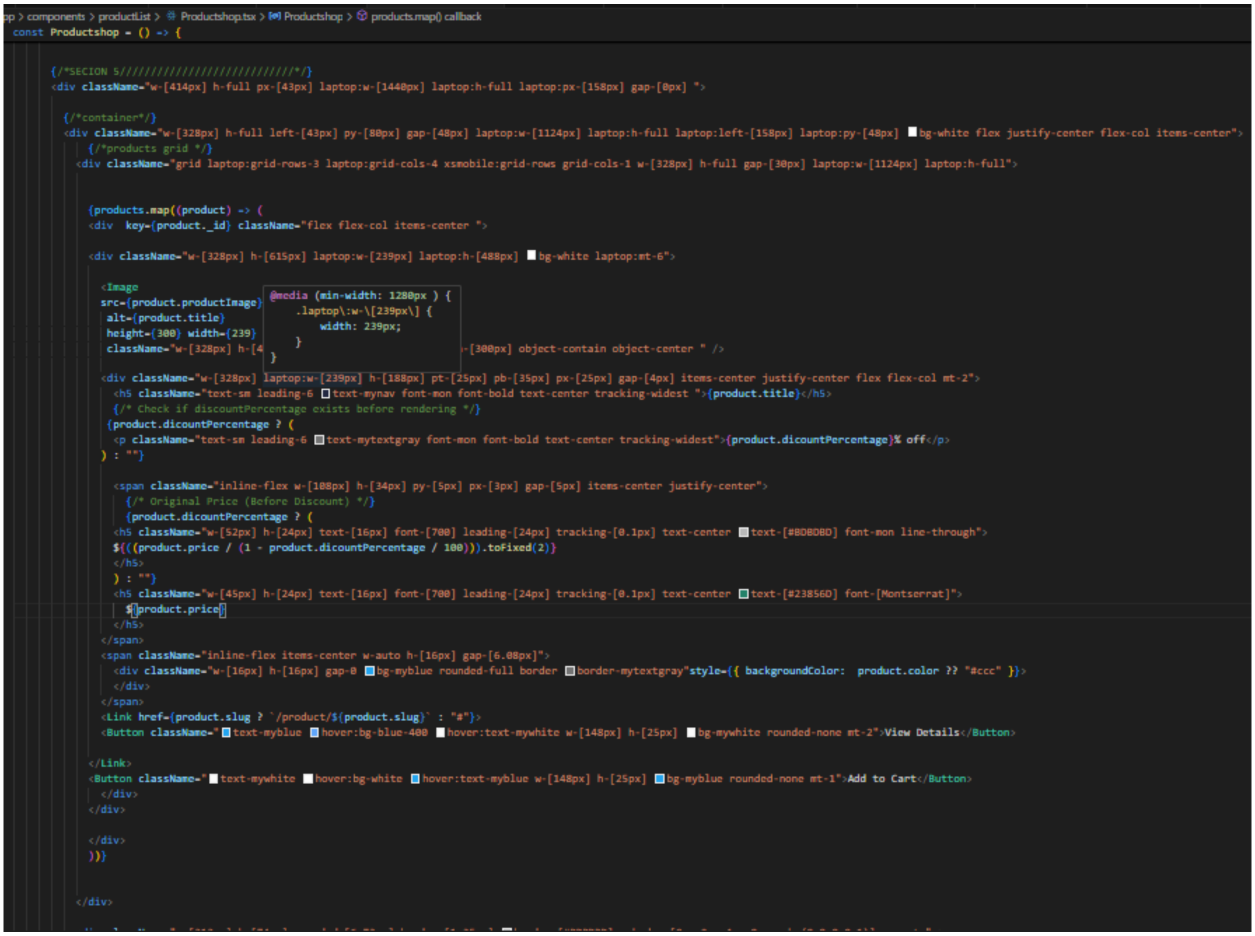


EDITOR'S PICK

Problems trying to resolve the conflict between

2. Code Deliverables:

