1. Component
2. FilterSortBar.tsx

export default function FilterSortBar({

onFilter,

onSort,

}: {

onFilter: (cat: string) => void;

onSort: (by: string) => void;

}) {

return (

<div className="flex flex-col md:flex-row gap-4 mb-4">

<select onChange={(e) => onFilter(e.target.value)} className="p-2 border rounded">

<option value="">Semua Kategori</option>

<option value="men's clothing">Pakaian Pria</option>

<option value="women's clothing">Pakaian Wanita</option>

<option value="electronics">Elektronik</option>

<option value="jewelery">Perhiasan</option>

</select>

<select onChange={(e) => onSort(e.target.value)} className="p-2 border rounded">

<option value="">Urutkan</option>

<option value="price-asc">Harga Terendah</option>

<option value="price-desc">Harga Tertinggi</option>

<option value="rating">Rating Tertinggi</option>

<option value="stock">Stok Terbanyak</option>

</select>

</div>

);

}

1. Navbar.tsx

import Link from "next/link";

import { useContext } from "react";

import { ThemeContext } from "../context/ThemeContext";

export default function Navbar() {

const { theme, toggle } = useContext(ThemeContext);

return (

<nav className="bg-gray-100 dark:bg-gray-900 text-black dark:text-white p-4 flex justify-between">

<Link href="/" className="font-bold text-xl">🛍️ E-Shop</Link>

<div className="flex gap-4">

<Link href="/cart">Cart</Link>

<button onClick={toggle}>

{theme === "light" ? "🌙" : "☀️"}

</button>

</div>

</nav>

);

}

1. ProductCard.tsx

import { Product } from "../types/product";

import Link from "next/link";

export default function ProductCard({ product }: { product: Product }) {

return (

<div className="border rounded-xl p-4 shadow-md hover:shadow-lg">

<Link href={`/product/${product.id}`}>

<img src={product.images[0]} alt={product.title} className="w-full h-48 object-cover rounded-md" />

<h2 className="text-lg font-semibold mt-2">{product.title}</h2>

<p className="text-sm text-gray-600">{product.category}</p>

<p className="text-md font-bold text-blue-600">${product.price}</p>

<p className="text-sm text-yellow-500">⭐ {product.rating.rate} ({product.rating.count})</p>

</Link>

</div>

);

}

1. SearchBar.tsx

export default function SearchBar({ onSearch }: { onSearch: (q: string) => void }) {

return (

<input

type="text"

placeholder="Cari produk..."

onChange={(e) => onSearch(e.target.value)}

className="w-full p-2 border rounded-md mb-4"

/>

);

}

1. Context
2. CartContext.tsx

import { createContext, useState, useEffect, ReactNode } from "react";

type CartItem = { product: any; qty: number };

export const CartContext = createContext<{

cart: CartItem[];

add: (item: CartItem) => void;

remove: (id: number) => void;

clear: () => void;

}>({

cart: [],

add: () => {},

remove: () => {},

clear: () => {},

});

export function CartProvider({ children }: { children: ReactNode }) {

const [cart, setCart] = useState<CartItem[]>([]);

useEffect(() => {

const data = localStorage.getItem("cart");

if (data) setCart(JSON.parse(data));

}, []);

useEffect(() => {

localStorage.setItem("cart", JSON.stringify(cart));

}, [cart]);

const add = (item: CartItem) => setCart(prev => [...prev, item]);

const remove = (id: number) => setCart(prev => prev.filter(i => i.product.id !== id));

const clear = () => setCart([]);

return (

<CartContext.Provider value={{ cart, add, remove, clear }}>

{children}

</CartContext.Provider>

);

}

1. ProductContext.tsx

import { createContext, useState, useEffect, ReactNode } from "react";

import api from "../utils/api";

import { Product } from "../types/product";

interface ProductContextType {

products: Product[];

loading: boolean;

}

export const ProductContext = createContext<ProductContextType>({

products: [],

loading: true,

});

export function ProductProvider({ children }: { children: ReactNode }) {

const [products, setProducts] = useState<Product[]>([]);

const [loading, setLoading] = useState(true);

useEffect(() => {

api.get<{ products: Product[] }>("products?limit=100")

.then(res => {

setProducts(res.data.products);

setLoading(false);

})

.catch(err => console.error(err));

}, []);

return (

<ProductContext.Provider value={{ products, loading }}>

{children}

</ProductContext.Provider>

);

}

1. ThemeContext

import { createContext, useState, useEffect, ReactNode } from "react";

export const ThemeContext = createContext<{ theme: string; toggle: () => void }>({ theme: "light", toggle: () => {} });

export function ThemeProvider({ children }: { children: ReactNode }) {

const [theme, setTheme] = useState("light");

useEffect(() => {

const t = localStorage.getItem("theme") || "light";

setTheme(t);

document.documentElement.className = t;

