**A PROJECT REPORT**

**on**

**“SHOE – STORE WEBSITE”**

***Submitted by***

**Mr. Soyab Shaikh**

**Seat No :-**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF SCIENCE**

**In**

**Information Technology**

***under the guidance of***

**Mrs. Anita Joshi**

**Department of Information Technology**



**VIDYAVARDHINI’S**

**SKC JR. & DEGREE COLLEGE,**

**(Affiliated to University of Mumbai),**

**VASAI(WEST), PALGHAR-401 202,**

**MAHARASHTRA**

**(SEM-V)**  **(2024-2025)**

# ABSTRACT

# The objective of the Shoes e-commerce website is to provide users with the ability to search for quality shoes or sneakers, enable them to make purchase of their desired footwear seamlessly, and enhance their overall shopping experience.

# The system will be developed using a combination of front-end and back-end technologies, including HTML, TailwindCSS, JavaScript, and a backend it uses NextJS . A relational database will be used to store user information, shoes details, and order records.

# The system will prioritize user experience, security, and scalability. It will ensure a seamless and efficient shoes browsing and purchasing process, while protecting user data and preventing fraudulent activities.

**CHAPTER 1: INTRODUCTION**

* 1. **BACKGROUND:**

The Shoe store web Application is a cutting-edge solution designed to provide users with seamless access to a diverse range of shoes, allowing them to

discover, buy, and enjoy their favourite shoes with ease.

This innovative web app offers a user-friendly interface, robust features, and a secure platform for shoes enthusiasts.

The objective of the Shoe Store Web Application is to provide users with the ability to search for quality shoes or sneakers, enable them to make purchase of their desired footwear seamlessly, and enhance their overall shopping

experience.

The web application aims to become a user-friendly and interactive platform for shopping and browsing shoes/sneakers.

* 1. **OBJECTIVE:**
* The objective of the Shoe Store Web Application is to provide users with the ability to search for quality shoes or sneakers, enable them to make purchase of their desired footwear seamlessly, and enhance their overall shopping experience.
* **User-Centric Design:**
* Develop a user-friendly and intuitive interface.
* Ensure a seamless and efficient checkout process.
* View detailed product information and images.
* **Secure and Reliable Platform:**
* Implement robust security measures to protect user data.
* Ensure system reliability and availability.
* Implement fraud prevention mechanisms.
* **Efficient Operations:**
* Add products to the shopping cart.
* **Scalability:**
* Design the system to handle increasing user load and future growth.
* Implement scalable infrastructure to accommodate peak demands.
  1. **PURPOSE:**

The web application aims to become a user-friendly and interactive platform for shopping and browsing shoes/sneakers.

**CHAPTER 2 : SURVEY OF TECHNOLOGY**

For our Online shoe store , we are using the following software and technologies:

* **Backend Technologies:**
* NextJSFramework: Next.js is a React framework for building full-stack web applications. You use React Components to build user interfaces,

and Next.js for additional features and optimizations.

* JavaScript: Next.js is a React framework for building full-stack web applications. You use React Components to build user interfaces, and Next.js for additional features and optimizations.
* PostgreSQL: A powerful, open-source relational database system for storing user data, movie information, and booking details.
* **Frontend Technologies:**
* HTML, CSS, and JavaScript: The core trio for structuring, styling, and adding interactivity to web pages.
* Bootstrap: A popular front-end framework for rapid development of responsive and visually appealing user interfaces.
* **Version Control:**
* Git: A distributed version control system for efficient team collaboration and code management.

By combining these technologies, we aim to create a reliable, efficient, and user-friendly online shoe store.

# CHAPTER 3 : REQUIRMENTS ANALYSIS

**PROBLEM STATEMENT:**

The current traditional method of purchasing and browsing shoes

involves visiting the store or using websites that has certain requirements,

which can be time-consuming and inconvenient. There's a need for a user-friendly, efficient, and secure online platform to browse the shoes and

purchase.

**FUNCTIONAL REQUIREMENTS:**

1. **Shoes Information:**
   * Browse and explore a diverse collection of shoes.
   * View detailed product information and images.
2. **Size Selection:**
   * Allow users to select their preferred size of shoes their comfortable with.
3. **Online Payment:**
   * Integrate secure payment gateways (e.g. Stripe) to process payments.
   * Offer various payment options (credit/debit cards, net banking, digital wallets).
   * Generate a text message upon successful & unsuccessful payment.
4. **Cart:**
   * User can add items to cart and proceed for checkout, the cart items will be saved.
5. **Wishlist:**
   * User can add items to wishlist to save for future payments.
6. **Customer Support:**
   * Provide a customer support system (e.g., email, chat, phone) to address user queries and issues.

**NON-FUNCTIONAL REQUIREMENTS:**

1. **Performance:** 
   * Ensure fast response times and smooth user experience.
   * Optimize database queries and minimize page load times.
2. **Security:** 
   * Implement robust security measures to protect user data and prevent unauthorized access.
   * Use encryption techniques to safeguard sensitive information.
3. **Usability:** 
   * Design a user-friendly and intuitive interface.
   * Provide clear instructions and guidance for users.
   * Optimize the website for mobile devices.
4. **Reliability:** 
   * Ensure high availability and minimal downtime.
   * Implement regular backups and disaster recovery plans.
5. **Scalability:** 
   * Design the system to handle increasing user load and future growth.
   * Use scalable infrastructure to accommodate peak demands.

