**SHOPSY ECOMMERCE STORE**

A Project Report

Submitted in partial fulfillment of the

Requirements for the award of the Degree of

**BACHELOR OF SCIENCE [COMPUTER SCIENCE]**

Designed And Developed

By

**Mr. Mohammad Riyaz Shaikh**

Under the esteemed guidance of

**Mr. Hasan Phudinawala**

**Assistant Professor**



**Since 1989**

**DEPARTMENT OF COMPUTER SCIENCE**

**ROYAL COLLEGE OF ARTS, SCIENCE & COMMERCE**

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

**MUMBAI 401107**

**MAHARASHTRA**

**2022-2023**

**ACKNOWLEDGEMENT**

**ACKNOWLEDGEMENT**

Achievement is finding out what you would be doing rather than what you have to do. It is not until you undertake such a project that you realize how much effort and hard work it really is, what are your capabilities and how well you can present yourself or other things. It tells us how much we rely on the efforts and goodwill of others. It gives me immense pleasure to present this report towards the fulfilment of my project.

It has been rightly said that we are built on the shoulder of others. For everything I have achieved, the credit goes to all those who had helped me to complete this project successfully.

I take this opportunity to express my profound gratitude to management of **Royal College of Arts, Commerce & Science** for giving me this opportunity to accomplish this project work.

A special vote of thanks to **Prof. Hasan Phudinawala** who is our professor & project guide for his most sincere, useful and encouraging contribution throughout the project span.

Finally, I would like to thank entire Computer Science department who directly or indirectly helped me in completion of this project & to my family without whose support, motivation & encouragement this would not have been possible.

**MOHAMMAD RIYAZ SHAIKH**

**DECLARATION**

**DECLARATION**

I “Mr. MOHAMMAD RIYAZ SHAIKH” hereby declare that I myself have completed the project under the guidance of **Prof. Hasan Phudinawala**. I on my own have designed the system and have done all the programming required.

It may require some modifications in the future as per the user’s requirements. From practical implementation point of view flexibility in the changes have incorporated in the package.

I am sure that I can do any kind of modification suggested while practical implementation by modifying file design or the program code if necessary.

**MOHAMMAD RIYAZ SHAIKH**

**CERTIFICATE**



**Since 1989**

**DEPARTMENT OF COMPUTER SCIENCE**

Class: **TYBSc**  Roll. No. 41

Seat No.\_\_\_\_\_\_\_

**Certificate**

Certified that **Mr. MOHAMMAD RIYAZ SHAIKH** of T.Y.BSc Semester-V has successfully completed the project as prescribed by the University of Mumbai on **SHOPSY ECOMMERCE STORE** as partial fulfillment of requirement for completing Bachelor’s Degree in Computer Science during the academic year 2022-2023.

Signature of Project Guide

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature of Examiner

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

H.O.D

Dept. of Computer Science

**ABOUT THE PROJECT**

**OBJECTIVES**

Reduce Management Cost

Businesses aim at reducing the costs incurred for the betterment of their revenue. Automating the ecommerce business can help in reducing the management cost significantly. Moreover, the right use of digital marketing can help in reducing the cost spent on driving customers to such an extent that businesses can bring customers for free of cost.

Providing a unique customer experience

Uncountable ecommerce businesses are functioning out there in the market. When a customer searches for a certain product (for instance, shampoo), they will probably click on the first three links that are shown on the Google Search Engine Results Page. All the rest links are either avoided, never seen, or are visited by a few. This itself shows the competition in the ecommerce market. One of the best ways to stand out from the crowd is by providing a unique customer experience. This includes giving a personalized experience to each customer or visitor of your online store, website, or mobile app. Some other pointers to consider are round the clock customer service, immediate responses to the queries rose, engaging with the customers, and so on.

Increasing the number of loyal customers

Customers are the core of all business strategies. Therefore, ensuring the great customer experience is of prime importance for the growth of the business. You need to meet your customers where they spend their time. More than 60% of consumers look for purchasing goods and services online. If you meet your customers where they are already active, the chances of them, interacting with your business increases two folds. You can increase the number of loyal customers by giving the best experience to your already existing customers as well as bring in newer customers.

Boosting the efficiency of services

With the continually evolving technology, you need to enhance the efficiency of your services. By choosing an online ecommerce platform to create an online store, you can efficiently reduce the cost of managing and selling online. You have various opportunities to boost the efficiency of your service that eventually enhances the revenue earned. By reducing the delivery time, you can witness happy customers getting back to your business two times faster. Another way is to provide your customers with automated services such as status update, invoice creating, chat support, etc. When you update your efficiency of delivering products or services to your customers, you are creating a strong online presence that helps you sell more.

**INDEX**

**INDEX**

|  |  |  |
| --- | --- | --- |
| SR.  NO. | TOPIC | PAGE  NO. |
| I. | **Preliminary Investigation** | 12 |
| 1.1 | Present System and its advantage | 13 |
| 1.2 | System Requirements | 14 |
| 1.3 | Phase Title | 15 |
| 1.4 | Gantt Chart | 17 |
| II. | **System Analysis** | 18 |
| 2.1 | Event Table | 19 |
| 2.2 | Use Case Diagram | 20 |
| 2.3 | ERD | 21 |
| 2.4 | Class Diagram | 22 |
| III. | **System Design** | 23 |
| 3.1 | Sequence Diagram | 24 |
| 3.2 | Activity Diagram | 25 |
| IV. | **System Coding** | 34 |
| 4.1 | System Coding | 35 |
| V. | **Future Enhancement** | 69 |
| VI. | **Reference Requirement** | 70 |

**PRELIMINARY INVESTIGATION**

**PRELIMINARY INVESTIGATION**

Present System and its advantages:

* The user has to manually go and shop that can be tiring. But if the user uses this site he/she can shop from anywhere at anytime.
* The proposed e-commerce website allows user to shop any product of his/her choice by sitting at home.
* Consumers can get full information about the product while its review being passed by the existing user. If one wants to buy a product he/she is no longer limited to asking the friends and families because there are many products reviews on the web which gives opinions of the existing users of the product.
* In online shopping, consumers are facilitated with online tracking as they can easily track the order and delivery status.
* The buyer has multiple options and multiple brands to choose and from at a single place. There is a large volume of stock is available online.
* During festivals, weekends, or holidays it is common to see the crowd at shopping places; it will be a huge headache and hectic to purchase products in this type of environment. But in online shopping, we do not have to face crowds, and also, we do not have any need to do unusual battles for parking.
* In a pandemic situation like appeared before the user will prefer online shopping rather than physical shopping.
* No need to work, save time and effort.

**System Requirements :**

**Hardware Requirements :**

* Operating System : Windows/Linux
* Ram : Minimum 4GB RAM
* Internal Storage : Minimum 4GB
* Storage Type : Solid State Drive (SSD) / Hard Disk Drive (HDD)
* Processor : Intel Core i3 5th generation (min) or above AMD Ryzen 3000 (min)
* Processor Architecture : x64 or ARM 64

**Software Requirements :**

* Internet Connectivity : Yes
* Database : MongoDB
* Language : JavaScript ES(6), JSX, HTML5, CSS,

Node JS, Express JS, React-BootStrap5

* Framework : React JS

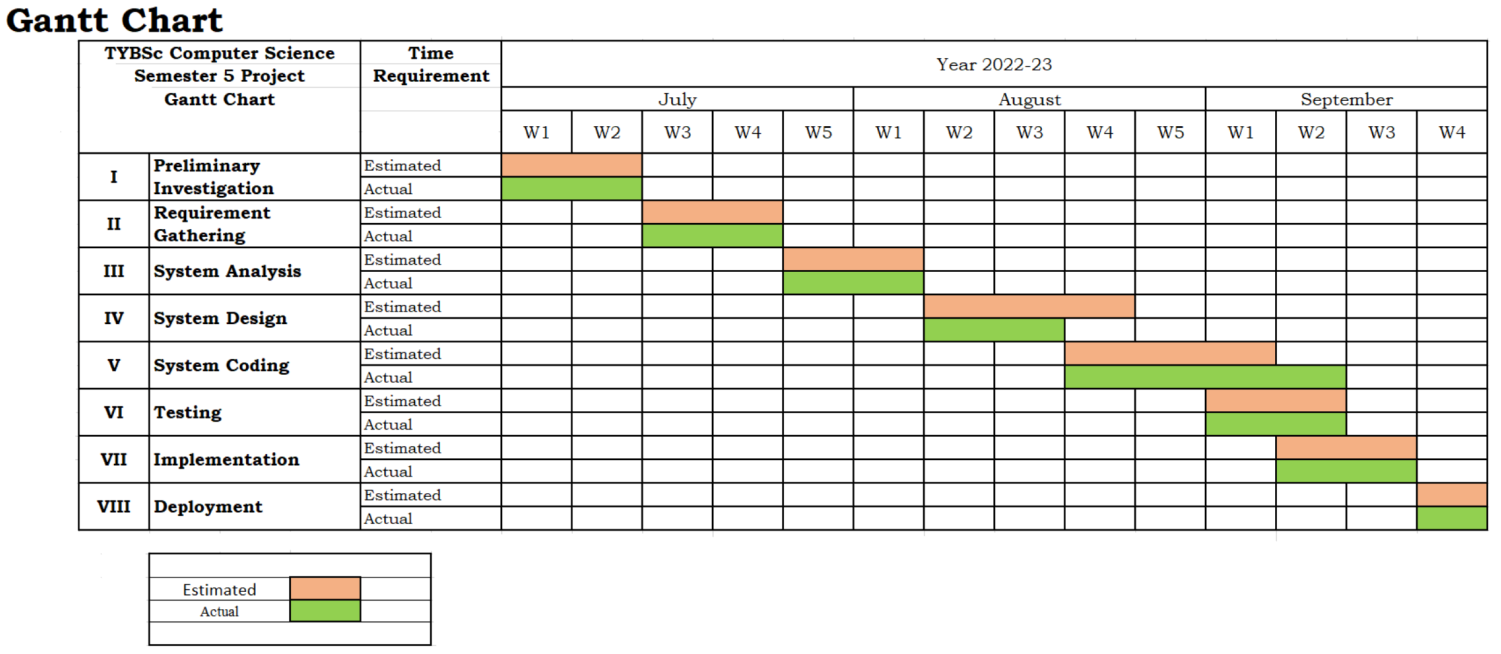
**PHASE TITLE:**

**TYBSC Computer Science Semester 5**

**2022 -2023**

|  |  |  |  |
| --- | --- | --- | --- |
| **Phase Title** | **Expected Date of Completion** | **Actual Time of Completion with Guide’s Signature** | **Remarks** |
| **I. Preliminary Investigation** |  |  |  |
| (i) Organizational Overview |  |  |  |
| (ii) Present System and its advantages | 1 02/07/2022 |  |  |
| (iii) System Requirements |  |  |  |
| (iv) Feasibility Study |  |  |  |
| (v) Fact Finding Methods |  |  |  |
| (vi) Phase Title | 25/07/2022 |  |  |
| (vii) Gantt Chart |  |  |  |
| **II. System Analysis** |  |  |  |
| (i) Event Table |  |  |  |
| (ii) Use Case Diagram | 10/08/2022 |  |  |
| (iii) ERD |  |  |  |
| (iv) Class Diagram |  |  |  |
| **III. System Design** |  |  |  |
| 1. Sequence Diagram 2. Activity Diagram |  |  |  |
| **IV. System Coding** | 24/09/2022 |  |  |
| (i) System Coding |  |  |  |
| (ii) Test Cases |  |  |  |
| (iii) Form Layouts |  |  |  |
| (iv) Report Layouts |  |  |  |
| **V. Future Enhancements** |  |  |  |
| **VI. Reference and Bibliography** | 29/09/2022 |  |  |