}, []);

const toggle = () => {

const t = theme === "light" ? "dark" : "light";

setTheme(t);

document.documentElement.className = t;

localStorage.setItem("theme", t);

};

return <ThemeContext.Provider value={{ theme, toggle }}>{children}</ThemeContext.Provider>;

}

1. Pages

3.1product

a. [id].tsx

import { useRouter } from "next/router";

import { useContext } from "react";

import { ProductContext } from "../../context/ProductContext";

import { CartContext } from "../../context/CartContext";

export default function ProductDetail() {

const router = useRouter();

const { id } = router.query;

const { products } = useContext(ProductContext);

const { add } = useContext(CartContext);

const product = products.find((p) => p.id === Number(id));

if (!product) return <p className="p-4">Produk tidak ditemukan.</p>;

return (

<div className="p-6 max-w-4xl mx-auto">

<div className="flex flex-col md:flex-row gap-6">

<img src={product.images[0]} alt={product.title} className="w-full md:w-1/2 rounded-lg" />

<div>

<h2 className="text-2xl font-bold">{product.title}</h2>

<p className="text-gray-600">{product.category}</p>

<p className="text-lg font-semibold mt-2 text-blue-700">${product.price}</p>

<p className="text-sm mt-2">{product.description}</p>

<p className="text-sm mt-2 text-yellow-500">⭐ {product.rating.rate} ({product.rating.count})</p>

<button

className="mt-4 px-4 py-2 bg-green-600 text-white rounded hover:bg-green-700"

onClick={() => {

add({ product, qty: 1 });

router.push("/cart");

}}

>

Tambahkan ke Keranjang

</button>

</div>

</div>

</div>

);

}

1. \_app.tsx

import "@/styles/globals.css";

import type { AppProps } from "next/app";

import { ThemeProvider } from "@/context/ThemeContext";

import { ProductProvider } from "@/context/ProductContext";

import { CartProvider } from "@/context/CartContext";

import Navbar from "@/components/Navbar";

export default function App({ Component, pageProps }: AppProps) {

return (

<ThemeProvider>

<ProductProvider>

<CartProvider>

<Navbar />

<Component {...pageProps} />

</CartProvider>

</ProductProvider>

</ThemeProvider>

);

}

1. Cart.tsx

import { useContext } from "react";

import { CartContext } from "../context/CartContext";

import Link from "next/link";

export default function CartPage() {

const { cart, remove, clear } = useContext(CartContext);

const total = cart.reduce((sum, item) => sum + item.product.price \* item.qty, 0);

if (cart.length === 0)

return <p className="p-6 text-center">Keranjang kosong. <Link href="/">Belanja yuk!</Link></p>;

return (

<div className="p-6 max-w-4xl mx-auto">

<h2 className="text-xl font-bold mb-4">🛒 Keranjang Belanja</h2>

<ul className="divide-y">

{cart.map((item) => (

<li key={item.product.id} className="py-4 flex justify-between">

<span>{item.product.title} x{item.qty}</span>

<span>${item.product.price \* item.qty}</span>

<button onClick={() => remove(item.product.id)} className="text-red-600 hover:underline ml-4">

Hapus

</button>

</li>

))}

</ul>

<p className="mt-4 font-semibold">Total: ${total.toFixed(2)}</p>

<div className="flex gap-4 mt-4">

<button className="bg-gray-300 px-3 py-2 rounded" onClick={clear}>Kosongkan</button>

<Link href="/checkout">

<button className="bg-blue-600 text-white px-4 py-2 rounded hover:bg-blue-700">Checkout</button>

</Link>

</div>

</div>

);

}

1. Checkout.tsx

import { useContext, useEffect } from "react";

import { CartContext } from "../context/CartContext";

export default function Checkout() {

const { cart, clear } = useContext(CartContext);

const total = cart.reduce((sum, item) => sum + item.product.price \* item.qty, 0);

useEffect(() => {

if (cart.length > 0) {

localStorage.setItem("orderHistory", JSON.stringify(cart));

clear();

}

}, []);

return (

<div className="p-6 max-w-xl mx-auto text-center">

<h1 className="text-2xl font-bold text-green-700">✅ Pembayaran Berhasil</h1>

<p className="mt-4">Total pembayaran: <strong>${total.toFixed(2)}</strong></p>

<p className="mt-2 text-sm text-gray-500">Order Anda telah disimpan ke Local Storage sebagai riwayat pembelian.</p>

</div>

);