# REQUIREMENT SPECIFICATIONS :

## SOFTWARE REQUIREMENT :

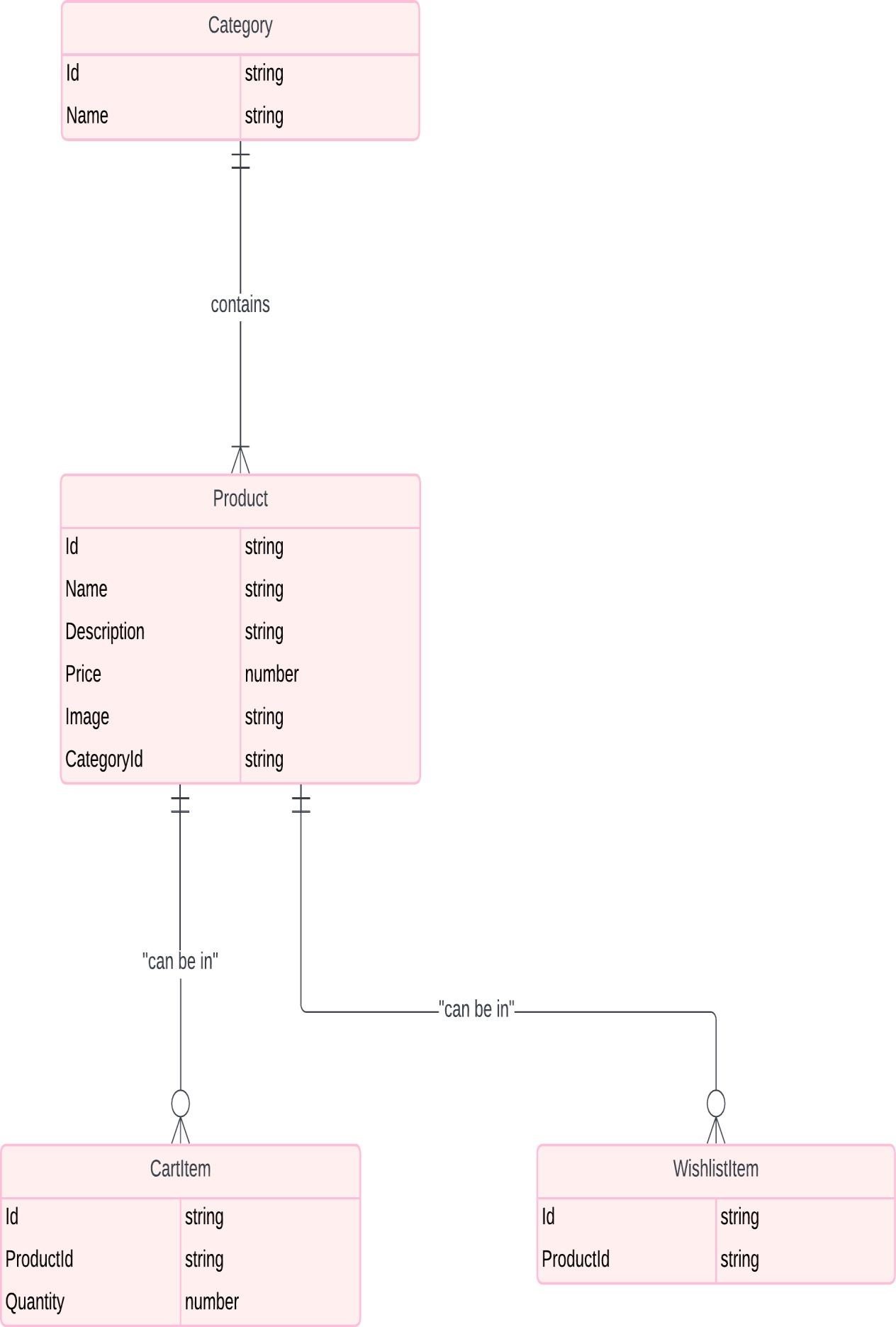
* + IDE - Visual Studio Code 1.87.1
  + Database : PostgreSQL
  + Programming Language : NextJS
  + Html/Telwind-CSS

## HARDWARE REQUIREMENT :

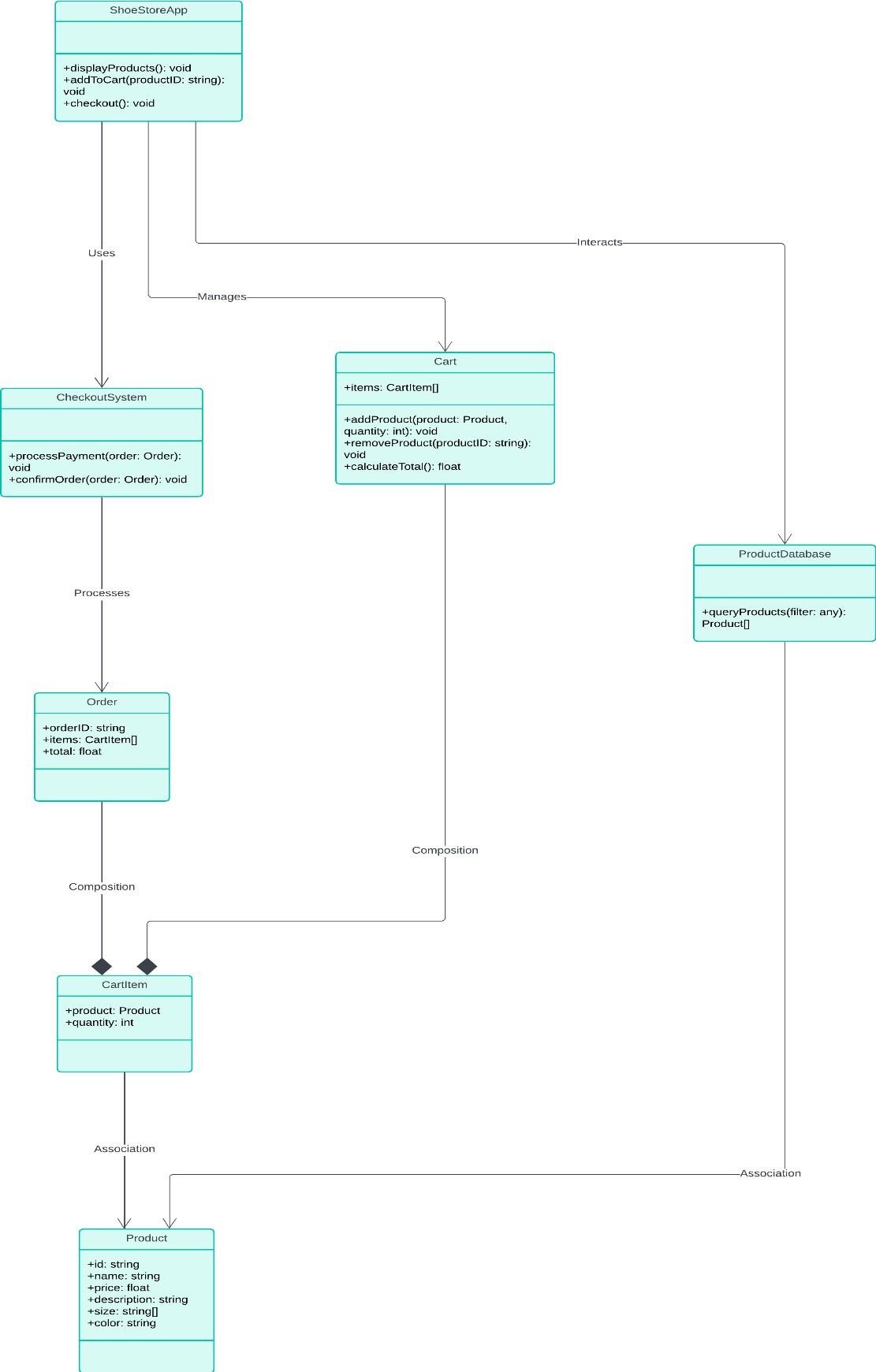
* + Microsoft windows 64 bit or above
  + 512 GB SSD
  + 4 GB RAM
  + OperatingSystem**:** Windows 7 above