Student Name : MOHAMMAD RIYAZ SHAIKH Roll No : 41

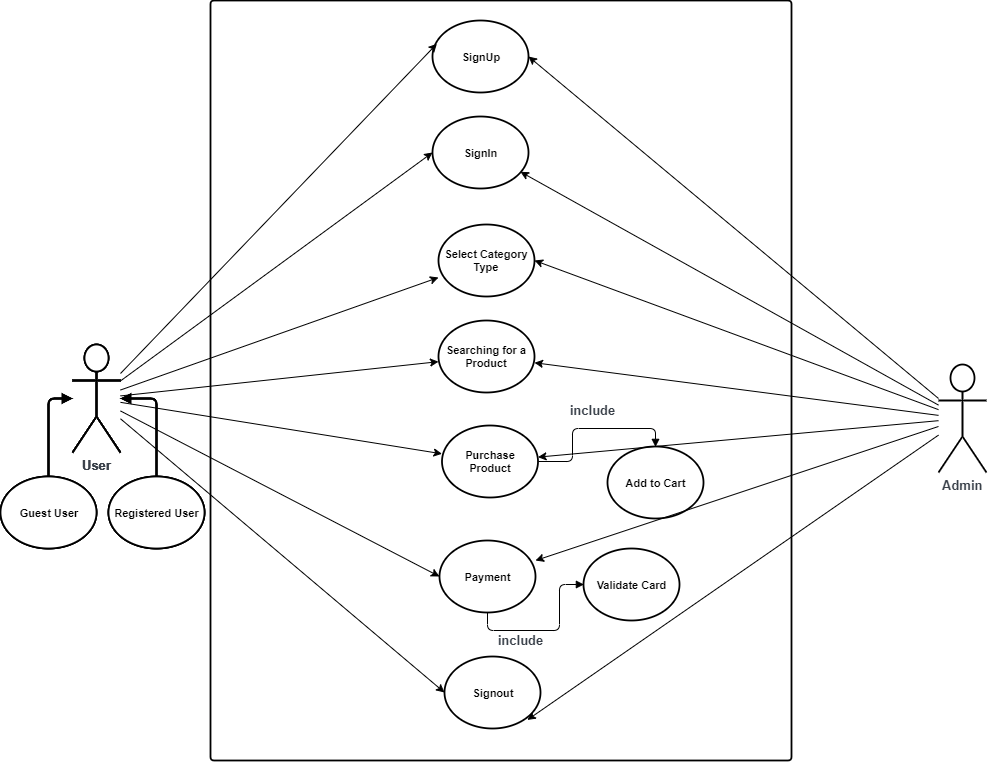


**SYSTEM ANALYSIS**

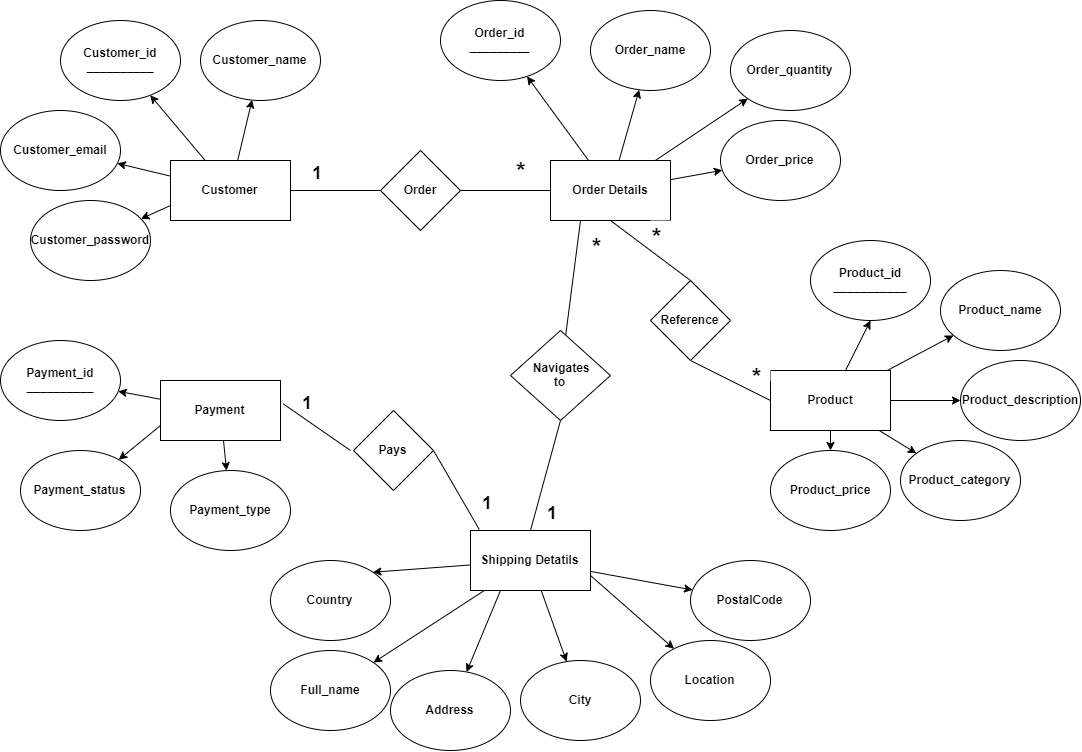
**Event Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Event** | **Trigger** | **Source** | **Activity** | **Response** | **Destination** |
| User visits the website | Site is viewed | User | Signup Page is Viewed | Signup Information | System |
| User Registers | Accept Details | User | Recording Details | Customer Added | System |
| User Logs in | Signin | User | Signin with Credentials | Login Successful | System |
| User Add Product to Cart | Add Product to Cart | User | Display Cart | Product info displayed | System |
| User Removes Product from Cart | Product is Selected | User | Product is Removed | Total Products Shown | System |
| User Profile | Update existing Customer | User | Changing Customer info | Customer info changed | System |
| Search Products | Searching the Product | User | Search Related Product | Matching Product Appears | System |
| Place Order | Generate Order Request | User | Bank Validation | Order Generated | User |