}

1. History.tsx

import { useEffect, useState } from "react";

export default function HistoryPage() {

const [history, setHistory] = useState<any[]>([]);

useEffect(() => {

const h = localStorage.getItem("orderHistory");

if (h) setHistory(JSON.parse(h));

}, []);

if (history.length === 0) return <p className="p-6">Belum ada riwayat pembelian.</p>;

return (

<div className="p-6 max-w-4xl mx-auto">

<h2 className="text-xl font-bold mb-4">🧾 Riwayat Pembelian</h2>

<ul className="divide-y">

{history.map((item, i) => (

<li key={i} className="py-2 flex justify-between">

<span>{item.product.title} x{item.qty}</span>

<span>${(item.product.price \* item.qty).toFixed(2)}</span>

</li>

))}

</ul>

</div>

);

}

1. Index.tsx

import { useContext, useState, useMemo } from "react";

import { ProductContext } from "../context/ProductContext";

import ProductCard from "../components/ProductCard";

import SearchBar from "../components/SearchBar";

import FilterSortBar from "../components/FilterSortBar";

export default function Home() {

const { products, loading } = useContext(ProductContext);

const [searchQuery, setSearchQuery] = useState("");

const [categoryFilter, setCategoryFilter] = useState("");

const [sortOption, setSortOption] = useState("");

const filteredProducts = useMemo(() => {

let filtered = [...products];

if (searchQuery) {

filtered = filtered.filter((p) =>

p.title.toLowerCase().includes(searchQuery.toLowerCase())

);

}

if (categoryFilter) {

filtered = filtered.filter((p) => p.category === categoryFilter);

}

if (sortOption === "price-asc") {

filtered.sort((a, b) => a.price - b.price);

} else if (sortOption === "price-desc") {

filtered.sort((a, b) => b.price - a.price);

} else if (sortOption === "rating") {

filtered.sort((a, b) => b.rating.rate - a.rating.rate);

} else if (sortOption === "stock") {

filtered.sort((a, b) => b.stock - a.stock);

}

return filtered;

}, [products, searchQuery, categoryFilter, sortOption]);

if (loading) return <p className="text-center mt-20">Loading...</p>;

return (

<div className="p-4 max-w-6xl mx-auto">

<h1 className="text-2xl font-bold mb-4">🛍️ Semua Produk</h1>

<SearchBar onSearch={setSearchQuery} />

<FilterSortBar onFilter={setCategoryFilter} onSort={setSortOption} />

<div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-6">

{filteredProducts.map((product) => (

<ProductCard key={product.id} product={product} />

))}

</div>

</div>

);

}

1. Styles
2. Global.css

@tailwind base;

@tailwind components;

@tailwind utilities;

1. Types
2. Product.ts

export interface Rating {

rate: number;

count: number;

}

export interface Product {

id: number;

title: string;

description: string;

price: number;

discountPercentage?: number;

rating: Rating;

stock: number;

brand?: string;

category: string;

images: string[];

}

1. next.config.js

/\*\* @type {import('next').NextConfig} \*/

const nextConfig = {

reactStrictMode: true,

swcMinify: true,

images: {

domains: ["fakestoreapi.in", "fakestoreapi.com"], // Tambahkan domain image yang dipakai

},

};

module.exports = nextConfig;

1. package.json

{

"name": "ecommerce-app",

"version": "1.0.0",

"scripts": {

"dev": "next dev",

"build": "next build",

"start": "next start"

},

"dependencies": {

"next": "^13.0.0",

"react": "^18.0.0",

"react-dom": "^18.0.0",

"axios": "^1.0.0"

},

"devDependencies": {

"tailwindcss": "^3.0.0",

"postcss": "^8.0.0",

"autoprefixer": "^10.0.0"

}

}

1. postcss.config.js

module.exports = {

plugins: {

tailwindcss: {},

autoprefixer: {},

},

};

1. tailwind.config.js

module.exports = {

content: ["./pages/\*\*/\*.{ts,tsx}", "./src/components/\*\*/\*.{ts,tsx}"],

theme: { extend: {} },

plugins: [],

};

1. tsconfig.js

{

"compilerOptions": {

"target": "es6",

"lib": ["dom", "dom.iterable", "esnext"],

"allowJs": true,

"skipLibCheck": true,

"strict": true,

"forceConsistentCasingInFileNames": true,

"noEmit": true,

"esModuleInterop": true,

"module": "esnext",

"moduleResolution": "node",

"resolveJsonModule": true,

"isolatedModules": true,

"jsx": "preserve",

"incremental": true,

"baseUrl": ".",

"paths": {

"@/\*": ["./\*"]

}

},

"include": ["next-env.d.ts", "\*\*/\*.ts", "\*\*/\*.tsx"],

"exclude": ["node\_modules"]

}