# CHAPTER 4 : SYSTEM DESIGN

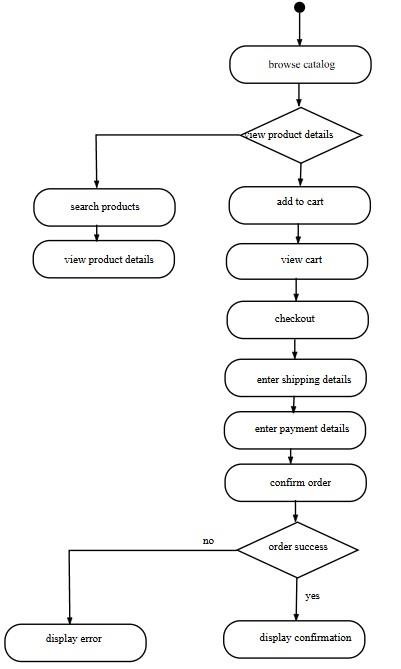
**ER Diagram :**



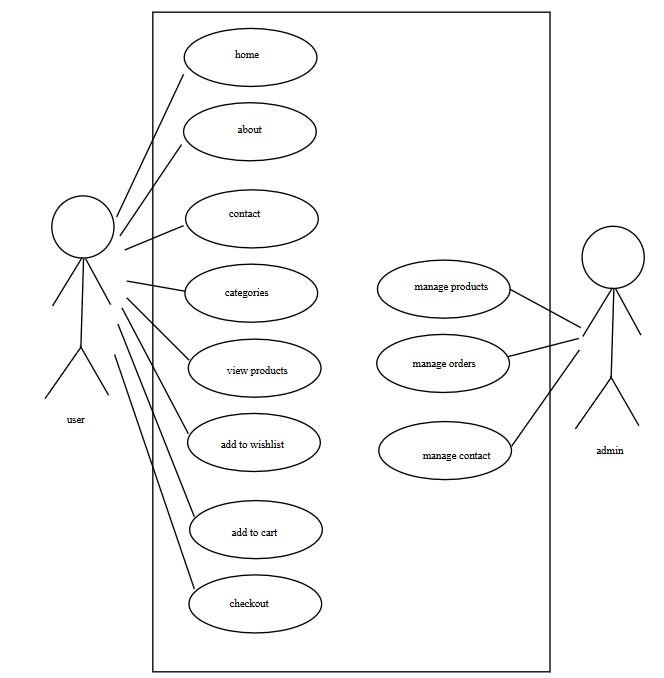
**Class Diagram :**



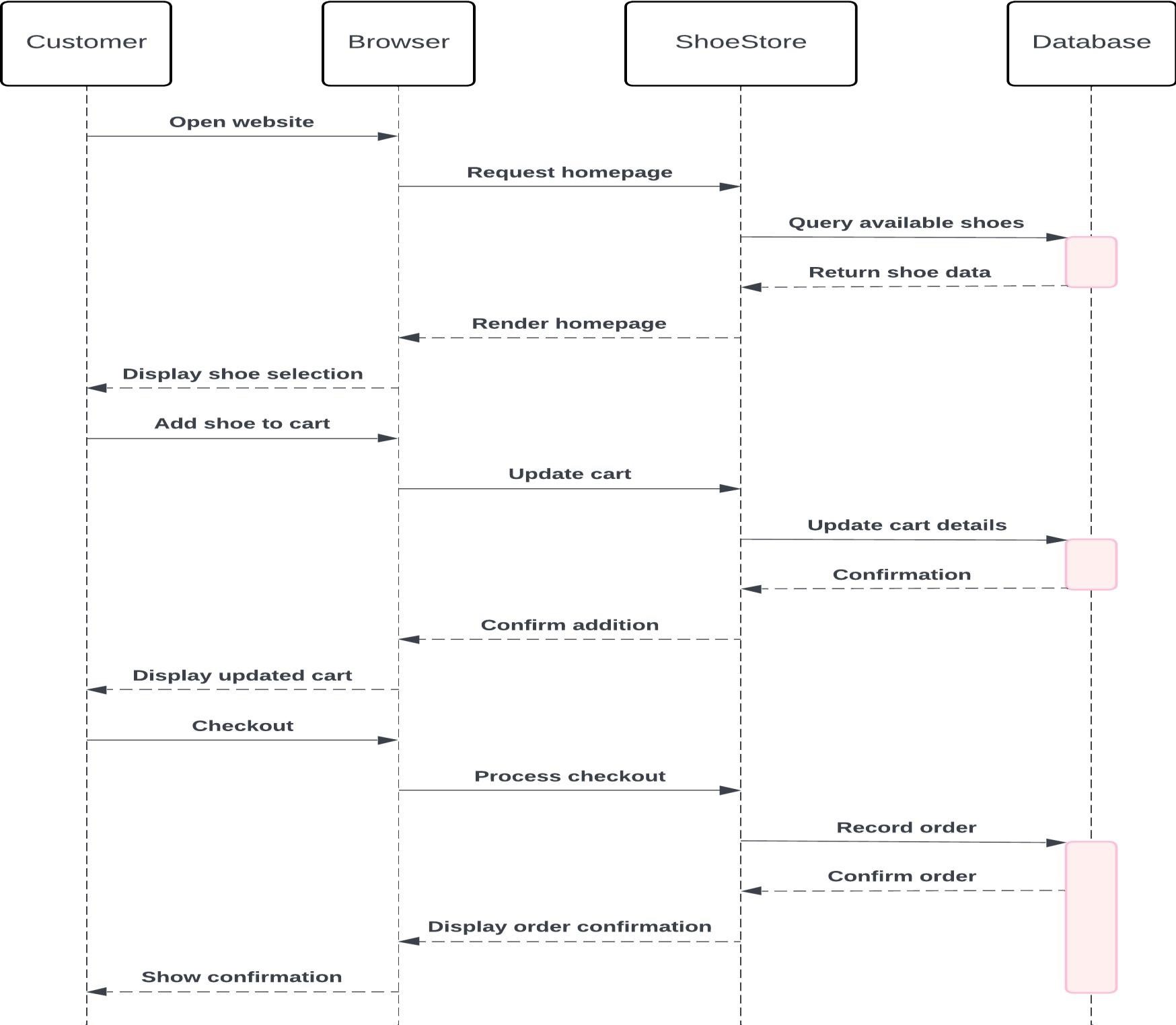
**Activity Diagram :**



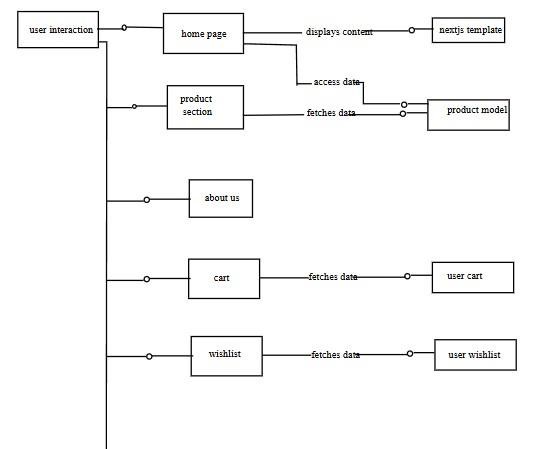
**Usecase Diagram :**



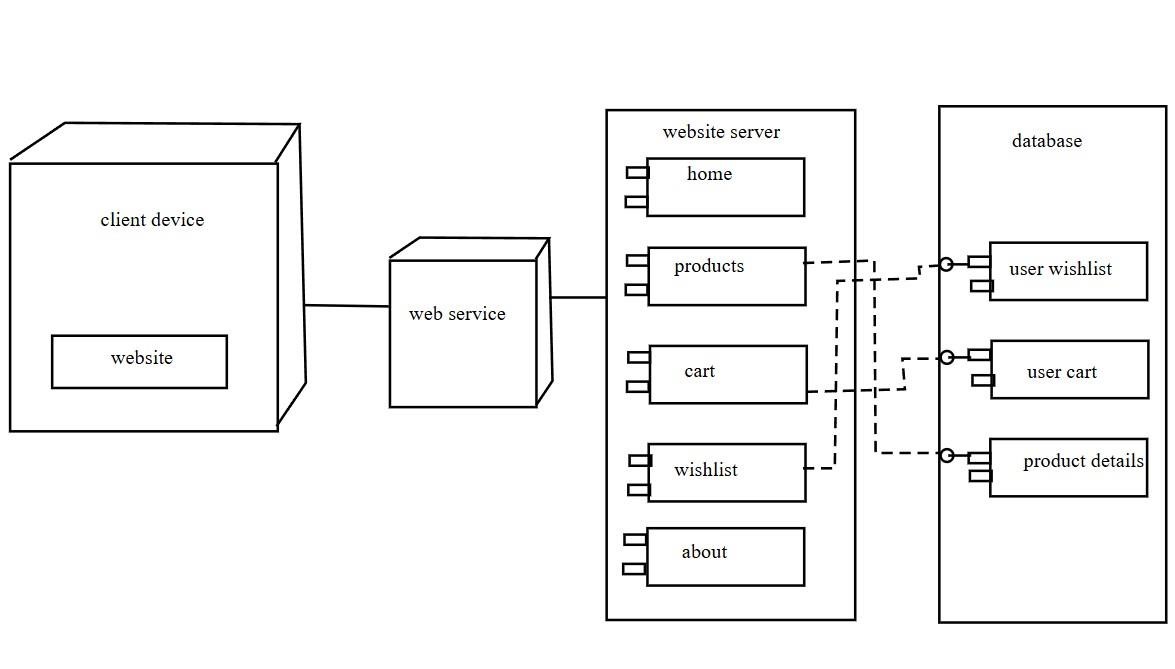
**Sequence Diagram :**



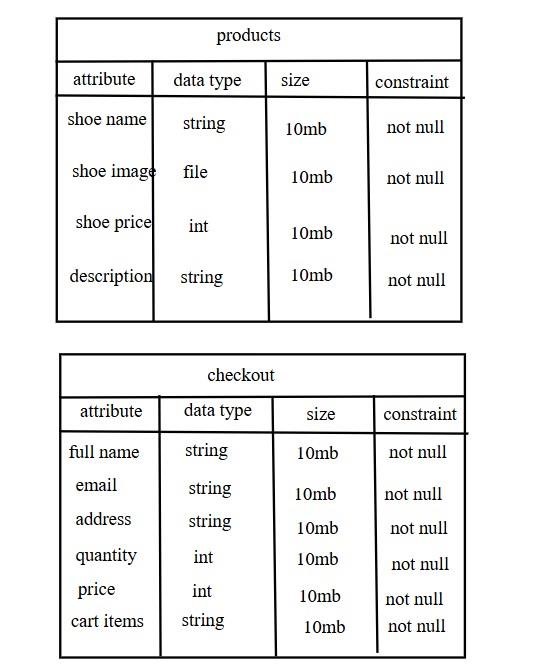
**Component Diagram :**



**Deployment Diagram :**



**Database Model :**



# SYSTEM IMPLEMENTATION

**Menu.jsx :**

import { fetchDataFromAPI } from "@/utils/api";

import Link from "next/link"; import React, { useEffect, useState } from "react"; import { BsChevronDown } from "react-icons/bs";

const data = [

{ id: 1, name: "Home", url: "/" },

{ id: 2, name: "About", url: "/about" },

{ id: 3, name: "Categories", subMenu: true },

{ id: 4, name: "Contact", url: "mailto:nike@gmail.com" },

];

const Menu = ({ showCatMenu, setShowCatMenu, categories }) => { return (

<ul className="hidden md:flex items-center gap-8 font-medium text-black">

{data.map((item) => { return (

<React.Fragment key={item.id}>

{!!item?.subMenu ? (

<li

className="cursor-pointer flex items-center gap-2 relative"

onMouseEnter={() => setShowCatMenu(true)}

onMouseLeave={() => setShowCatMenu(false)}

>

{item.name}

<BsChevronDown size={14} />

{showCatMenu && categories && (

<ul className="bg-white absolute top-6 left-0 min-w-[250px] px-1 py-1 text-black shadow-lg">

{categories.map((category) =>

{ return ( <Link key={category.id} href={`/category/${category.attributes.slug}`} onClick={() => setShowCatMenu(false)}

>

<li className="h-12 flex justify-between items-center px-3 hover:bg-black/[0.03] rounded-md">

{category.attributes.name}

<span className="opacity-50 text-sm">

&#40;

{category.attributes.products.data.length}

&#41;

</span>

</li>

</Link>

);

})}

</ul>

)}

</li>

) : (

<li className="cursor-pointer">

<Link href={item?.url}>{item.name}</Link>

</li>

)}

</React.Fragment>

);

})}

</ul>

);

};

export default Menu; **Index.js**

**:**

import Hero from "@/components/Hero"; import ProductCard from

"@/components/ProductCard"; import Wrapper from

