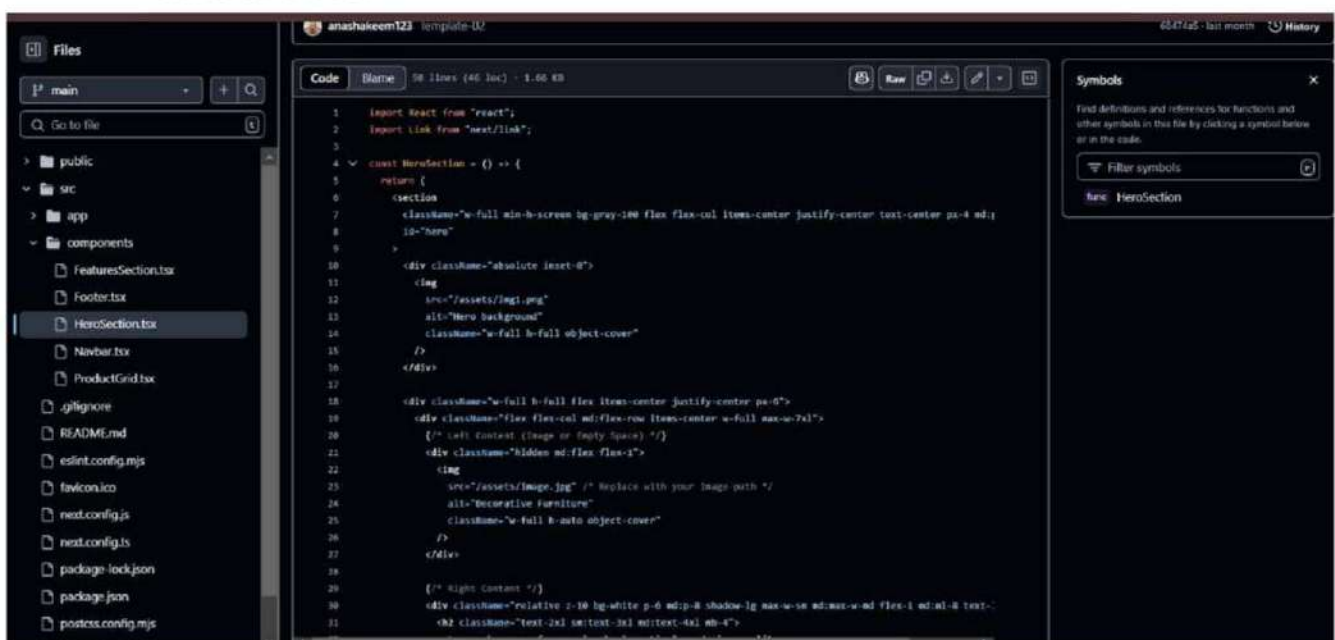


DAY 4 - BUILDING DYNAMIC FRONTEND COMPONENTS FOR YOUR MARKETPLACE

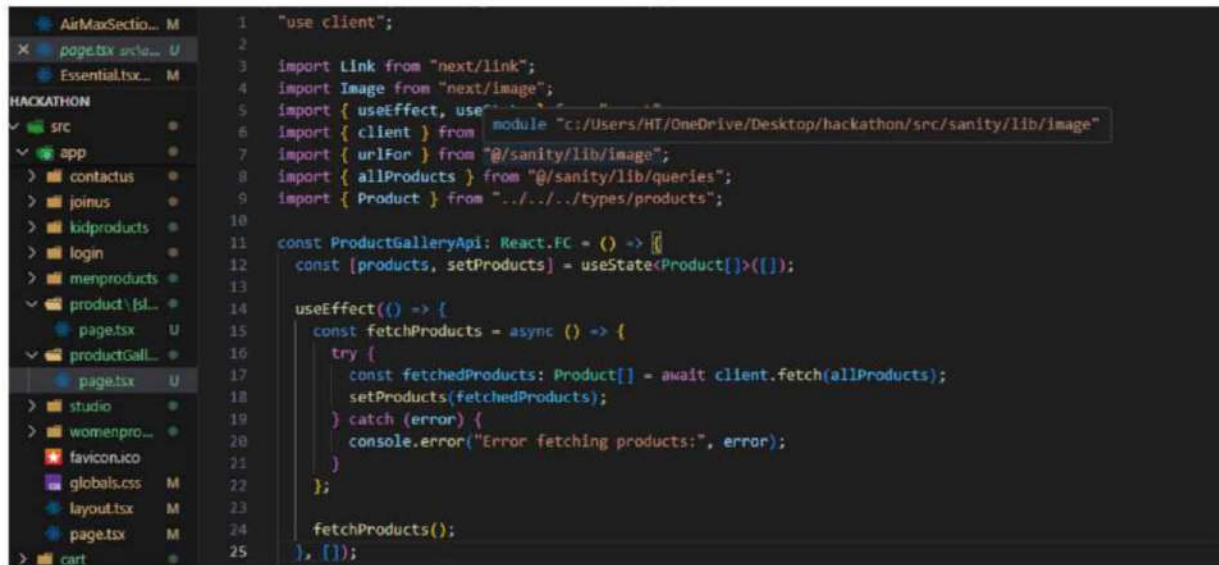
- Document Title: "Day 4 - Dynamic Frontend Components - Template 02 Avion "