**USE CASE DIAGRAM**



**ENTITY RELATIONSHIP DIAGRAM**

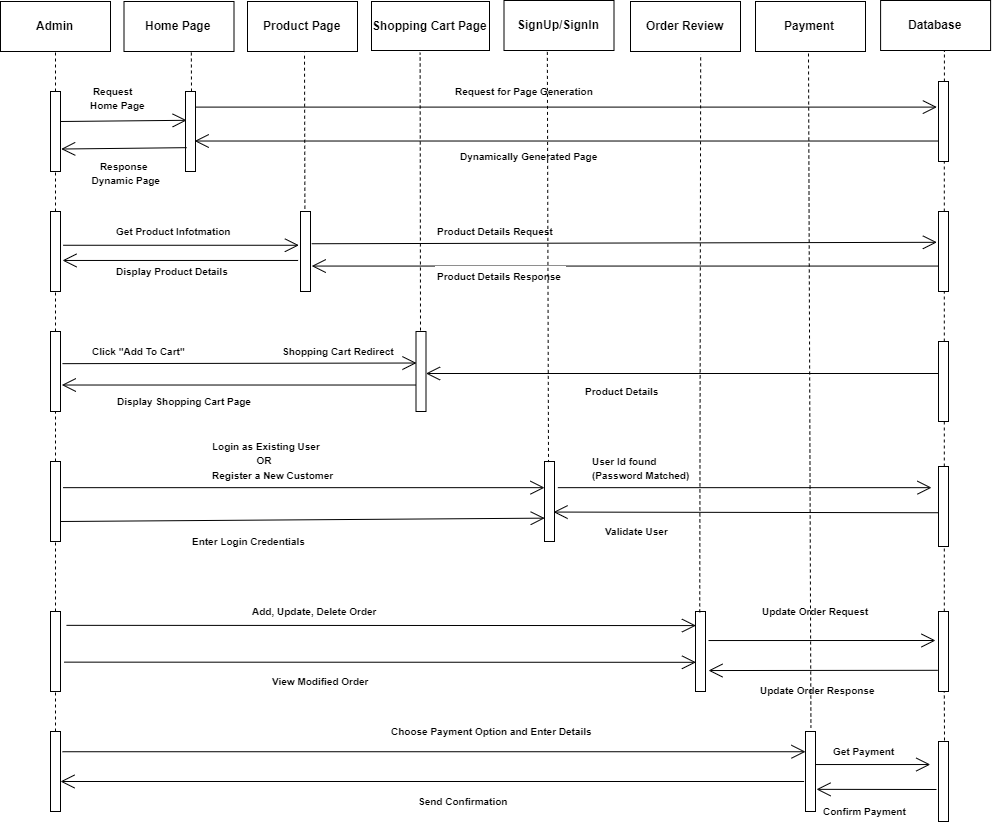


**CLASS DIAGRAM**

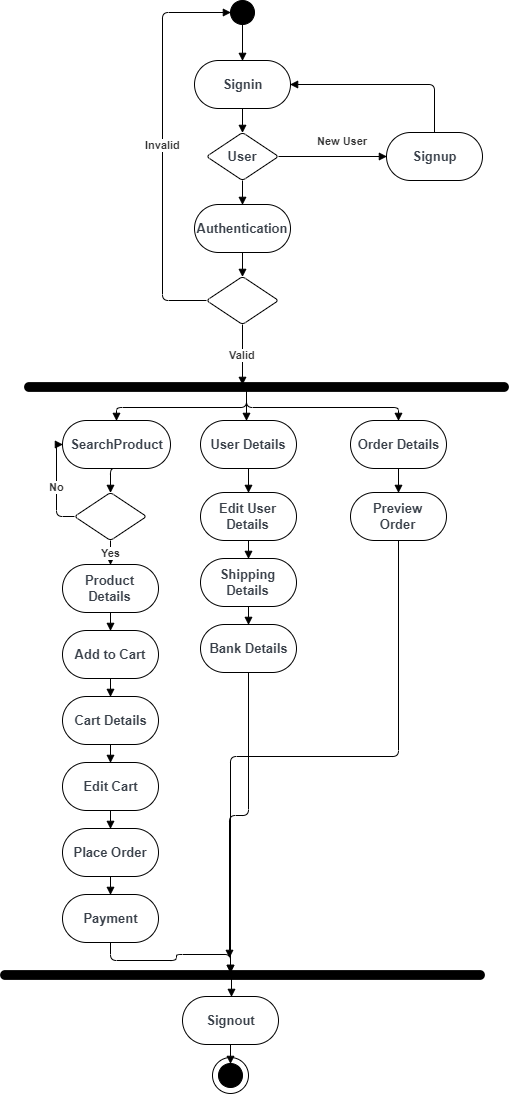


**SYSTEM DESIGN**

**SEQUENCE DIAGRAM**

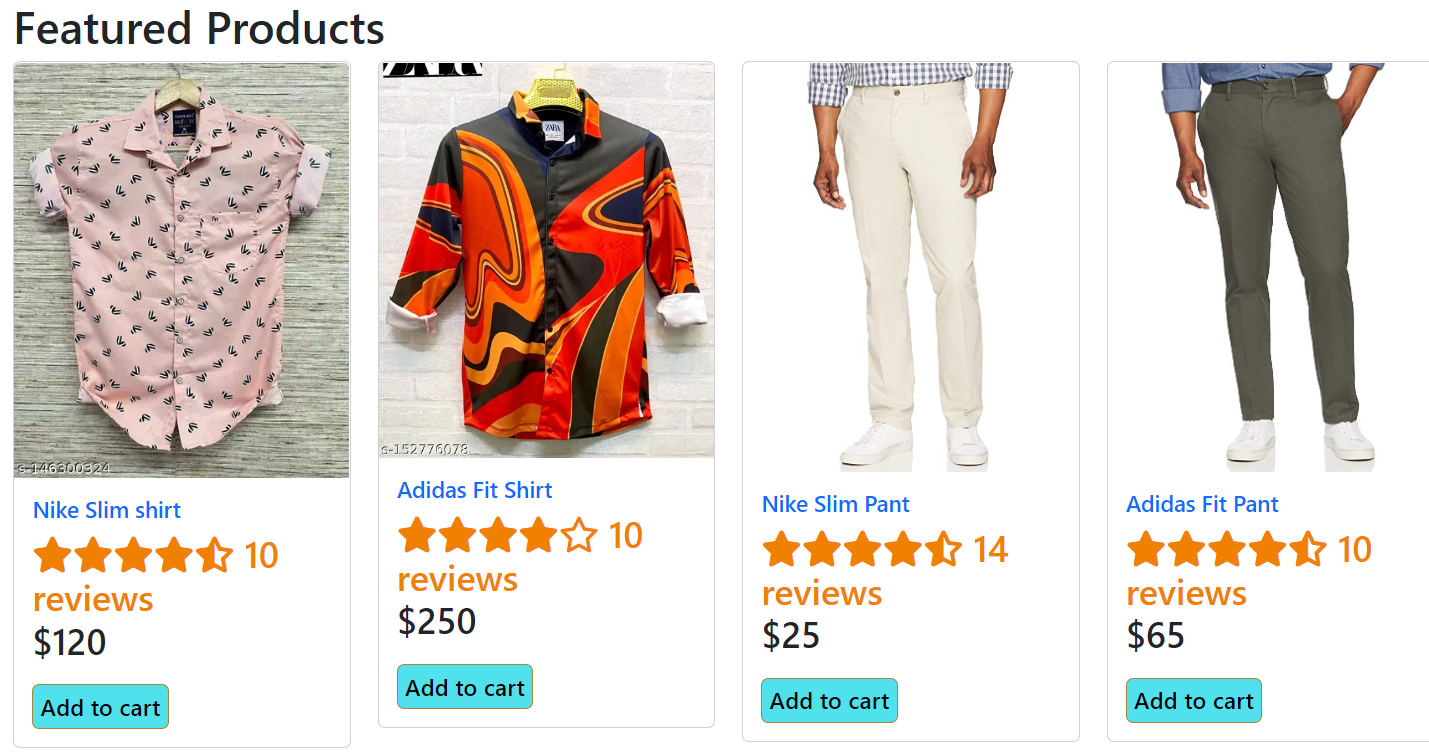
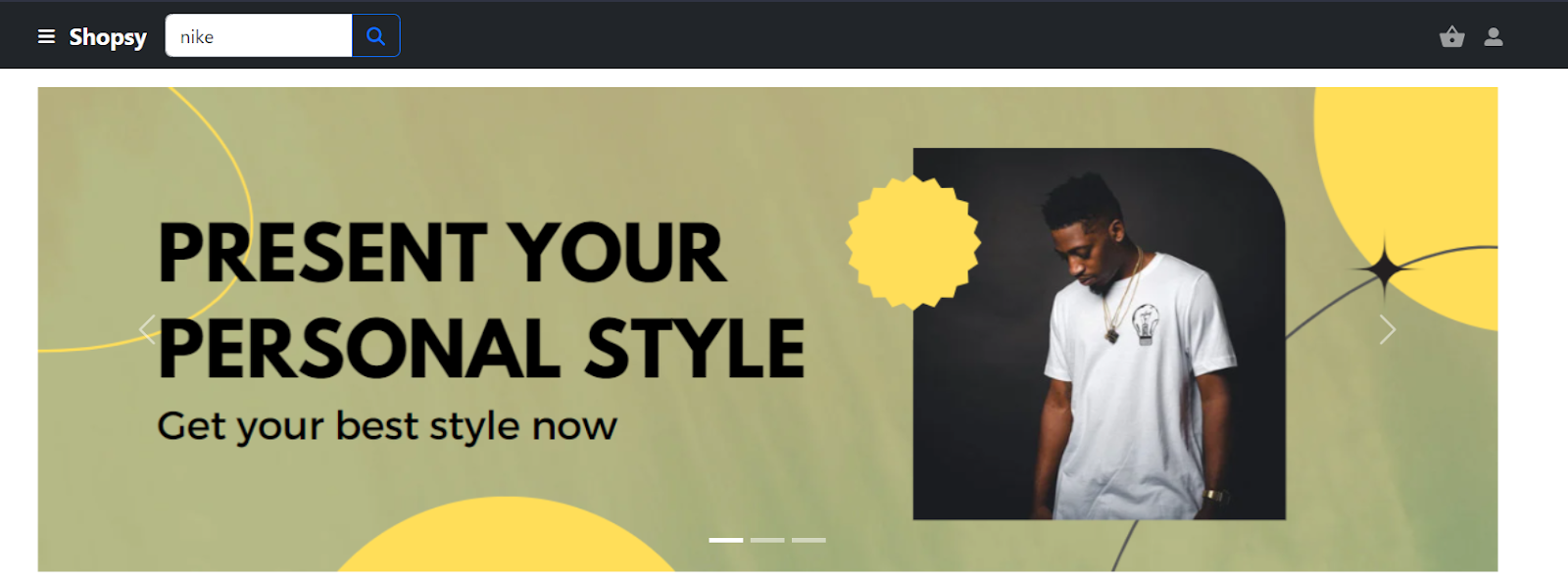