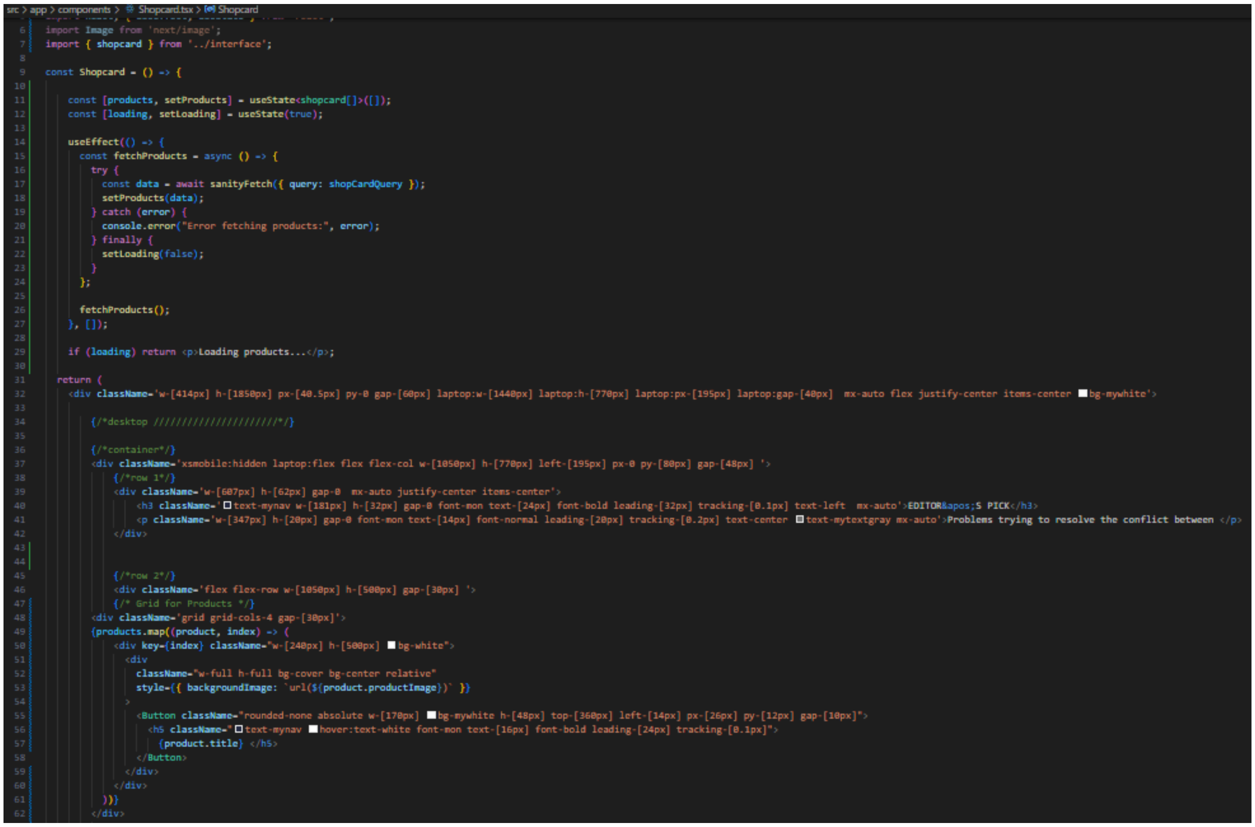
Code snippets for key components (e.g., ProductCard, ProductList, SearchBar).