Key Learning Outcomes:

1. Build dynamic frontend components to display data from Sanity CMS or APIs.



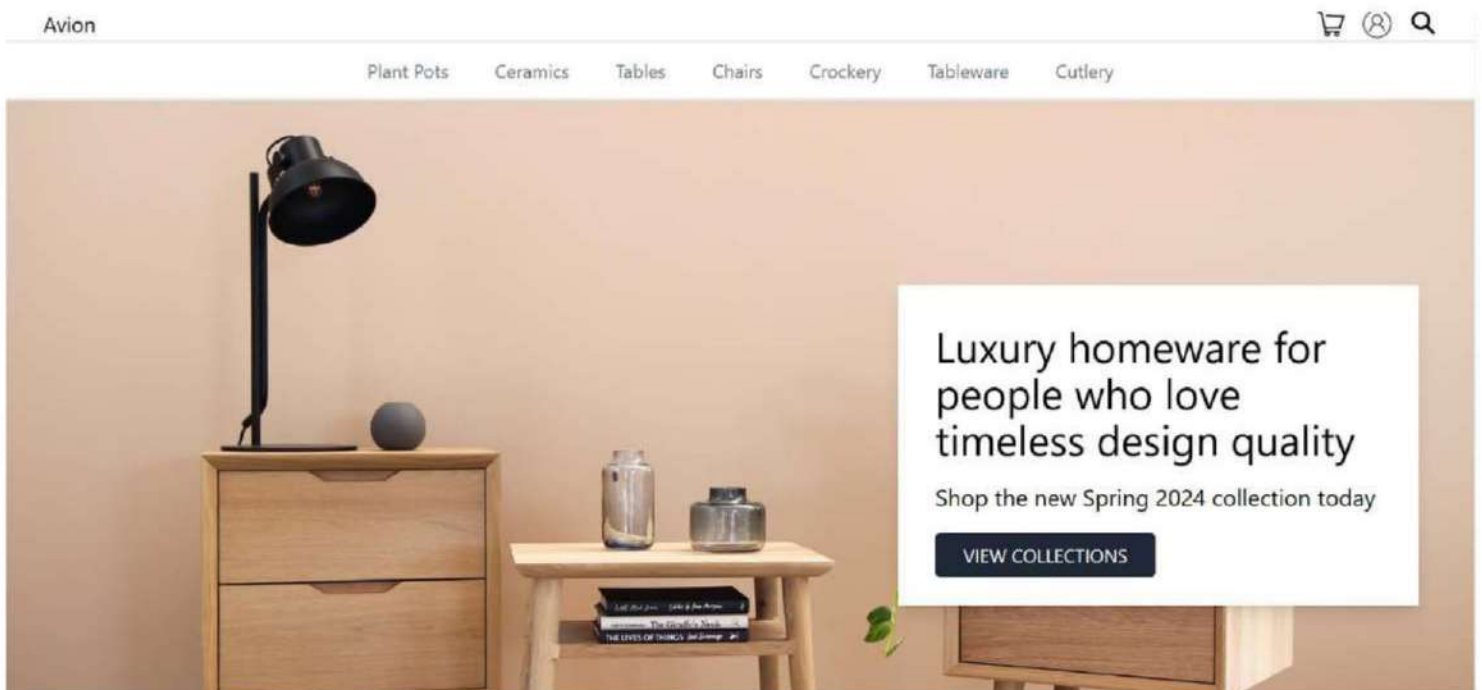
2. Implement reusable and modular components.
3. Understand and apply state management techniques.



```
1 "use client";
2
3 import Link from "next/link";
4 import Image from "next/image";
5 import { useEffect, use
6 import { client } from module "c:/Users/HT/OneDrive/Desktop/hackathon/src/sanity/lib/image"
7 import { urlFor } from "@sanity/lib/image";
8 import { allProducts } from "@sanity/lib/queries";
9 import { Product } from "../../types/products";
10
11 const ProductGalleryApi: React.FC = () => {
12   const [products, setProducts] = useState<Product[]>([]);
13
14   useEffect(() => {
15     const fetchProducts = async () => {
16       try {
17         const fetchedProducts: Product[] = await client.fetch(allProducts);
18         setProducts(fetchedProducts);
19       } catch (error) {
20         console.error("Error fetching products:", error);
21       }
22     };
23
24     fetchProducts();
25   }, []);
26 }
```

4. Learn the importance of responsive design and UX/UI best practices.
5. 5. Prepare for real-world client projects by replicating professional workflows.

