

## Tóm tắt các bước chỉnh sửa lọc theo danh mục + phân trang 4 sản phẩm/trang đúng với cấu trúc dự án

### ✅ 1. productsApi.js

**Mục đích:** Tách hàm gọi API riêng.

js

Sao chépChỉnh sửa

```
// src/api/productsApi.js
```

```
import axios from "axios";
```

```
export const getAllProducts = () => {  
  return axios.get("https://fakestoreapi.com/products");  
};
```

```
export const getAllProductsByCategory = (category) => {  
  return axios.get(`https://fakestoreapi.com/products/category/${category}`);  
};
```

---

### ✅ 2. ReactHeaderTemplates.jsx


**Mục đích:** Khi nhấn vào menu, đổi URL để lọc sản phẩm qua category, có thêm All Products.

**Đã thêm:**

js

Sao chépChỉnh sửa

```
const handleCategoryClick = (value) => {  
  navigate(`/?category=${encodeURIComponent(value)}&page=1`);  
};
```

```
const menuCategories = [
  { label: "All Products", value: "" }, //  thêm
  { label: "Men", value: "men's clothing" },
  { label: "Women", value: "women's clothing" },
  { label: "Jewelry", value: "jewelery" },
  { label: "Electronics", value: "electronics" },
];
```

**Và sửa map() để dùng key đúng + gọi handleClick:**

jsx

Sao chépChỉnh sửa

```
{menuCategories.map((category) => (
  <button
    key={category.value ?? "all"}
    onClick={() => handleClick(category.value)}
    className="btn btn-link text-secondary"
  >
    {category.label}
  </button>
))}
```

---

### 3. ResponsiveProductsGrid.jsx

**Mục đích:**

- Đọc category và page từ useSearchParams.
- Lọc sản phẩm theo danh mục.
- Phân trang hiển thị 4 sản phẩm/trang.

js

Sao chépChỉnh sửa

```
const [allProducts, setAllProducts] = useState([]);  
const [products, setProducts] = useState([]);  
const [currentPage, setCurrentPage] = useState(1);  
const itemsPerPage = 4;  
  
const [searchParams, setSearchParams] = useSearchParams();  
const category = searchParams.get("category");  
const page = parseInt(searchParams.get("page")) || 1;
```

**Cập nhật trong useEffect:**

js

Sao chépChỉnh sửa

```
useEffect(() => {  
  const fetchData = async () => {  
    try {  
      let res;  
      if (category) {  
        res = await getAllProductsByCategory(category);  
      } else {  
        res = await getAllProducts();  
      }  
      setAllProducts(res.data); // lưu full list  
      const startIndex = (page - 1) * itemsPerPage;  
      const paginatedProducts = res.data.slice(startIndex, startIndex + itemsPerPage);  
      setProducts(paginatedProducts);  
    }  
  }  
});
```

```

    setCurrentPage(page);
  } catch (err) {
    console.error("Lỗi khi gọi API:", err);
  }
};

fetchData();
}, [category, page]);

```

### Hiển thị sản phẩm:

jsx

Sao chépChỉnh sửa

```

{products.map((product) => (
  <div key={product.id} onClick={() => handleProductClick(product)} style={{ cursor: "pointer" }}>
    <Product {...product} />
  </div>
))}

```

### Hiển thị phân trang:

jsx

Sao chépChỉnh sửa

```

<Pagination
  totalItems={allProducts.length}
  itemsPerPage={itemsPerPage}
  currentPage={currentPage}
  onPageChange={(page) =>
    setSearchParams((prev) => {
      prev.set("page", page);
      return prev;
    })
  }
/>

```

```
  })  
}  
/>
```

---

#### ✅ 4. Pagination.jsx (nếu có)

Hiển thị các nút phân trang theo currentPage và totalItems.

---

#### 💡 Gợi ý thêm:

- Có thể thêm Loading... khi gọi API.
- Có thể làm "**No products found**" nếu không có sản phẩm.
- Sau này ghép thêm search, sort, price filter → gắn vào URL luôn.