"@/components/Wrapper"; import { fetchDataFromAPI

} from "@/utils/api";

export default function Home({ products }) {

return (

<main>

<Hero />

<Wrapper>

{/\* Heading and Paragraph \*/}

<section className="text-center max-w-[800px] mx-auto my-[50px] md:my-[80px]">

<div className="text-[28px] md:text-[34px] mb-5 font-semibold">

Cushioning for Your Miles

</div>

<div className="text-md md:text-xl">

A lightweight Nike ZoomX midsole is combined with increased stack height to help provide cushioning during extended stretches of running. </div>

</section>

{/\* Product Grid \*/}

<section className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-5 my-14 px-5 md:px0">

{products?.data?.map((product) => (

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

))}

</section>

</Wrapper>

</main>

);

}

export async function getStaticProps() { const products = await

fetchDataFromAPI("/api/products?populate=\*");

return { props: { products },

};

}

**App.js :**  import "@/styles/globals.css";

import Head from "next/head";

import Header from "@/components/Header";

import Footer from "@/components/Footer";

import { Provider } from "react-redux";

import store from "@/store/store"; import { persistStore } from "redux-persist"; import { PersistGate } from "redux-persist/integration/react"; export default function App({ Component, pageProps }) {

let persistor = persistStore(store); return ( <>

<Head>

<title>Online Shoe Store</title>

<meta name="description" content="Buy best shoes online here." />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<link rel="icon" href="/favicon.ico" />

<link rel="preconnect" href="https://fonts.googleapis.com" />

<link rel="preconnect"

href="https://fonts.gstatic.com"

crossOrigin="true"

/> <link

href="https://fonts.googleapis.com/css2?family=Oswald:wght@200;300;400;500;600

;700&family=Urbanist:ital,wght@0,100;0,200;0,300;0,400;0,500;0,600;0,700;0,800; 0,900;1,100;1,200;1,300;1,400;1,500;1,600;1,700;1,800;1,900&display=swap" rel="stylesheet"

/>

</Head>

<Provider store={store}>

<PersistGate persistor={persistor}>

<Header />

<Component {...pageProps} />

<Footer />

</PersistGate>

</Provider>

</>

);