pp > components > productList > Productshop.tsx > Productshop > products.map() callback

```
const Productshop = () => {

      {/*SECION 5/////////////////////////////*/}
      <div className="w-[414px] h-full px-[43px] laptop:w-[1440px] laptop:h-full laptop:px-[158px] gap-[0px] ">

        {/*container*/}
        <div className="w-[328px] h-full left-[43px] py-[80px] gap-[48px] laptop:w-[1124px] laptop:h-full laptop:left-[158px] laptop:py-[48px] bg-white flex justify-center flex-col items-center">
            {/*products grid */}
          <div className="grid laptop:grid-rows-3 laptop:grid-cols-4 xsmobile:grid-rows grid-cols-1 w-[328px] h-full gap-[30px] laptop:w-[1124px] laptop:h-full">


            {products.map((product) => (
            <div  key={product._id} className="flex flex-col items-center ">

            <div className="w-[328px] h-[615px] laptop:w-[239px] laptop:h-[488px] bg-white laptop:mt-6">

              <Image
              src={product.productImage}
               alt={product.title}
               height={300} width={239}
               className="w-[328px] h-[4                                   -[300px] object-contain object-center " />

              <div className="w-[328px] laptop:w-[239px] h-[188px] pt-[25px] pb-[35px] px-[25px] gap-[4px] items-center justify-center flex flex-col mt-2">
                <h5 className="text-sm leading-6 text-mynav font-mon font-bold text-center tracking-widest ">{product.title}</h5>
                {/* Check if discountPercentage exists before rendering */}
               {product.dicountPercentage ? (
                <p className="text-sm leading-6 text-mytextgray font-mon font-bold text-center tracking-widest">{product.dicountPercentage}% off</p>
              ) : ""}

                <span className="inline-flex w-[108px] h-[34px] py-[5px] px-[3px] gap-[5px] items-center justify-center">
                  {/* Original Price (Before Discount) */}
                  {product.dicountPercentage ? (
                <h5 className="w-[52px] h-[24px] text-[16px] font-[700] leading-[24px] tracking-[0.1px] text-center text-[#BDBDBD] font-mon line-through">
                ${((product.price / (1 - product.dicountPercentage / 100))).toFixed(2)}
                </h5>
                ) : ""}
                <h5 className="w-[45px] h-[24px] text-[16px] font-[700] leading-[24px] tracking-[0.1px] text-center text-[#23856D] font-[Montserrat]">
                  ${product.price}
                </h5>
              </span>
              <span className="inline-flex items-center w-auto h-[16px] gap-[6.08px]">
                <div className="w-[16px] h-[16px] gap-0 bg-myblue rounded-full border border-mytextgray"style={{ backgroundColor:  product.color ?? "#ccc" }}>
                </div>
              </span>
              <Link href={product.slug ? `/product/${product.slug}` : "#"}>
              <Button className=" text-myblue hover:bg-blue-400 hover:text-mywhite w-[148px] h-[25px] bg-mywhite rounded-none mt-2">View Details</Button>

            </Link>
            <Button className=" text-mywhite hover:bg-white hover:text-myblue w-[148px] h-[25px] bg-myblue rounded-none mt-1">Add to Cart</Button>
              </div>
            </div>

            </div>
            ))}


          </div>
```

```
@media (min-width: 1280px ) {
    .laptop\:w-\[239px\] {
        width: 239px;
    }
}
```