Laptop Screen RESPONSIVE



Mobile Screen Responsive



Key Components to Build

1. Product Listing Component:

- Render product data dynamically in a grid layout.

```
/* Product Grid */
<main className="w-full md:w-3/4 grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 md:gap-6">
  {products.map((product) => {
    <div
      key={product._id}
      className="bg-white p-2 relative group transition-all duration-300 ease-in-out group-hover:bg-gray-200 group-hover:shadow"
    >
      <Link href={` /product/${product.slug?.current}`} >
        {product.image ? (
          <Image
            src={urlFor(product.image).url()}
            alt={product.productName}
            width={348}
            height={348}
            className="md:w-[348px] md:h-[348px] sm:w-[128px] sm:h-[128px] object-cover mb-4 transition-transform duration-300 ease-in-out"
          />
        ) : null}
      </Link>
      <p className="text-orange-600 sm:text-[12px] md:text-lg font-semibold">
        {product.status}
      </p>
      <h3 className="sm:text-[8px] md:text-sm text-black">
        {product.productName}
      </h3>
    </div>
  )}
</main>
```

- Include fields like:

```
src > sanity > schemaTypes > TS product.ts > productSchema > fields
1  export const productSchema = {
2    name: 'product',
3    title: 'Product',
4    type: 'document',
5    fields: [
6      {
7        name: 'productName',
8        title: 'Product Name',
9        type: 'string',
10       },
11      {
12        name: 'category',
13        title: 'Category',
14        type: 'string',
15       },
16      {
17        name: 'slug',
18        title: 'Slug',
19        type: 'slug',
20        options: {
21          source: 'productName',
22        },
23      },
24    ],
25  },
26  {
27    name: 'price',
28    title: 'Price',
29    type: 'number',
30  },
31  },
32  }
```

```
/* Product Grid */
<main className="w-full md:w-3/4 grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 md:gap-6">
  {products.map((product) => {
    <div
      key={product._id}
      className="bg-white p-2 relative group transition-all duration-300 ease-in-out group-hover:bg-gray-200 group-hover:shadow"
    >
      <Link href={` /product/${product.slug?.current}`} >
        {product.image ? (
          <Image
            src={urlFor(product.image).url()}
            alt={product.productName}
            width={348}
            height={348}
            className="md:w-[348px] md:h-[348px] sm:w-[128px] sm:h-[128px] object-cover mb-4 transition-transform duration-300 ease-in-out"
          />
        ) : null}
      </Link>
      <p className="text-orange-600 sm:text-[12px] md:text-lg font-semibold">
        {product.status}
      </p>
      <h3 className="sm:text-[8px] md:text-sm text-black">
        {product.productName}
      </h3>
    </div>
  })}
</main>
```

- Include fields like:

```
src > sanity > schemaTypes > TS product.ts > productSchema > fields
1  export const productSchema = {
2    name: 'product',
3    title: 'Product',
4    type: 'document',
5    fields: [
6      {
7        name: 'productName',
8        title: 'Product Name',
9        type: 'string',
10      },
11      {
12        name: 'category',
13        title: 'Category',
14        type: 'string',
15      },
16      {
17        name: 'slug',
18        title: 'Slug',
19        type: 'slug',
20        options: {
21          source: 'productName',
22        },
23      },
24    ],
25  },
26  {
27    name: 'price',
28    title: 'Price',
29    type: 'number',
30  },
31  },
32  }
```


- Example layout: cards displaying product details.

```
const Carousel: React.FC = () => {
  <div className="flex space-x-4 overflow-x-auto sm:overflow-hidden">
    {visibleProducts.map((product) => {
      <div
        key={product._id}
        className="flex-shrink-0 w-full sm:w-1/2 md:w-1/3 bg-white p-4 text-center rounded-lg shadow ho
      >
        <Link href={` /product/${product.slug?.current}`}>
          <Image
            src={urlFor(product.image).url()}
            alt={product.productName}
            width={400}
            height={400}
            className="mx-auto object-cover mb-4 rounded-lg"
          />
        </Link>
        <div className="md:px-10 sm:text-[8px] md:text-sm flex sm:flex-col md:flex-row items-center just
          <div className="text-start flex sm:items-center md:items-start flex-col" >
            <h3 className="text-black font-medium">
              {product.productName}
            </h3>
            <p className="text-gray-600 sm:text-start md:text-start">{product.category}</p>
          </div>
          <div>
            <p className="text-black">MRP : ₹ {product.price}</p>
          </div>
        </div>
      </div>
    })}
  </div>
}
```

2. Product Detail Component:

- Create individual product detail pages using dynamic routing in Next.js.
- Include detailed fields such as:
 - o Product Description
 - o Price
 - o Available Sizes or Colors