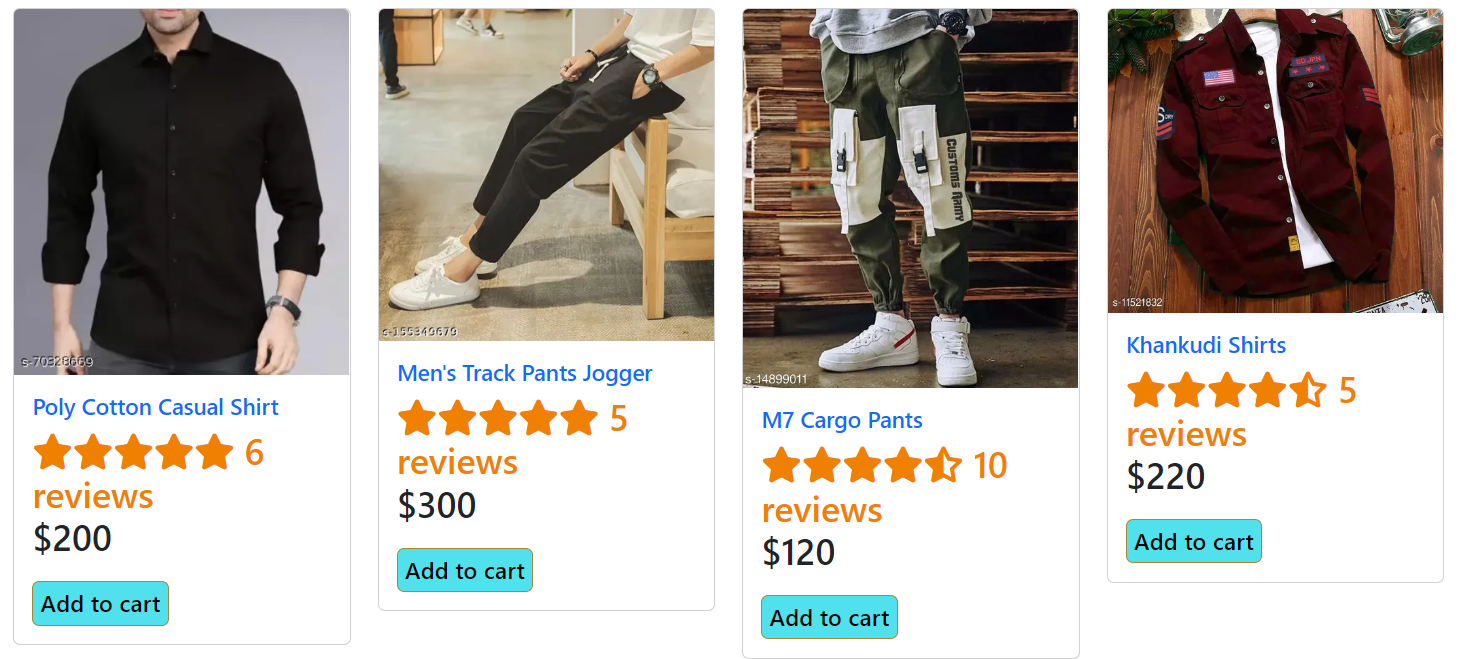
**ACTIVITY DIAGRAM**

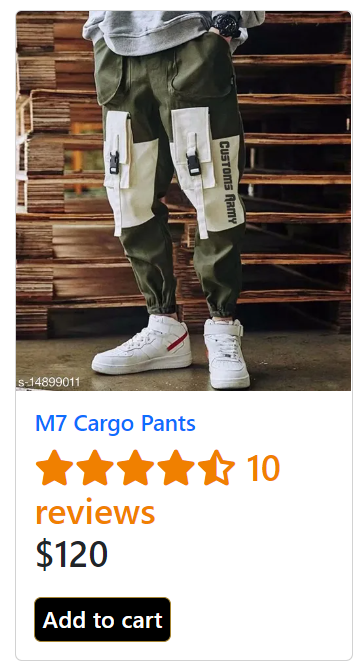


**SHOPSY ECOMMERCE STORE**

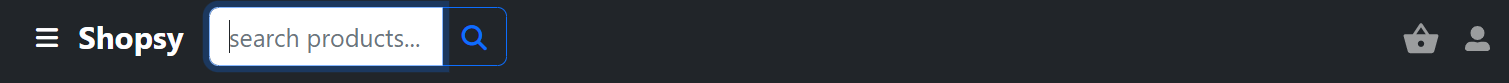
**HomeScreen**



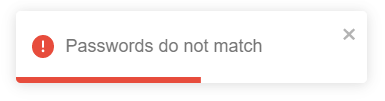
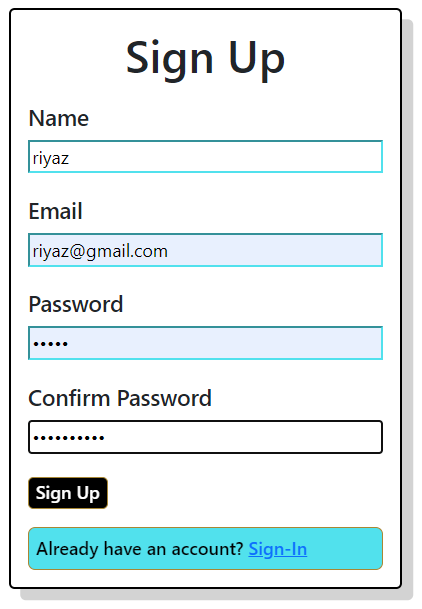




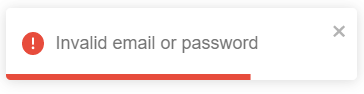
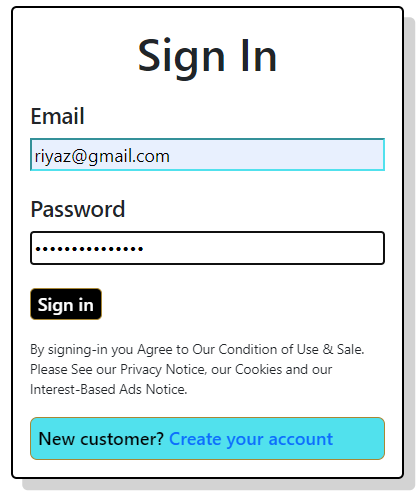
**CATEGORY**