} **Cart.js:**  import Wrapper from "@/components/Wrapper"; import CartItem from

"@/components/CartItem"; import Image from "next/image"; import Link from "next/link"; import React, { useState } from "react"; import { useSelector } from "react-redux"; import { useMemo } from "react";

import { loadStripe } from "@stripe/stripe-js";

import { makePaymentReq } from "@/utils/api";

const stripePromise = loadStripe(

process.env.NEXT\_PUBLIC\_STRIPE\_PUBLISHABLE\_KEY

);

const Cart = () => {

const [loading, setLoading] = useState(false); const {

cartItems } = useSelector((state) => state.cart);

const subTotal = useMemo(() => { return cartItems.reduce((total, value) => total + value.totalPrice, 0); }, [cartItems]);

const handlePayment = async () => {

setLoading(true);

const stripe = await stripePromise; const res = await makePaymentReq("/api/orders", { products: cartItems,

});

await stripe.redirectToCheckout({ sessionId: res.stripeSession.id,

});

};

return (

<main className="w-full md:py-20">

<Wrapper>

{cartItems.length > 0 ? (

<>

{/\* Heading \*/}

<div className="text-center max-w-[800px] mx-auto mt-8 md:mt-0">

<div className="text-[28px] md:text-[32px] mb-5 font-semibold leadingtight">

Shopping cart

</div>

</div>

{/\* Cart Content \*/}

<div className="flex flex-col lg:flex-row gap-12 py-10">

{/\* Cart Items \*/}

<section className="flex-[2]">

<h2 className="text-lg font-bold">Cart Items</h2>

{cartItems.map((item) => (

<CartItem key={item.id} data={item} />

))}

</section>

{/\* Cart Sumamry \*/}

<section className="flex-1">

<h2 className="text-lg font-bold">Summary</h2>

<div className="p-5 my-5 bg-black/[0.05] rounded-xl">

{/\* Subtotal div \*/}

<div className="flex justify-between">

<div className="uppercase text-base md:text-xl font-medium textblack">

Subtotal

</div>

<div className="text-base md:text-xl font-medium text-black ">

MRP: &#8377;{subTotal}

</div> </div>

{/\* Message \*/}

<div className="text-sm md:text-base py-5 border-t mt-5"> The subtotal reflects the total price of your order, including duties and taxes, before any applicable discounts. It does not include delivery costs and international transaction fees.

</div>

</div>

{/\* Checkout Btn \*/}

<button onClick={handlePayment} className="flex items-center justify-center gap-5 w-full py-4 roundedfull

bg-black text-white text-lg font-medium transition-transform active:scale-95 mb3 hover:opacity-75"

>

Checkout

{loading && <img src="/spinner.svg" />}

</button>

</section>

</div>

</>

) : (

<>

{/\* Empty cart Screen\*/}

<div className="flex-[2] flex flex-col items-center pb-[50px] md:-mt-14"> {/\* Empty cart Image \*/}

<Image src="/empty-cart.jpg"

alt="Empty Cart"

width={300}

height={300}

className="w-[300px] md:w-[400px]"

priority={false}

/>

{/\* Empty cart Message \*/}

<span className="text-xl font-bold">Your cart is empty</span>

<span className="text-center mt-4">

Looks like you have not added anything in your cart.

<br />

Go ahead and explore top categories.

</span>

{/\* Link to homepage \*/}

<Link className="py-4 px-8 rounded-full bg-black text-white text-lg

fontmedium transition-transform active:scale-95 mb-3 hover:opacity-75 mt-8" href="/"

>

Continue Shopping

</Link>

</div>

</>

)}

</Wrapper>

</main>

);

};

export default Cart;

**Cartitem.jsx :**  import { updateCart, deleteFromCart } from

"@/store/slices/cartSlice"; import Image from "next/image"; import Link from "next/link"; import React from "react"; import { RiDeleteBin6Line } from "react-icons/ri"; import { useDispatch } from "react-redux";

const CartItem = ({ data }) => { const item = data.attributes;

const dispatch = useDispatch();

// Update cart item fn const updateCartItem = (e, key) => {

let payload = {

key,

val: key === "quantity" ? parseInt(e.target.value) : e.target.value,

id: data.id,

};

dispatch(updateCart(payload));

};

return (

<div className="flex py-5 gap-3 mf:gap-5 border-0 border-b last:border-b0"> {/\* Product Image \*/}

<Link href={`/product/${item.slug}`} className="shrink-0 aspect-square w-[50px] md:w-[120px]"

>

<Image src={item?.thumbnail?.data?.attributes?.url} alt={item.slug} width={120}

height={120}

/>

</Link>

{/\* Product Details \*/}

<div className="w-full flex flex-col">

<div className="flex flex-col md:flex-row justify-between">

{/\* Product Title \*/}

<h2 className="text-lg md:text-2xl font-semibold text-black/[.8]">

{item.name}

</h2>

{/\* Product Subtitle Mobile \*/}

<h3 className="text-sm md:text-md font-medium text-black/[.5] block md:hidden">

{item.subtitle}

</h3>

{/\* Product Price \*/}

<h3 className="text-sm md:text-md font-bold text-black/[.5] mt-2">

MRP: &#8377;{item.price}

</h3>

</div>

{/\* Product Subtitle Desktop \*/}

<h3 className="text-sm md:text-md font-medium text-black/[.5] hidden md:block">

{item.subtitle}

</h3>

{/\* Selectors \*/}

<div className="flex item justify-between mt-4">

<div className="flex items-center gap-2 md:gap-10 text-blaack/[.5] text-sm md:text-md">

{/\* Size Selector \*/}

<div className="flex items-center gap-1">

{/\* label for size selector \*/}

<div className="font-semibold">Size:</div>

{/\* dropdown for size selector \*/}

<select

className="hover:text-black bg-white rounded "

onChange={(e) => updateCartItem(e, "selectedSize")}

>

{item.size.data.map((sizeData, index) => (

<option key={index} value={sizeData.size} disabled={!sizeData.enabled}

selected={data.selectedSize === sizeData.size}

>

{sizeData.size}

</option>

))}

</select> </div>

{/\* Quantity Selector \*/}

<div className="flex items-center gap-1">

{/\* label for quantity selector \*/}

<div className="font-semibold">Quantity: </div>

{/\* dropdown for quantity selector \*/}

<select

className="hover:text-black bg-white rounded "

onChange={(e) => updateCartItem(e, "quantity")}

>

{Array.from({ length: 10 }, (\_, i) => i + 1).map(

(elem, index) => (

<option selected={data.quantity === elem} key={index} value={elem}

>

{elem}

</option>

)

)}

</select>

</div>

</div>

{/\* Delete Btn \*/}

<RiDeleteBin6Line className="cursor-pointer text-black/[.5] hover:text-black text-[16px] md:text-[20px]" onClick={() => dispatch(deleteFromCart({ id: data.id }))}

/>

</div>

</div>

</div>

);

};

export default CartItem;

**Wishlistitem.jsx:**  import { addToWishlist, deleteFromWishlist,

} from "@/store/slices/wishlistSlice"; import Image from "next/image"; import Link from "next/link"; import React from "react"; import { IoMdHeart, IoMdHeartEmpty } from "react-icons/io"; import