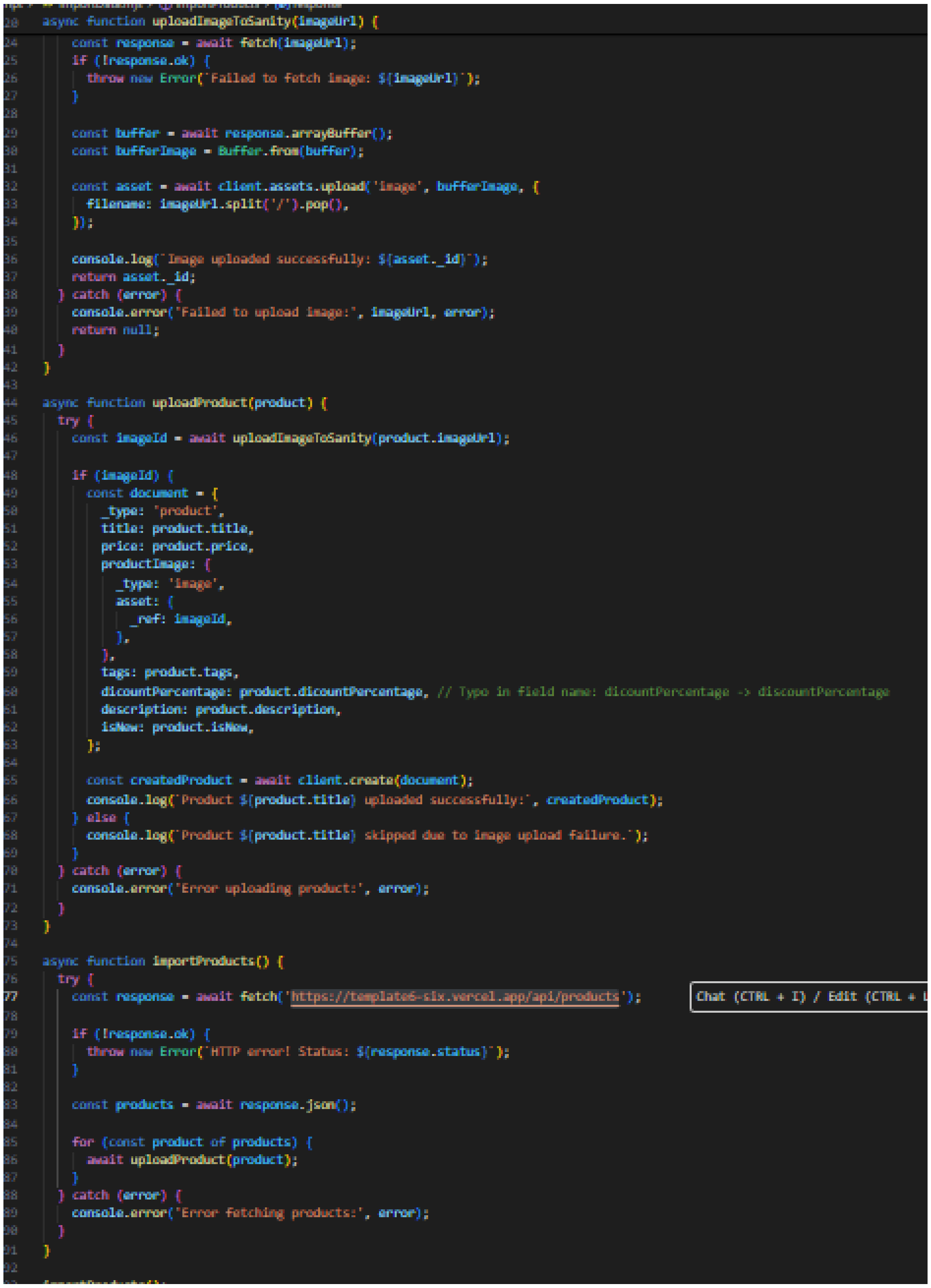


src > app > components > Shopcard.tsx > Shopcard

```
import Image from 'next/image';
import { shopcard } from '../interface';

const Shopcard = () => {

    const [products, setProducts] = useState<shopcard[]>([]);
    const [loading, setLoading] = useState(true);

    useEffect(() => {
      const fetchProducts = async () => {
        try {
          const data = await sanityFetch({ query: shopCardQuery });
          setProducts(data);
        } catch (error) {
          console.error("Error fetching products:", error);
        } finally {
          setLoading(false);
        }
      };

      fetchProducts();
    }, []);

    if (loading) return <p>Loading products...</p>;

  return (
    <div className='w-[414px] h-[1850px] px-[40.5px] py-0 gap-[60px] laptop:w-[1440px] laptop:h-[770px] laptop:px-[195px] laptop:gap-[40px]  mx-auto flex justify-center items-center bg-mywhite'>

        {/*desktop //////////////////////*/}

        {/*container*/}
        <div className='xsmobile:hidden laptop:flex flex flex-col w-[1050px] h-[770px] left-[195px] px-0 py-[80px] gap-[48px] '>
            {/*row 1*/}
            <div className='w-[607px] h-[62px] gap-0  mx-auto justify-center items-center'>
                <h3 className='text-mynav w-[181px] h-[32px] gap-0 font-mon text-[24px] font-bold leading-[32px] tracking-[0.1px] text-left  mx-auto'>EDITOR&apos;S PICK</h3>
                <p className='w-[347px] h-[20px] gap-0 font-mon text-[14px] font-normal leading-[20px] tracking-[0.2px] text-center text-mytextgray mx-auto'>Problems trying to resolve the conflict between </p>
            </div>


            {/*row 2*/}
            <div className='flex flex-row w-[1050px] h-[500px] gap-[30px] '>
            {/* Grid for Products */}
        <div className='grid grid-cols-4 gap-[30px]'>
        {products.map((product, index) => (
            <div key={index} className="w-[240px] h-[500px] bg-white">
              <div
                className="w-full h-full bg-cover bg-center relative"
                style={{ backgroundImage: `url(${product.productImage})` }}
              >
                <Button className="rounded-none absolute w-[170px] bg-mywhite h-[48px] top-[360px] left-[14px] px-[26px] py-[12px] gap-[10px]">
                  <h5 className="text-mynav hover:text-white font-mon text-[16px] font-bold leading-[24px] tracking-[0.1px]">
                   {product.title} </h5>
                </Button>
              </div>
            </div>
          ))}
        </div>
```