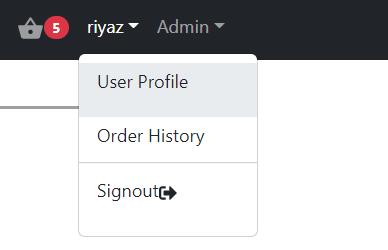
**SIGNUP**

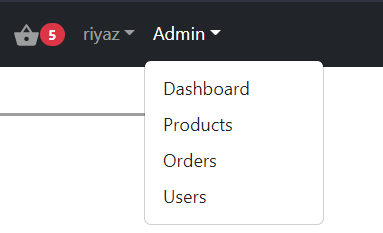


**SIGNIN**

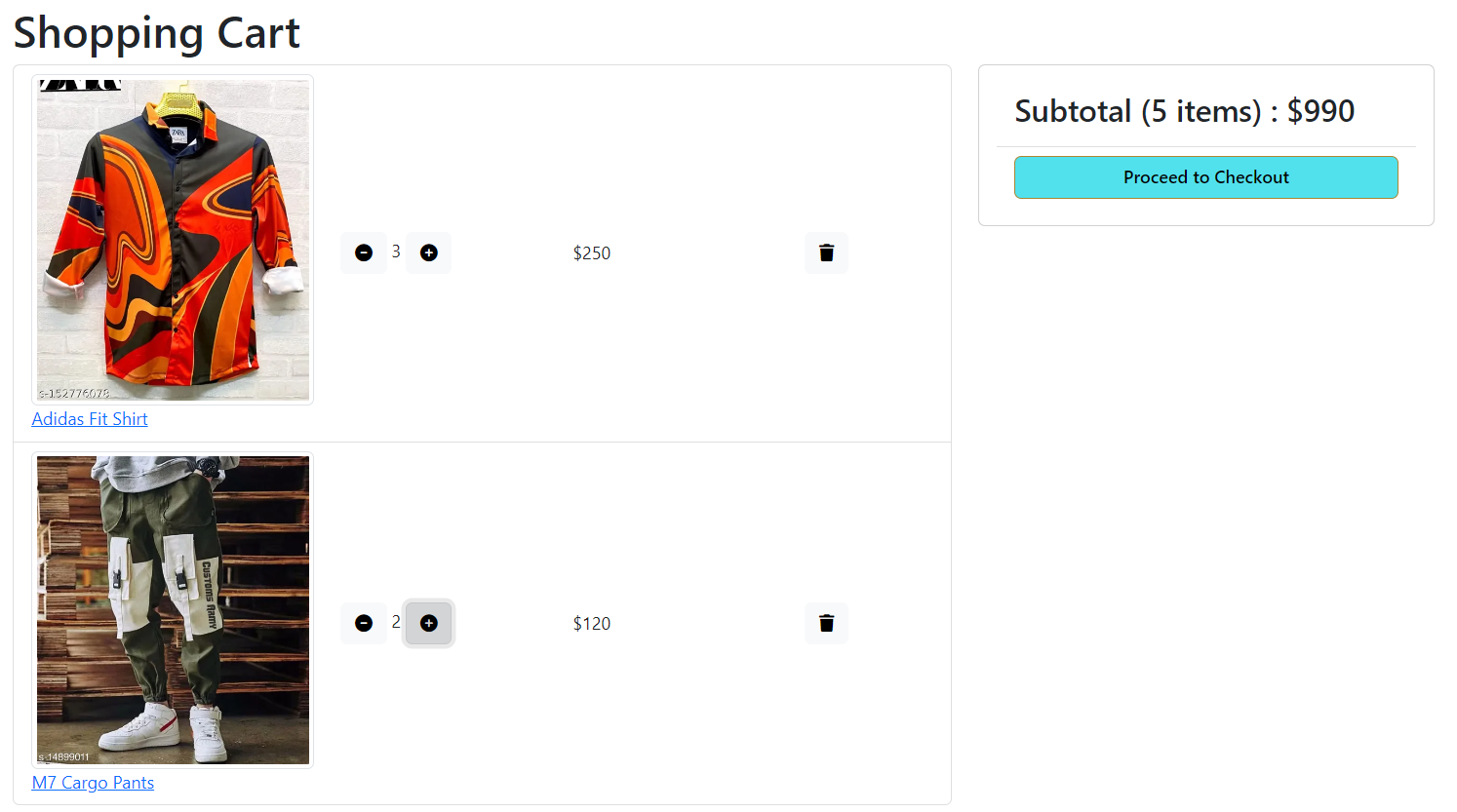


**NAVBAR**

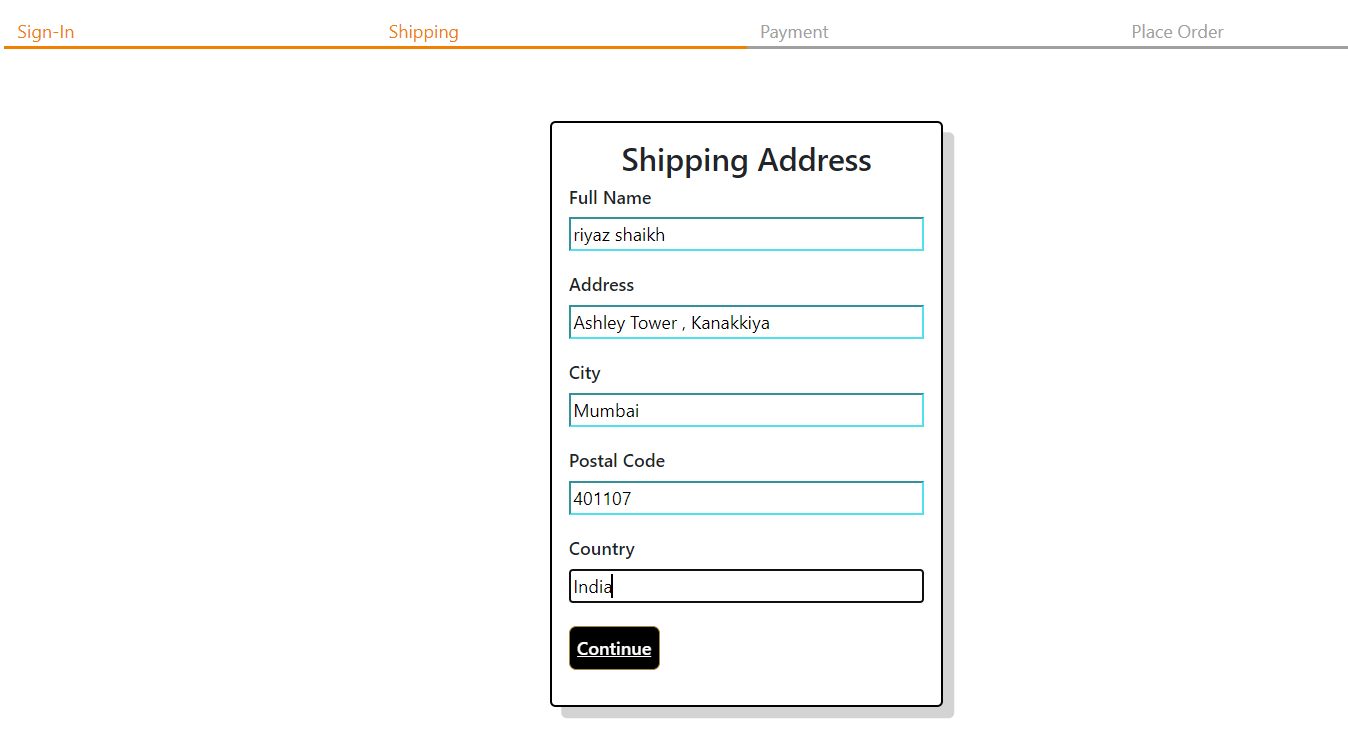




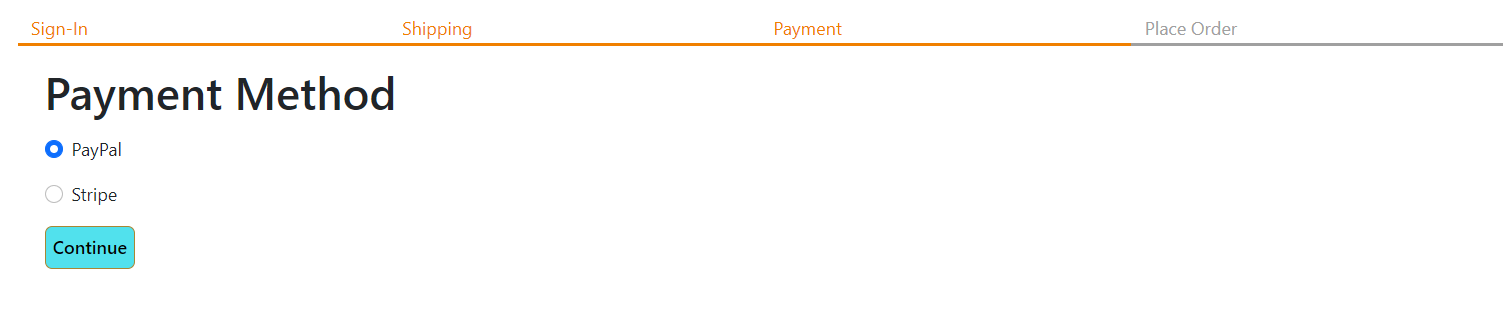
**SHOPPING CART**



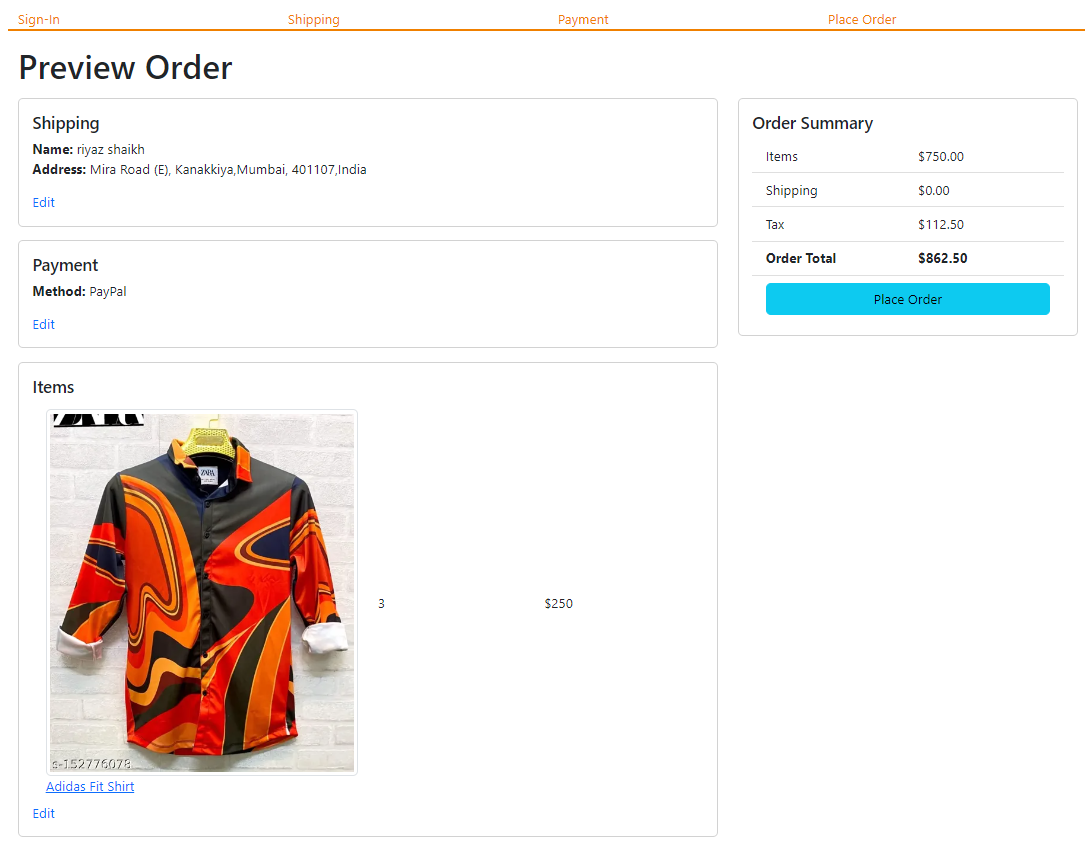
**SHOPPING ADDRESS**



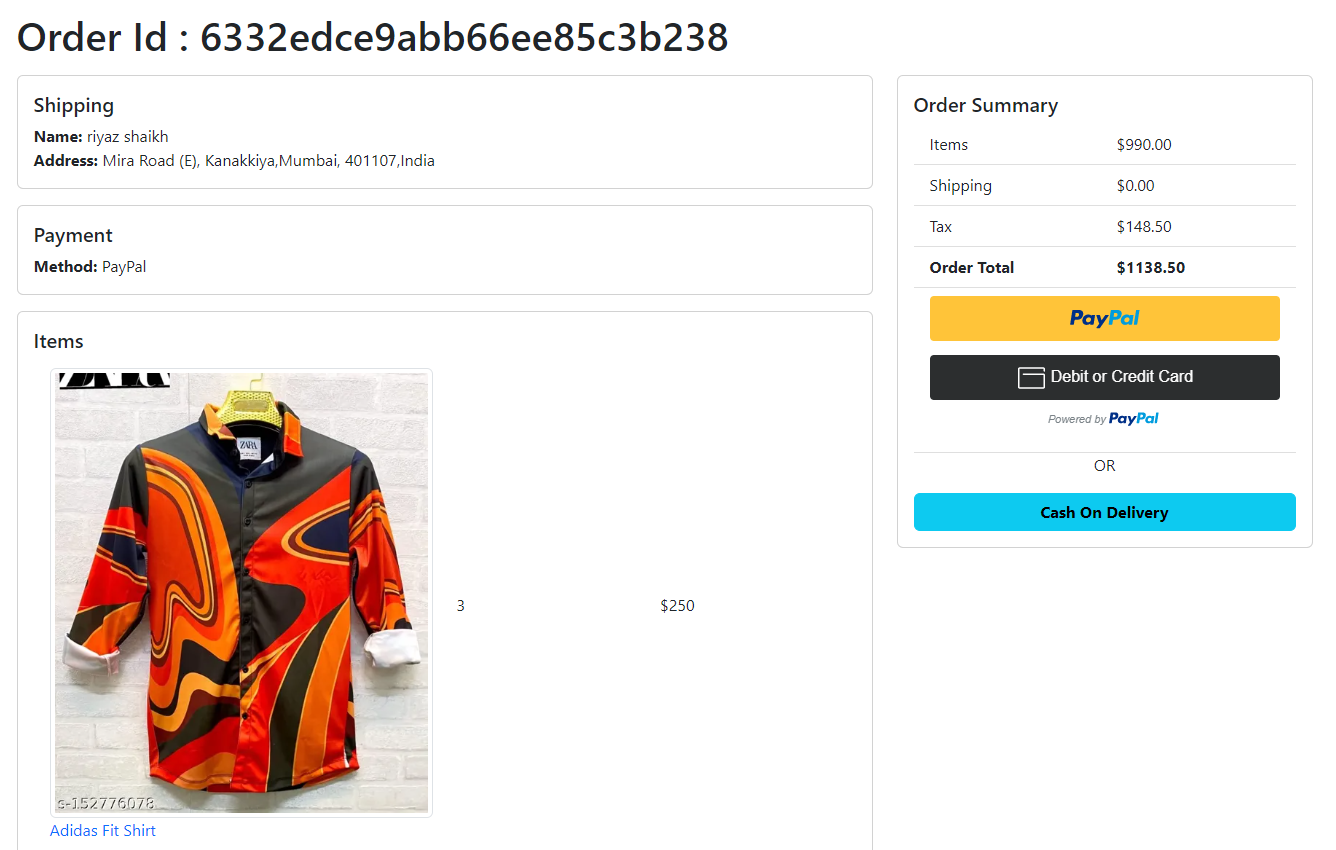
**PAYMENT METHOD**



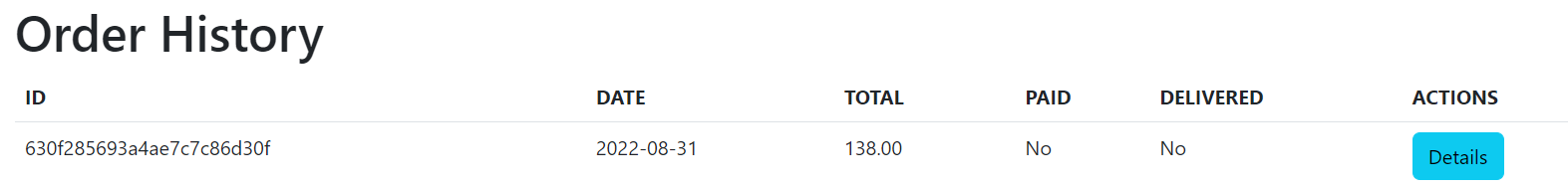
**PLACE ORDER**



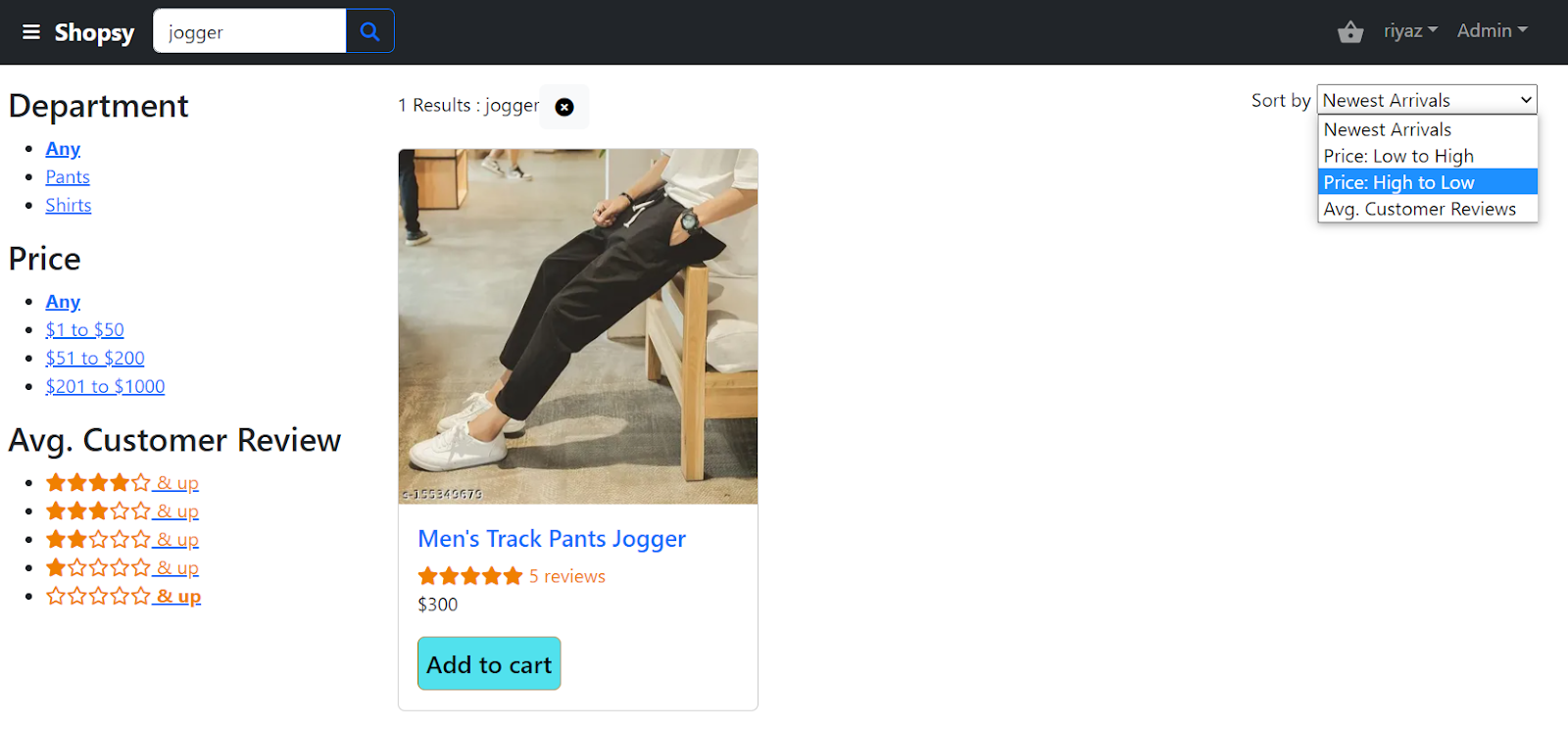
**ORDER SCREEN**



**ORDER HISTORY**



**SEARCH SCREEN**



**SYSTEM CODING**

**BACKEND**

**config.env**

PORT=4000

MONGODB\_URI=mongodb+srv://riyaz07:riyaz07@cluster0.xnw7h.mongodb.net/E-Commerce?retryWrites=true&w=majority

SECRET\_KEY=MERNSTACKDEVELOPMENT

PAYPAL\_CLIENT\_ID=AWRL7FNIrwylbEj03eZAblzwGPvKif5xnNo4OeTCQC4VFz60BoUN5Tm0h6jqUY5iJfLA5mOKDTRVglH6

**server.js**

import express from "express";

import path from "path";

import dotenv from "dotenv";

import mongoose from "mongoose";

import seedRouter from "./routes/seedRouter.js";

import productRouter from "./routes/productRouter.js";

import userRouter from "./routes/userRouter.js";

import orderRouter from "./routes/orderRouter.js";

dotenv.config();

const app = express();

const port = process.env.PORT || 5000;

mongoose

.connect(process.env.MONGODB\_URI)

.then(() => {

console.log("connected to db");

})

.catch((err) => {

console.log(err.message);

});

app.listen(port, () => {

console.log(`serve at http://localhost:${port}`);

});

app.use(express.json());

app.use(express.json());

app.use(express.urlencoded({ extended: true }));

app.get("/api/keys/paypal", (req, res) => {

res.send(process.env.PAYPAL\_CLIENT\_ID || "sb");

});

app.use("/api/seed", seedRouter);

app.use("/api/products", productRouter);

app.use("/api/users", userRouter);

app.use("/api/orders", orderRouter);

const \_\_dirname = path.resolve();

app.use(express.static(path.join(\_\_dirname, "/frontend/build")));

app.get("\*", (req, res) => {

res.sendFile(path.join(\_\_dirname, "frontend/build/index.html"));

});

app.use((err, req, res, next) => {

res.status(500).send({ message: err.message });

});

**Routes :**

**userRouter.js**

import express from "express";

import User from "../models/userModel.js";

import expressAsyncHandler from "express-async-handler";

import bcrypt from "bcryptjs";

import { generateToken, isAuth } from "../utils/util.js";

const userRouter = express.Router();

userRouter.post(

"/signin",

expressAsyncHandler(async (req, res) => {

const user = await User.findOne({ email: req.body.email });

if (user) {

if (bcrypt.compareSync(req.body.password, user.password)) {

res.send({

\_id: user.\_id,

name: user.name,

email: user.email,

isAdmin: user.isAdmin,

token: generateToken(user),

});

return;

}

}

res.status(401).send({ message: "Invalid email or password" });

})

);

userRouter.post(

"/signup",

expressAsyncHandler(async (req, res) => {

const newUser = new User({

name: req.body.name,

email: req.body.email,

password: bcrypt.hashSync(req.body.password),

});