{ useDispatch } from "react-redux";

const WishlistItem = ({ data }) => {

const item = data.attributes;

const dispatch = useDispatch();

return (

<div className="flex py-5 gap-3 mf:gap-5 border-b last:border-b-0 ">

{/\* Product Image \*/}

<Link href={"/product/" + item.slug} className="shrink-0 aspect-square w-[50px] md:w-[120px]"

>

<Image

src={item?.thumbnail?.data?.attributes?.url}

alt={item.slug} width={120}

height={120}

/>

</Link>

{/\* Product Details \*/}

<div className="w-full flex flex-col">

<div className="flex flex-col md:flex-row justify-between">

{/\* Product Title \*/}

<h2 className="text-lg md:text-2xl font-semibold text-black/[.8]"> {item.name}

</h2>

{/\* Product Subtitle Mobile \*/}

<h3 className="text-sm md:text-md font-medium text-black/[.5] block md:hidden">

{item.subtitle}

</h3>

{/\* Product Price \*/}

<h3 className="text-sm md:text-md font-bold text-black/[.5] mt-2">

MRP: &#8377;{item.price}

</h3>

</div>

{/\* Product Subtitle Desktop \*/}

<h3 className="text-sm md:text-md font-medium text-black/[.5] hidden md:block">

{item.subtitle}

</h3>

<div className=" flex flex-col md:flex-row md:justify-between "> <div className="text-sm">

{item.description.substring(0, item.description.indexOf(".") + 1)} </div>

{/\* Selectors \*/}

<div className="flex items-center md:justify-center my-3 md:my-0 "> {/\* Delete Btn \*/}

{false ? (

<IoMdHeartEmpty

className="cursor-pointer text-black/[.5] hover:text-black text-[20px] md:text-[24px]"

onClick={() => dispatch(addToWishlist({ id: data.id }))}

/>

) : (

<IoMdHeart

className="cursor-pointer text-red-600 hover:text-red-700 text-[20px] md:text-[24px]"

onClick={() => dispatch(deleteFromWishlist({ id: data.id }))}

/>

)}

</div>

</div>

</div>

</div>

);

};

export default WishlistItem**;** **Database.js :**  const path =

require('path');

module.exports = ({ env }) => { const client = env('DATABASE\_CLIENT', 'sqlite');

const connections = { mysql: { connection: { connectionString: env('DATABASE\_URL'), host: env('DATABASE\_HOST', 'localhost'), port: env.int('DATABASE\_PORT', 3306), database: env('DATABASE\_NAME', 'strapi'), user: env('DATABASE\_USERNAME', 'strapi'), password: env('DATABASE\_PASSWORD', 'strapi'), ssl: env.bool('DATABASE\_SSL', false) && { key: env('DATABASE\_SSL\_KEY', undefined), cert: env('DATABASE\_SSL\_CERT', undefined), ca: env('DATABASE\_SSL\_CA', undefined), capath: env('DATABASE\_SSL\_CAPATH', undefined), cipher:

env('DATABASE\_SSL\_CIPHER', undefined), rejectUnauthorized: env.bool(

'DATABASE\_SSL\_REJECT\_UNAUTHORIZED', true

),

},

},

pool: { min: env.int('DATABASE\_POOL\_MIN', 2), max: env.int('DATABASE\_POOL\_MAX',

10) },

},

mysql2: {

connection: {

host: env('DATABASE\_HOST', 'localhost'), port: env.int('DATABASE\_PORT', 3306), database: env('DATABASE\_NAME', 'strapi'), user: env('DATABASE\_USERNAME', 'strapi'), password: env('DATABASE\_PASSWORD', 'strapi'), ssl: env.bool('DATABASE\_SSL', false) && { key: env('DATABASE\_SSL\_KEY', undefined), cert: env('DATABASE\_SSL\_CERT', undefined), ca: env('DATABASE\_SSL\_CA', undefined), capath: env('DATABASE\_SSL\_CAPATH', undefined), cipher:

env('DATABASE\_SSL\_CIPHER', undefined), rejectUnauthorized: env.bool(

'DATABASE\_SSL\_REJECT\_UNAUTHORIZED', true ),

},

},

pool: { min: env.int('DATABASE\_POOL\_MIN', 2), max: env.int('DATABASE\_POOL\_MAX', 10) },

},

postgres: {

connection: { connectionString: env('DATABASE\_URL'), host: env('DATABASE\_HOST', 'localhost'), port: env.int('DATABASE\_PORT', 5432), database: env('DATABASE\_NAME', 'strapi'), user: env('DATABASE\_USERNAME', 'strapi'), password: env('DATABASE\_PASSWORD', 'strapi'), ssl: env.bool('DATABASE\_SSL', false) && { key: env('DATABASE\_SSL\_KEY', undefined), cert: env('DATABASE\_SSL\_CERT', undefined), ca: env('DATABASE\_SSL\_CA', undefined), capath: env('DATABASE\_SSL\_CAPATH', undefined), cipher:

env('DATABASE\_SSL\_CIPHER', undefined), rejectUnauthorized: env.bool(

'DATABASE\_SSL\_REJECT\_UNAUTHORIZED', true ),

},

schema: env('DATABASE\_SCHEMA', 'public'),

},

pool: { min: env.int('DATABASE\_POOL\_MIN', 2), max: env.int('DATABASE\_POOL\_MAX', 10) },

}, sqlite: { connection: { filename: path.join(

\_\_dirname,

'..',

env('DATABASE\_FILENAME', 'data.db')

),

},

useNullAsDefault: true,

},

};

return {

connection: { client,

...connections[client], acquireConnectionTimeout:

## env.int('DATABASE\_CONNECTION\_TIMEOUT',

60000),

},

};

};

**Wishlist.js:**

import WishlistItem from "@/components/WishlistItem"; import Wrapper from "@/components/Wrapper"; import Image from "next/image"; import Link from "next/link"; import React from "react";

import { useSelector } from "react-redux";

const Wishlist = () => {

const { wishlistItems } = useSelector((state) => state.wishlist);

return (

<main className="w-full md:py-20">

<Wrapper>

{wishlistItems.length > 0 ? (

<>

<div className="text-center max-w-[800px] mx-auto mt-8 md:mt-0">

<div className="text-[28px] md:text-[32px] mb-5 font-semibold leading-tight">

Wishlist

</div>

</div>

<section className="flex-[2] max-w-[1000px] mx-auto ">

{wishlistItems.map((item) => (

<WishlistItem key={item.id} data={item} />

))}

</section>

</>

) : (

<>

{/\* Empty cart Screen\*/}

<div className="flex-[2] flex flex-col items-center pb-

[50px] md:-mt-14">

{/\* Empty cart Image \*/}

<Image src="/empty-cart.jpg"

alt="Empty Cart"

width={300}

height={300}

className="w-[300px] md:w-[400px]"

priority={false}

/>

{/\* Empty cart Message \*/}

<span className="text-xl font-bold">Your wishlist is empty</span>

<span className="text-center mt-4">

Looks like you have not added anything to your wishlist.

<br />

Go ahead and explore top categories.