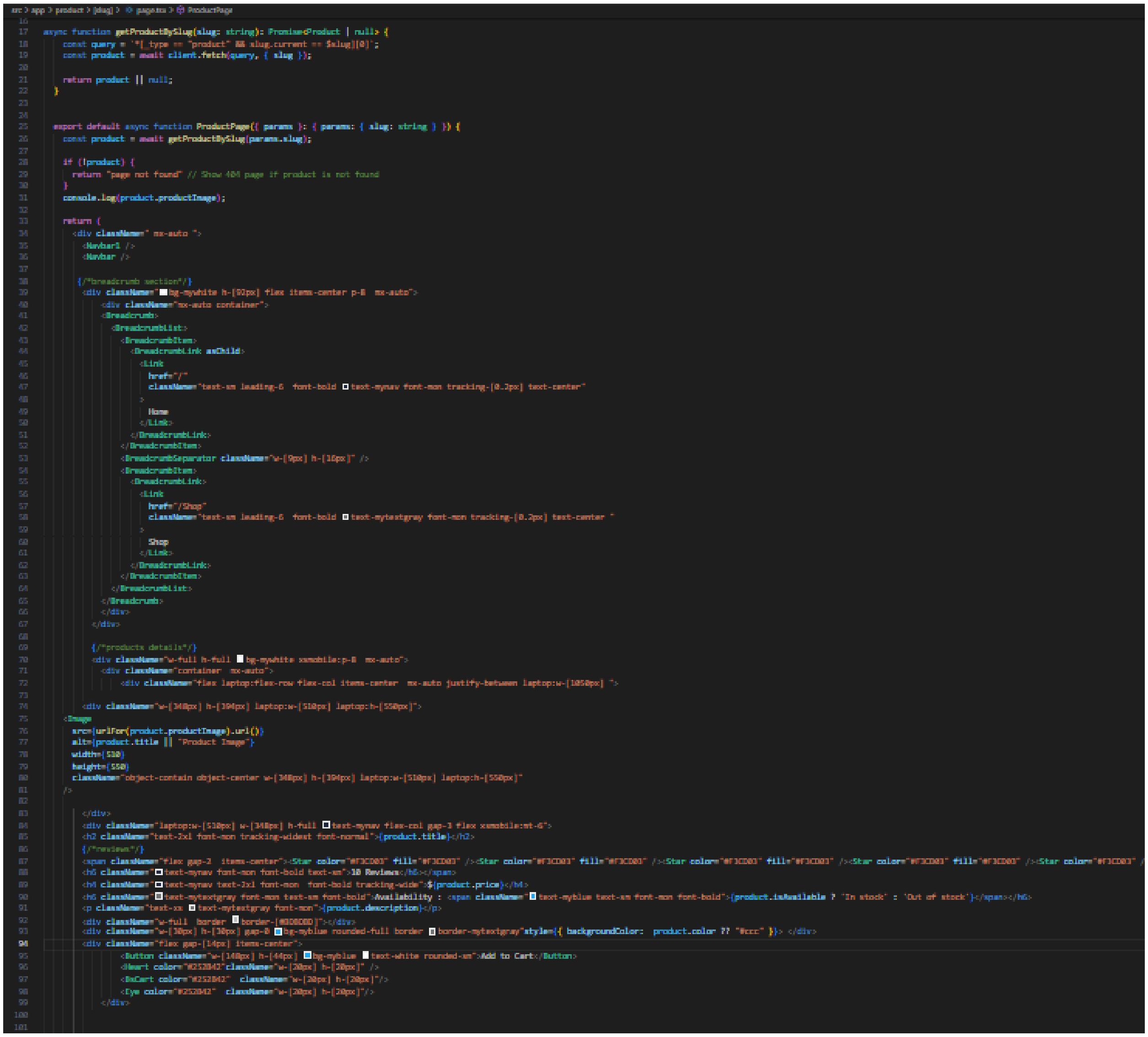
Scripts or logic for API integration and dynamic routing.