const user = await newUser.save();

res.send({

\_id: user.\_id,

name: user.name,

email: user.email,

isAdmin: user.isAdmin,

token: generateToken(user),

});

})

);

userRouter.put(

"/profile",

isAuth,

expressAsyncHandler(async (req, res) => {

const user = await User.findById(req.user.\_id);

// if user exists then change the name and email of that..

if (user) {

user.name = req.body.name || user.name;

user.email = req.body.email || user.email;

if (req.body.password) {

user.password = bcrypt.hashSync(req.body.password, 8);

}

const updatedUser = await user.save();

res.send({

\_id: updatedUser.\_id,

name: updatedUser.name,

email: updatedUser.email,

isAdmin: updatedUser.isAdmin,

token: generateToken(updatedUser),

});

} else {

res.status(404).send({ message: "User not found" });

}

})

);

export default userRouter;

**productRouter.js**

import express from "express";

import expressAsyncHandler from "express-async-handler";

import Product from "../models/productModel.js";

import { isAuth, isAdmin } from "../utils/util.js";

const productRouter = express.Router();

productRouter.get("/", async (req, res) => {

const products = await Product.find();

res.send(products);

});

productRouter.post(

"/",

isAuth,

isAdmin,

expressAsyncHandler(async (req, res) => {

const newProduct = new Product({

name: "sample name " + Date.now(),

slug: "sample-name-" + Date.now(),

image: "/images/p1.jpg",

price: 0,

category: "sample category",

brand: "sample brand",

countInStock: 0,

rating: 0,

numReviews: 0,

description: "sample description",

});

const product = await newProduct.save();

res.send({ message: "Product Created", product });

})

);

productRouter.put(

"/:id",

isAuth,

isAdmin,

expressAsyncHandler(async (req, res) => {

const productId = req.params.id;

const product = await Product.findById(productId);

if (product) {

product.name = req.body.name;

product.slug = req.body.slug;

product.price = req.body.price;

product.image = req.body.image;

product.images = req.body.images;

product.category = req.body.category;

product.brand = req.body.brand;

product.countInStock = req.body.countInStock;

product.description = req.body.description;

await product.save();

res.send({ message: "Product Updated" });

} else {

res.status(404).send({ message: "Product Not Found" });

}

})

);

productRouter.delete(

"/:id",

isAuth,

isAdmin,

expressAsyncHandler(async (req, res) => {

const product = await Product.findById(req.params.id);

if (product) {

await product.remove();

res.send({ message: "Product Deleted" });

} else {

res.status(404).send({ message: "Product Not Found" });

}

})

);

productRouter.get(

"/admin",

isAuth,

isAdmin,

expressAsyncHandler(async (req, res) => {

const { query } = req;

const page = query.page || 1;

const pageSize = query.pageSize || PAGE\_SIZE;

const products = await Product.find()

.skip(pageSize \* (page - 1))

.limit(pageSize);

const countProducts = await Product.countDocuments();

res.send({

products,

countProducts,

page,

pages: Math.ceil(countProducts / pageSize),

});

})