</span>

{/\* Link to homepage \*/}

<Link

className="py-4 px-8 rounded-full bg-black textwhite

text-lg font-medium transition-transform active:scale-95 mb-3 hover:opacity-75 mt-8"

href="/"

>

Continue Shopping

</Link>

</div>

</>

)}

</Wrapper>

</main>

);

};

export default Wishlist;

**cartSlice.js:**  const { createSlice } = require("@reduxjs/toolkit"); export const cartSlice = createSlice({ name: "cart", initialState: { cartItems: [],

}, reducers: { // Add to cart reducer addToCart: (state, action) => { const item = state.cartItems.find((p) => p.id === action.payload.id); if (item) { item.quantity++;

item.totalPrice = item.attributes.price \* item.quantity;

} else { state.cartItems.push({ ...action.payload, quantity: 1,

totalPrice: action.payload.attributes.price,

});

}

},

//Update cart reducer updateCart: (state, action) => { state.cartItems =

state.cartItems.map((item) => { if (item.id === action.payload.id) { if (action.payload.key === "quantity") { item.totalPrice = item.attributes.price \* action.payload.val;

} return {

...item,

[action.payload.key]: action.payload.id,

}; } else { return item;

}

});

},

// Delete cart item reducer deleteFromCart: (state, action) => { state.cartItems = state.cartItems.filter(

(item) => item.id !== action.payload.id

);

},

//Clear cart reducer clearCart: (state, action) => {

state.cartItems = [];

},

},

});

export const { addToCart, updateCart, deleteFromCart, clearCart

} =

cartSlice.actions;

export default cartSlice.reducer; **wishlistSlice.js:**

const { createSlice } =

require("@reduxjs/toolkit");

export const wishlistSlice = createSlice({

name: "wishlist", initialState: { wishlistItems: [],

},

reducers: {

// Add to wishlist reducer addToWishlist: (state, action) => { const item = state.wishlistItems.find((p) => p.id === action.payload.id);

if (!item) {

state.wishlistItems.push({

...action.payload,

});

}

}, // Delete wishlist item reducer deleteFromWishlist: (state, action) => { state.wishlistItems = state.wishlistItems.filter(

(item) => item.id !== action.payload.id

);

},

},

});

export const { addToWishlist, deleteFromWishlist } = wishlistSlice.actions;

export default wishlistSlice.reducer;

**Header.jsx :**  import React, { useState, useEffect } from "react"; import Wrapper from "./Wrapper"; import Link from "next/link";

import Menu from "./Menu";

import { IoMdHeartEmpty } from "react-icons/io"; import { BsCart } from "react-icons/bs"; import { BiMenuAltRight } from "react-icons/bi"; import {

VscChromeClose } from "react-icons/vsc"; import MenuMobile from "./MenuMobile"; import { fetchDataFromAPI } from "@/utils/api"; import { useSelector } from "react-redux";

import Image from "next/image";

const Header = () => { const [mobileMenu,

setMobileMenu] = useState(false); const [showCatMenu, setShowCatMenu] = useState(false); const [show, setShow] = useState("translate-y-0");

const [lastScrollY, setLastScrollY] = useState(0);

const [categories, setCategories] = useState(null);

const { cartItems } = useSelector((state) => state.cart); const { wishlistItems } = useSelector((state) => state.wishlist);

const handleCategoriesFetch = async () => { const res = await

fetchDataFromAPI("/api/categories?populate=\*"

);

setCategories(res.data);

};

useEffect(() => {

handleCategoriesFetch();

}, []);

const controlNavbar = () => {

if (window.scrollY > 200) {

if (window.scrollY > lastScrollY && !mobileMenu) { setShow("-translate-y-[80px]");

} else { setShow("shadow-sm");

} } else {

setShow("translate-y-0");

}

setLastScrollY(window.scrollY);

};

useEffect(() => {

window.addEventListener("scroll", controlNavbar);

return () => { window.removeEventListener("scroll", controlNavbar);

};

// eslint-disable-next-line

}, [lastScrollY]);

return (

<header className={`w-full h-[50px] md:h-[80px] bg-white flex items-center

justify-between z-20 sticky top-0 transition-transform

duration-300 ${show}`}

>

<Wrapper className="h-[60px] flex justify-between items-center">

<Link href="/">

<Image src="/logo.svg" alt="nike-logo"

width={60}

height={60} className="w-[40px] md:w-[60px]"

/>

</Link>

<Menu showCatMenu={showCatMenu} setShowCatMenu={setShowCatMenu} categories={categories}

/> {mobileMenu && ( <MenuMobile showCatMenu={showCatMenu} setShowCatMenu={setShowCatMenu} setMobileMenu={setMobileMenu} categories={categories}

/>

)}

{/\* Icons Section \*/}

<section className="flex items-center gap-2 text-black">

{/\* Wishlist \*/}

<Link href="/wishlist">

<div className="w-8 md:w-12 h-8 md:h-12 rounded- full flex justify-center items-center hover:bgblack/[0.05]

cursor-pointer relative ">

<IoMdHeartEmpty className="text-[19px] md:text-

[24px]" />

{wishlistItems.length > 0 && (

<div className="h-[14px] md:h-[18px] min-w[14px] md:min-w-[18px] rounded-full bg-red600 absolute top-1 left-5 md:left-7 text-white text[10px] md:text-[12px] flex justify-center itemscenter px-[2px] md:px-[5px]">

{wishlistItems.length}

</div>

)}

</div>

</Link>

{/\* Cart \*/}

<Link href="/cart">

<div className="w-8 md:w-12 h-8 md:h-12 rounded- full flex justify-center items-center hover:bgblack/[0.05]

cursor-pointer relative ">

<BsCart className="text-[15px] md:text-[20px]" />

{cartItems.length > 0 && (

<div className="h-[14px] md:h-[18px] min-w[14px] md:min-w-[18px] rounded-full bg-red600 absolute top-1 left-5 md:left-7 text-white text[10px] md:text-[12px] flex justify-center itemscenter px-[2px] md:px-[5px]">

{cartItems.length}

</div>

)}

</div>

</Link>

{/\* Mobile Menu \*/}

<div className="w-8 md:w-12 h-8 md:h-12 roundedfull flex md:hidden justify-center items-center hover:bg-black/[0.05] cursor-pointer relative mr2">

{mobileMenu ? ( <VscChromeClose className="text-16px "

onClick={() => setMobileMenu(false)}

/>

) : (

<BiMenuAltRight className="text-20px " onClick={()

=> setMobileMenu(true)}

/>

)}

</div>

</section>

</Wrapper>

</header>

);

};

export default Header;

**Footer.jsx :** import React from "react"; import { FaFacebookF, FaTwitter, FaYoutube, FaInstagram } from "react-icons/fa";

import Wrapper from "./Wrapper";

import Link from "next/link";

const Footer = () => { return (

<footer className="bg-black text-white pt-14 pb-3">

<Wrapper className="flex justify-between flex-col md:flex-row gap-[50px] md:gap-0">

{/\* Left \*/}

<div className="flex gap-[50px] md:gap-[75px] lg:gap-[100px] flex-col md:flex-row ">