API integration:



```
async function uploadImageToSanity(imageUrl) {
    const response = await fetch(imageUrl);
    if (!response.ok) {
      throw new Error(`Failed to fetch image: ${imageUrl}`);
    }

    const buffer = await response.arrayBuffer();
    const bufferImage = Buffer.from(buffer);

    const asset = await client.assets.upload('image', bufferImage, {
      filename: imageUrl.split('/').pop(),
    });

    console.log(`Image uploaded successfully: ${asset._id}`);
    return asset._id;
  } catch (error) {
    console.error('Failed to upload image:', imageUrl, error);
    return null;
  }
}

async function uploadProduct(product) {
  try {
    const imageId = await uploadImageToSanity(product.imageUrl);

    if (imageId) {
      const document = {
        _type: 'product',
        title: product.title,
        price: product.price,
        productImage: {
          _type: 'image',
          asset: {
            _ref: imageId,
          },
        },
        tags: product.tags,
        dicountPercentage: product.dicountPercentage, // Typo in field name: dicountPercentage -> discountPercentage
        description: product.description,
        isNew: product.isNew,
      };

      const createdProduct = await client.create(document);
      console.log(`Product ${product.title} uploaded successfully:`, createdProduct);
    } else {
      console.log(`Product ${product.title} skipped due to image upload failure.`);
    }
  } catch (error) {
    console.error('Error uploading product:', error);
  }
}

async function importProducts() {
  try {
    const response = await fetch('https://template6-six.vercel.app/api/products');

    if (!response.ok) {
      throw new Error(`HTTP error! Status: ${response.status}`);
    }

    const products = await response.json();

    for (const product of products) {
      await uploadProduct(product);
    }
  } catch (error) {
    console.error('Error fetching products:', error);
  }
}
```

Dynamic Routing:



src > app > product > [slug] > page.tsx > ProductPage

```
async function getProductBySlug(slug: string): Promise<Product | null> {
    const query = `*[_type == "product" && slug.current == $slug][0]`;
    const product = await client.fetch(query, { slug });

    return product || null;
  }


  export default async function ProductPage({ params }: { params: { slug: string } }) {
    const product = await getProductBySlug(params.slug);

    if (!product) {
      return "page not found" // Show 404 page if product is not found
    }
    console.log(product.productImage);

    return (
      <div className=" mx-auto ">
        <Navbar1 />
        <Navbar />

        {/*breadcrumb section*/}
        <div className="bg-mywhite h-[92px] flex items-center p-8  mx-auto">
            <div className="mx-auto container">
            <Breadcrumb>
              <BreadcrumbList>
                <BreadcrumbItem>
                  <BreadcrumbLink asChild>
                    <Link
                      href="/"
                      className="text-sm leading-6  font-bold text-mynav font-mon tracking-[0.2px] text-center"
                    >
                      Home
                    </Link>
                  </BreadcrumbLink>
                </BreadcrumbItem>
                <BreadcrumbSeparator className="w-[9px] h-[16px]" />
                <BreadcrumbItem>
                  <BreadcrumbLink>
                    <Link
                      href="/Shop"
                      className="text-sm leading-6  font-bold text-mytextgray font-mon tracking-[0.2px] text-center "
                    >
                      Shop
                    </Link>
                  </BreadcrumbLink>
                </BreadcrumbItem>
              </BreadcrumbList>
            </Breadcrumb>
            </div>
          </div>

          {/*products details*/}
          <div className="w-full h-full bg-mywhite xsmobile:p-8  mx-auto">
            <div className="container  mx-auto">
                <div className="flex laptop:flex-row flex-col items-center  mx-auto justify-between laptop:w-[1050px] ">

        <div className="w-[348px] h-[394px] laptop:w-[510px] laptop:h-[550px]">
      <Image
        src={urlFor(product.productImage).url()}
        alt={product.title || "Product Image"}
        width={510}
        height={550}
        className="object-contain object-center w-[348px] h-[394px] laptop:w-[510px] laptop:h-[550px]"
      />

        </div>
        <div className="laptop:w-[510px] w-[348px] h-full text-mynav flex-col gap-3 flex xsmobile:mt-6">
        <h2 className="text-2xl font-mon tracking-widest font-normal">{product.title}</h2>
        {/*reviews*/}
        <span className="flex gap-2  items-center"><Star color="#F3CD03" fill="#F3CD03" /><Star color="#F3CD03" fill="#F3CD03" /><Star color="#F3CD03" fill="#F3CD03" /><Star color="#F3CD03" fill="#F3CD03" /><Star color="#F3CD03" /
        <h6 className="text-mynav font-mon font-bold text-sm">10 Reviews</h6></span>
        <h4 className="text-mynav text-2xl font-mon  font-bold tracking-wide">${product.price}</h4>
        <h6 className="text-mytextgray font-mon text-sm font-bold">Availability : <span className="text-myblue text-sm font-mon font-bold">{product.isAvailable ? 'In stock' : 'Out of stock'}</span></h6>
        <p className="text-xs text-mytextgray font-mon">{product.description}</p>
        <div className="w-full  border border-[#BDBDBD]"></div>
        <div className="w-[30px] h-[30px] gap-8 bg-myblue rounded-full border border-mytextgray"style={{ backgroundColor:  product.color ?? "#ccc" }}> </div>
        <div className="flex gap-[14px] items-center">
                <Button className="w-[148px] h-[44px] bg-myblue text-white rounded-sm">Add to Cart</Button>
                <Heart color="#252B42"className="w-[20px] h-[20px]" />
                <BsCart color="#252B42"  className="w-[20px] h-[20px]"/>
                <Eye color="#252B42"  className="w-[20px] h-[20px]"/>
            </div>
```

Technical Report

Steps Taken to Build and Integrate Components

I. Data Fetching & API Integration:

- Integrated Sanity as the backend CMS for managing product data.
- Used Next.js's `getServerSideProps` or `getStaticProps` to fetch and display product data dynamically.
- Implemented API routes to handle fetching filtered and paginated data.

2. Component Development:

- Designed `ProductCard` to display product information dynamically.
- Developed `ProductList` to fetch and render multiple products.
- Implemented `SearchBar` and category filters for enhanced user experience.

3. Dynamic Routing:

- Configured Next.js dynamic routes (`app/product/[slug].tsx`) to display individual product details.
- Utilized `slug` from Sanity to ensure proper URL structuring.

4. Challenges Faced and Solutions Implemented

- **Issue:** Slug missing in some products
  - *Solution:* Ensured that all products in Sanity had a unique `slug` field.
- **Issue:** Image URLs from Sanity not rendering properly
  - *Solution:* Used `@sanity/image-url` to correctly format image URLs.
- **Issue:** Search and filter functionalities causing performance issues
  - *Solution:* Implemented debounce logic to optimize search performance

5. Best Practices Followed During Development

- **Code Optimization:**
  - Used reusable components to maintain clean and scalable code.
  - Followed Next.js conventions for optimal performance.
- **Security Measures:**
  - Implemented input validation for search and filters.
  - Sanitized API responses to prevent potential vulnerabilities.
- **User Experience Improvements:**
  - Ensured fast page loading by optimizing images and API calls.
  - Maintained responsive design for mobile and desktop views.

Conclusion

The implementation of dynamic frontend components has significantly improved the user experience and functionality of our marketplace. By integrating dynamic product listings, filters, search functionality, and optimized routing, we have built a seamless and interactive platform. However, there is always room for improvement, and I plan to refine the UI, enhance performance, and introduce more advanced features in future iterations.