);

productRouter.get(

"/categories",

expressAsyncHandler(async (req, res) => {

const categories = await Product.find().distinct("category");

res.send(categories);

})

);

productRouter.get("/slug/:slug", async (req, res) => {

const product = await Product.findOne({ slug: req.params.slug });

if (product) {

res.send(product);

} else {

res.status(404).send({ message: "Product Not Found" });

}

});

productRouter.get("/:id", async (req, res) => {

const product = await Product.findById(req.params.id);

if (product) {

res.send(product);

} else {

res.status(404).send({ message: "Product Not Found" });

}

});

export default productRouter;

**orderRouter.js**

import express from "express";

import expressAsyncHandler from "express-async-handler";

import Order from "../models/orderModel.js";

import User from "../models/userModel.js";

import Product from "../models/productModel.js";

import { isAuth, isAdmin } from "../utils/util.js";

const orderRouter = express.Router();

orderRouter.get(

"/",

isAuth,

isAdmin,

expressAsyncHandler(async (req, res) => {

const orders = await Order.find().populate("user", "name");

res.send(orders);

})

);

orderRouter.post(

"/",

isAuth,

expressAsyncHandler(async (req, res) => {

const newOrder = new Order({

orderItems: req.body.orderItems.map((x) => ({ ...x, product: x.\_id })),

shippingAddress: req.body.shippingAddress,

paymentMethod: req.body.paymentMethod,

itemsPrice: req.body.itemsPrice,

shippingPrice: req.body.shippingPrice,

taxPrice: req.body.taxPrice,

totalPrice: req.body.totalPrice,

user: req.user.\_id,

});

const order = await newOrder.save();

res.status(201).send({ message: "New Order Created", order });

})

);

orderRouter.get(

"/mine",

isAuth,

expressAsyncHandler(async (req, res) => {

const orders = await Order.find({ user: req.user.\_id });

res.send(orders);

})

);

orderRouter.get(

"/:id",

isAuth,

expressAsyncHandler(async (req, res) => {

const order = await Order.findById(req.params.id);

if (order) {

res.send(order);

} else {

res.status(404).send({ message: "Order Not Found" });

}

})

);

orderRouter.put(

"/:id/deliver",

isAuth,

expressAsyncHandler(async (req, res) => {

const order = await Order.findById(req.params.id);

if (order) {

order.isDelivered = true;

order.deliveredAt = Date.now();

await order.save();

res.send({ message: "Order Delivered" });

} else {

res.status(404).send({ message: "Order Not Found" });

}

})

);

);

orderRouter.delete(

"/:id",

isAuth,

isAdmin,

expressAsyncHandler(async (req, res) => {

const order = await Order.findById(req.params.id);

if (order) {

await order.remove();

res.send({ message: "Order Deleted" });

} else {

res.status(404).send({ message: "Order Not Found" });

}

})

);

export default orderRouter;

**Models :**

**userModel.js**

import mongoose from "mongoose";

const userSchema = new mongoose.Schema(

{

name: { type: String, required: true },

email: { type: String, required: true, unique: true },

password: { type: String, required: true },

isAdmin: { type: Boolean, default: false, required: true },

},

{

timestamps: true,

}

);

const User = mongoose.model("User", userSchema);

export default User;

**productModel.js**

import mongoose from "mongoose";

const productSchema = new mongoose.Schema(

{

name: {

type: String,

required: true,

unique: true,

},

slug: { type: String, required: true, unique: true },

image: { type: String, required: true },

images: [String],

brand: { type: String, required: true },

category: { type: String, required: true },

description: { type: String, required: true },

price: { type: Number, required: true },

countInStock: { type: Number, required: true },

rating: { type: Number, required: true },

numReviews: { type: Number, required: true },

},

{

timestamps: true,

}

);

const Product = mongoose.model("Product", productSchema);

export default Product;

**orderModel.js**

import mongoose from 'mongoose';

const orderSchema = new mongoose.Schema(

{

orderItems: [

{

slug: { type: String, required: true },

name: { type: String, required: true },

quantity: { type: Number, required: true },

image: { type: String, required: true },

price: { type: Number, required: true },

product: {

type: mongoose.Schema.Types.ObjectId,

ref: 'Product',

required: true,

},

},

],

shippingAddress: {

fullName: { type: String, required: true },

address: { type: String, required: true },

city: { type: String, required: true },

postalCode: { type: String, required: true },

country: { type: String, required: true },

location: {

lat: Number,

lng: Number,

address: String,

name: String,

vicinity: String,

googleAddressId: String,

},

},

paymentMethod: { type: String, required: true },

paymentResult: {

id: String,

status: String,

update\_time: String,

email\_address: String,

},

itemsPrice: { type: Number, required: true },

shippingPrice: { type: Number, required: true },

taxPrice: { type: Number, required: true },

totalPrice: { type: Number, required: true },

user: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },

isPaid: { type: Boolean, default: false },

paidAt: { type: Date },

isDelivered: { type: Boolean, default: false },

deliveredAt: { type: Date },

},

{

timestamps: true,

}

);

const Order = mongoose.model('Order', orderSchema);

export default Order;

**FRONTEND**

**index.js**

import React from "react";

import ReactDOM from "react-dom/client";

import "./index.css";

import { HelmetProvider } from "react-helmet-async";

import "bootstrap/dist/css/bootstrap.min.css";

import { StoreProvider } from "./Store";

import { PayPalScriptProvider } from "@paypal/react-paypal-js";

import App from "./App";

const root = ReactDOM.createRoot(document.getElementById("root"));

root.render(

<React.StrictMode>

<StoreProvider>

<HelmetProvider>

<PayPalScriptProvider deferLoading={true}>

<App />

</PayPalScriptProvider>

</HelmetProvider>

</StoreProvider>

</React.StrictMode>

);

**Store.js**

import { createContext, useReducer } from "react";

export const Store = createContext();

const initialState = {

fullBox: false,

userInfo: localStorage.getItem("userInfo")

? JSON.parse(localStorage.getItem("userInfo"))

: null,

case "CART\_REMOVE\_ITEM": {

const cartItems = state.cart.cartItems.filter(

(item) => item.\_id !== action.payload.\_id

);

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

return { ...state, cart: { ...state.cart, cartItems } };

}

case "CART\_CLEAR":

return { ...state, cart: { ...state.cart, cartItems: [] } };

case "USER\_SIGNIN":

return { ...state, userInfo: action.payload };

case "USER\_SIGNOUT":

return {

...state,

userInfo: null,

cart: {

cartItems: [],

shippingAddress: {},

paymentMethod: "",

},

};

default:

return state;

}

}

export function StoreProvider(props) {

const [state, dispatch] = useReducer(reducer, initialState);

const value = { state, dispatch };

return <Store.Provider value={value}>{props.children} </Store.Provider>;

}

**index.css**

.navbar-brand {

font-weight: bold;

}.side-navbar {

width: 240px;

height: 100%;

position: absolute;

left: -300px;

background-color: #100901;

transition: 0.5s;

}

.site-container {

transition: 0.4s;

}

.active-cont {

margin-left: 240px;

}

.active-nav {

left: 0;

}

.text-bold {

font-weight: bold;

}

.login {

margin-top: 0;

margin-top: auto;

display: flex;

flex-direction: column;

align-items: center;

box-sizing: border-box;

box-shadow: 0, 0, 0, 0.5;

background-color: white;

height: 100vh;

}

.login\_signIn {

display: flex;

align-items: center;

justify-content: center;

background: #51E1ED;

font-weight: 500;

border: 1px solid;

border-color: #a88734;

color: black;

border-radius: 0.3rem;

transition: 1s ease all;

}

.login\_container {

width: 350px;

margin-top: 4rem;

display: flex;

flex-direction: column;

border-radius: 5px;

padding: 15px;

box-sizing: border-box;

box-shadow: 10px lightgray;

border: 2px solid black;

}

.login\_container h1 {

margin-bottom: 20px;

}

.login\_container>form>h5 {

margin-bottom: 5px;

font-weight: 550;

}

.login\_container form input:hover {

width: 95%;

}

.login\_register\_button {

background: #51E1ED;

border: 1px solid;

padding: 0.4rem;

border-color: #a88734;

border-radius: 0.4rem;

color: #000000;

font-weight: 500;

transition: 1s ease all;

width: auto;

}

.login\_signIn:hover {

background-color: #000000;

color: #ffffff;

}

.home\_image {

width: 100%;

z-index: -1;

margin-bottom: -150px;

mask-image: linear-gradient(to bottom, rgba(0, 0, 0, 1), rgba(0, 0, 0, 0));

}

.product {

background-color: white;

display: flex;

flex-direction: column;

align-items: center;

justify-content: flex-end;

width: 100%;

min-width: 100px;

max-height: 800px;

padding: 15px;

margin: 10px;

z-index: 1;

box-sizing: border-box;

box-shadow: 10px lightgray;

transition: 0.6s all;

}

.product:hover {

box-shadow: 2px 20px 0 rgba(0, 0, 0, 0.3);

transform: translateY(-20px);

}

.product img {

max-height: 200px;

width: 100%;

object-fit: contain;

margin-bottom: 15px;

}

.product button {

display: flex;

align-items: center;

justify-content: center;

background: #f0c14b;

border: 1px solid;

margin-top: 10px;

border-color: #a88734;

color: black;

}

**App.js**

import "./App.css";

import { useState, useEffect } from "react";

import { BrowserRouter, Link, Route, Routes } from "react-router-dom";

import { toast, ToastContainer } from "react-toastify";

import "react-toastify/dist/ReactToastify.css";

import HomeScreen from "./screens/HomeScreen";

import ProductScreen from "./screens/ProductScreen";

import Navbar from "react-bootstrap/Navbar";

import Container from "react-bootstrap/Container";

import { LinkContainer } from "react-router-bootstrap";

import Badge from "react-bootstrap/Badge";

import Nav from "react-bootstrap/Nav";

import NavDropdown from "react-bootstrap/NavDropdown";

import { Store } from "./Store";

import { useContext } from "react";

import SearchBox from "./components/SearchBox";

import SigninScreen from "./screens/SigninScreen";

import ShippingAddressScreen from "./screens/ShippingAddressScreen";

import SignupScreen from "./screens/SignupScreen";

import PaymentScreen from "./screens/PaymentScreen";

import PlaceOrderScreen from "./screens/PlaceOrderScreen";

import OrderScreen from "./screens/OrderScreen";

import OrderHistoryScreen from "./screens/OrderHistoryScreen";

import ProfileScreen from "./screens/ProfileScreen";

import Button from "react-bootstrap/Button";

import { getError } from "./utils";

import axios from "axios";

import CartScreen from "./screens/CartScreen";

import SearchScreen from "./screens/SearchScreen";

import ProtectedRoute from "./components/ProtectedRoute";

function App() {

return (

<BrowserRouter>

<div

className={

sidebarIsOpen

? "d-flex flex-column site-container active-cont"