{/\* Menu \*/}

<div className="flex flex-col gap-3 shrink-0">

<div className="font-oswald font-medium uppercase text-sm cursorpointer">

Find a store

</div>

<div className="font-oswald font-medium uppercase text-sm cursorpointer">

Become a partner

</div>

<div className="font-oswald font-medium uppercase text-sm cursorpointer">

sign up for email </div>

<div className="font-oswald font-medium uppercase text-sm cursorpointer">

send us feedback </div>

<div className="font-oswald font-medium uppercase text-sm cursorpointer">

student discount

</div>

</div>

{/\* Normal Menu \*/}

<div className="flex gap-[50px] md:gap-[75px] lg:gap-[100px] shrink-0"> {/\* Menu \*/}

<div className="flex flex-col gap-3">

<div className="font-oswald font-medium uppercase textsm"> get help </div>

<div className="text-sm hover:text-white cursor-pointer text-white/[0.5] "> Order Status </div>

<div className="text-sm hover:text-white cursor-pointer text-white/[0.5] "> Delivery </div>

<div className="text-sm hover:text-white cursor-pointer text-white/[0.5] ">

Return

</div>

<div className="text-sm hover:text-white cursor-pointer text-white/[0.5] "> Payment Options </div>

<div className="text-sm hover:text-white cursor-pointer text-white/[0.5] "> Contact Us

</div> </div>

{/\* Menu \*/}

<div className="flex flex-col gap-3">

<div className="font-oswald font-medium uppercase text-sm">

About nike

</div>

<div className="text-sm hover:text-white cursor-pointer text-white/[0.5] "> News </div>

<div className="text-sm hover:text-white cursor-pointer text-white/[0.5] ">

Careers

</div>

<div className="text-sm hover:text-white cursor-pointer text-white/[0.5] "> Investors

</div>

<div className="text-sm hover:text-white cursor-pointer text-white/[0.5] "> Sustainability

</div>

</div>

{/\* Normal Menu End\*/}

</div>

{/\* Left End \*/}

</div>

{/\* Right \*/}

<div className="flex gap-4 justify-center md:justify-start">

<Link rel="noopener noreferrer"

target="\_blank" href="https://facebook.com/"

className="w-10 h-10 rounded-full bg-white/[.25] flex items-center

justifycenter text-black hover:bg-white/[.5] cursor-pointer"

>

<FaFacebookF size={20} />

</Link>

<Link

rel="noopener noreferrer"

target="\_blank"

href="https://twitter.com/" className="w-10 h-10 rounded-full bg-white/[.25] flex items-center

justifycenter text-black hover:bg-white/[.5] cursor-pointer"

>

<FaTwitter size={20} />

</Link>

<Link

rel="noopener noreferrer"

target="\_blank"

href="https://youtube.com/" className="w-10 h-10 rounded-full bg-white/[.25] flex items-center

justifycenter text-black hover:bg-white/[.5] cursor-pointer"

>

<FaYoutube size={20} />

</Link>

<Link

rel="noopener noreferrer"

target="\_blank"

href="https://www.instagram.com/" className="w-10 h-10 rounded-full bg-white/[.25] flex items-center

justifycenter text-black hover:bg-white/[.5] cursor-pointer"

>

<FaInstagram size={20} />

</Link>

</div>

{/\* Right End \*/}

</Wrapper>

<Wrapper className="flex justify-between mt-10 flex-col md:flex-row gap[10px] md:gap-0">

{/\* Left \*/}

<div className="text-[12px] text-white/[.5] hover:text-white cursor-pointer text-center md:text-left">

&copy; 2023 Nike, Inc. All Rights Reserved

</div>

{/\* Left End\*/}

{/\* Right \*/}

<div className="flex gap-2 md:gap-5 text-center md:text-left flex-wrap justifycenter">

<div className="text-[12px] text-white/[.5] hover:text-white cursor-pointer">

Guides

</div>

<div className="text-[12px] text-white/[.5] hover:text-white cursor-pointer"> Terms of Sale </div>

<div className="text-[12px] text-white/[.5] hover:text-white cursor-pointer"> Terms of Use </div>

<div className="text-[12px] text-white/[.5] hover:text-white cursor-pointer"> Privacy Policy

</div>

</div>

{/\* Right End\*/}

</Wrapper>

</footer>

);

};

export default Footer; **Category(slug.jsx) :**

import Loader from "@/components/Loader"; import ProductCard from "@/components/ProductCard"; import Wrapper from "@/components/Wrapper"; import { fetchDataFromAPI } from "@/utils/api"; import React,

{ useEffect, useState } from "react"; import useSWR from "swr";

const maxResults = 3;

const Category = ({ categoryData, categoryProductData, slug }) => {

const [pageIndex, setPageIndex] = useState(1);

// Every tine category changes go to page 1 useEffect(() => {

setPageIndex(1); }, [categoryData]);

const { data, error, isLoading } = useSWR(

`/api/products?populate=\*&[filters][categories][slug][$eq]=${slug}&pagi nation[page]=${pageIndex}&pagination[pageSize]=${maxResults}`,

fetchDataFromAPI,

{

fallbackData: categoryProductData,

}

);

return (

<main className="w-full md:py-20 relative">

<Wrapper>

{/\* Heading Section \*/}

<section className="text-center max-w-[800px] mx-auto mt-8 md:mt-0"> <h1 className="text-[28px] md:text-[34px] mb-5 font-semibold leading-tight capitalize">

{categoryData?.data?.[0]?.attributes?.name || slug}

</h1>

</section>

{/\* Product Grid \*/}

<section className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-5 my-14 px-5 md:px-0">

{data?.data.map((product) => (

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

))}

</section>

{/\* Pagination butttons \*/}

{data?.meta?.pagination?.total > maxResults && (

<div className="flex gap-3 items-center justify-center my-16 md:my-0">

<button className={`rounded py-2 px-4 bg-black text-white disabled:bg-gray-200 disabled:textgray-500`}

disabled={pageIndex === 1} onClick={() => setPageIndex(pageIndex - 1)}

>

Previous

</button>

<span className="font-bold">

{`${pageIndex} of ${data && data.meta.pagination.pageCount}`}

</span>

<button className={`rounded py-2 px-4 bg-black text-white disabled:bg-gray-200 disabled:textgray-500`}

disabled={pageIndex === data?.meta?.pagination?.pageCount}

onClick={() => setPageIndex(pageIndex + 1)}

>

Next

</button>

</div>

)}

{/\* Loading \*/}

{isLoading && <Loader />}

</Wrapper>

</main>

);

};

export default Category;

export async function getStaticPaths() { const categories = await fetchDataFromAPI("/api/categories?populate=\*");

const paths = categories?.data?.map((cat) => ({

params: {

slug: cat.attributes.slug,

},

}));

return {

paths, fallback: false,

};

}

export async function getStaticProps({ params: { slug } }) {

const categoryData = await fetchDataFromAPI(

`/api/categories?filters[slug][$eq]=${slug}`

);

const categoryProductData = await fetchDataFromAPI( `/api/products?populate=\*&[filters][categories][slug][$eq]=${slug}&pagi nation[page]=1&pagination[pageSize]=${maxResults}`

);

return { props: { categoryData, categoryProductData, slug } }; }

**USER INTERFACE**