: "d-flex flex-column site-container"

}

>

<ToastContainer position="bottom-center" limit={1} />

<header>

<Navbar bg="dark" variant="dark" expand="lg">

<Container>

<Button

variant="dark"

onClick={() => setSidebarIsOpen(!sidebarIsOpen)}

>

<i className="fas fa-bars"></i>

</Button>

<LinkContainer to="/">

<Navbar.Brand>Shopsy</Navbar.Brand>

</LinkContainer>

<Navbar.Toggle aria-controls="basic-navbar-nav" />

<Navbar.Collapse id="basic-navbar-nav">

<SearchBox />

<Nav className="me-auto w-100 justify-content-end">

<Link to="/cart" className="nav-link">

<ShoppingBasket />

{cart.cartItems.length > 0 && (

<Badge pill bg="danger">

{cart.cartItems.reduce((a, c) => a + c.quantity, 0)}

</Badge>

)}

</Link>

{userInfo ? (

<NavDropdown title={userInfo.name} id="basic-nav-dropdown">

<LinkContainer to="/profile">

<NavDropdown.Item>

<p>User Profile</p>

</NavDropdown.Item>

</LinkContainer>

<LinkContainer to="/orderhistory">

<NavDropdown.Item>Order History</NavDropdown.Item>

</LinkContainer>

<NavDropdown.Divider />

<Link

className="dropdown-item"

to="#signout"

onClick={signoutHandler}

>

<p>

Signout

<FaSignOutAlt />

</p>

</Link>

</NavDropdown>

) : (

<Link className="nav-link" to="/signin">

<FaUserAlt />

</Link>

)}

{userInfo && userInfo.isAdmin && (

<NavDropdown title="Admin" id="admin-nav-dropdown">

<LinkContainer to="/admin/dashboard">

<NavDropdown.Item>Dashboard</NavDropdown.Item>

</LinkContainer>

<LinkContainer to="/admin/products">

<NavDropdown.Item>Products</NavDropdown.Item>

</LinkContainer>

<LinkContainer to="/admin/orders">

<NavDropdown.Item>Orders</NavDropdown.Item>

</LinkContainer>

<LinkContainer to="/admin/users">

<NavDropdown.Item>Users</NavDropdown.Item>

</LinkContainer>

</NavDropdown>

)}

</Nav>

</Navbar.Collapse>

</Container>

</Navbar>

</header>

<div

className={

sidebarIsOpen

? "active-nav side-navbar d-flex justify-content-between flex-wrap flex-column"

: "side-navbar d-flex justify-content-between flex-wrap flex-column"

}

>

<Nav className="flex-column text-white w-100 p-2">

<Nav.Item>

<strong>Categories</strong>

</Nav.Item>

{categories.map((category) => (

<Nav.Item key={category}>

<LinkContainer

to={`/search?category=${category}`}

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

>

<Nav.Link>{category}</Nav.Link>

</LinkContainer>

</Nav.Item>

))}

</Nav>

</div>

<main className="mt-3">

<Container>

<Routes>

<Route path="/" element={<HomeScreen />} />

<Route path="/cash" element={<CashOnDelivery />} />

<Route path="/cart" element={<CartScreen />} />

<Route path="/signin" element={<SigninScreen />} />

<Route path="/signup" element={<SignupScreen />} />

<Route path="/search" element={<SearchScreen />} />

<Route path="/shipping" element={<ShippingAddressScreen />} />

<Route path="/payment" element={<PaymentScreen />} />

<Route

path="/profile"

element={

<ProtectedRoute>

<ProfileScreen />

</ProtectedRoute>

}

/>

<Route path="/placeorder" element={<PlaceOrderScreen />} />

<Route path="/product/:slug" element={<ProductScreen />} />

<Route

path="/order/:id"

element={

<ProtectedRoute>

<OrderScreen />

</ProtectedRoute>

}

/>

<Route

path="/orderhistory"

element={

<ProtectedRoute>

<OrderHistoryScreen />

</ProtectedRoute>

}

/>

<Route

path="/admin/dashboard"

element={

<AdminRoute>

<DashboardScreen />

</AdminRoute>

}

/>

</Routes>

</Container>

</main>

<footer>

<div className="bg-color-black text-center">

All rights reserved @2022

</div>

</footer>

</div>

</BrowserRouter>

);

}

export default App;

**Screens**

**Carousel.js**

import React from "react";

import Carousel from "react-bootstrap/Carousel";

import banner1 from "../utils/banner1.png";

import banner2 from "../utils/banner2.png";

import banner3 from "../utils/banner3.png";

const Carousel1 = () => {

return (

<Carousel>

<Carousel.Item>

<img className="d-block w-100" src={banner1} alt="First slide" />

</Carousel.Item>

<Carousel.Item>

<img className="d-block w-100" src={banner2} alt="Second slide" />

</Carousel.Item>

<Carousel.Item>

<img className="d-block w-100" src={banner3} alt="Third slide" />

</Carousel.Item>

</Carousel>

);

};

export default Carousel1;

**CartScreen.js**

import { useContext } from "react";

import { Store } from "../Store";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import MessageBox from "../components/MessageBox";

import { Link, useNavigate } from "react-router-dom";

import ListGroup from "react-bootstrap/esm/ListGroup";

import Button from "react-bootstrap/esm/Button";

import Card from "react-bootstrap/esm/Card";

import { Helmet } from "react-helmet-async";

import axios from "axios";

export default function CartScreen() {

return (

<div>

<Helmet>

<title>Shopping Cart</title>

</Helmet>

<h1>Shopping Cart</h1>

<Row>

<Col md={8}>

{cartItems.length === 0 ? (

<MessageBox>

Cart is Empty <Link to="/">Go to Shopping</Link>

</MessageBox>

)

</ListGroup>

)}{" "}

</Col>

<Col md={4}>

<Card>

<Card.Body>

<ListGroup variant="flush">

<ListGroup.Item>

<h3>

Subtotal ({cartItems.reduce((a, c) => a + c.quantity, 0)}{" "}

items) : $

{cartItems.reduce((a, c) => a + c.price \* c.quantity, 0)}

</h3>

</ListGroup.Item>

<ListGroup.Item>

<div className="d-grid">

<button

className="login\_register\_button"

style={{ textDecoration: "none" }}

type="button"

variant="info"

onClick={checkoutHandler}

disabled={cartItems.length === 0}

>

Proceed to Checkout

</button>

</div>

</ListGroup.Item>

</ListGroup>

</Card.Body>

</Card>

</Col>

</Row>

</div>

);

}

**CashOnDelivery.js**

import React from "react";

import { Helmet } from "react-helmet-async";

const CashOnDelivery = () => {

return (

<div>

<Helmet>

<title>Cash On Delivery</title>

</Helmet>

<h2>Thanks for ordering the product...</h2>

</div>

);

};

export default CashOnDelivery;

**DashboardScreen.js**

import React, { useReducer, useEffect } from "react";

import axios from "axios";

import logger from "use-reducer-logger";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import Product from "../components/Product";

import { Helmet } from "react-helmet-async";

import LoadingBox from "../components/LoadingBox";

import MessageBox from "../components/MessageBox";

return (

<div>

{/\* this is used for title \*/}

<Helmet>

<title>Amazon</title>

</Helmet>

<h1>Featured Products</h1>

<h2>

<div className="products">

{loading ? (

<LoadingBox />

) : error ? (

<MessageBox variant="danger">{error}</MessageBox>

) : (

<Row>

{products.map((product) => (

<Col key={product.slug} sm={6} md={4} lg={3} className="mb-3">

<Product product={product}></Product>

</Col>

))}

</Row>

)}

</div>

</h2>

</div>

);

};

export default DashboardScreen;

**HomeScreen.js**

import React, { useEffect, useReducer } from "react";

import axios from "axios";

import logger from "use-reducer-logger";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import Product from "../components/Product";

import { Helmet } from "react-helmet-async";

import LoadingBox from "../components/LoadingBox";

import MessageBox from "../components/MessageBox";

import Carousel from "./Carousel";

const reducer = (state, action) => {

switch (action.type) {

case "FETCH\_REQUEST":

return { ...state, loading: true };

case "FETCH\_SUCCESS":

return { ...state, products: action.payload, loading: false };

case "FETCH\_FAIL":

return { ...state, loading: false, error: action.payload };

default:

return state;

}

};

const HomeScreen = () => {

return (

<div className="container-fluid">

<Carousel />

<h1>Featured Products</h1>

<h2>

<div className="products">

{loading ? (

<LoadingBox />

) : error ? (

<MessageBox variant="danger">{error}</MessageBox>

)

</div>

);};

export default HomeScreen;

**OrderHistory.js**

import React, { useContext, useEffect, useReducer } from "react";

import { Helmet } from "react-helmet-async";

import axios from "axios";

import { useNavigate } from "react-router-dom";

import LoadingBox from "../components/LoadingBox";

import MessageBox from "../components/MessageBox";

import { Store } from "../Store";

import { getError } from "../utils";

import Button from "react-bootstrap/esm/Button";

const OrderHistoryScreen = () => {

return (

<div>

<Helmet>

<title>Order History</title>

</Helmet>

<h1>Order History</h1>

{loading ? (

<LoadingBox></LoadingBox>

) : error ? (

<MessageBox variant="danger">{error}</MessageBox>

) : (

<table className="table">

<thead>

<tr>

<th>ID</th>

<th>DATE</th>

<th>TOTAL</th>

<th>PAID</th>

<th>DELIVERED</th>

<th>ACTIONS</th>

</tr>

</thead>

<tbody>

{orders.map((order) => (

<tr key={order.\_id}>

<td>{order.\_id}</td>

<td>{order.createdAt.substring(0, 10)}</td>

<td>{order.totalPrice.toFixed(2)}</td>

<td>{order.isPaid ? order.paidAt.substring(0, 10) : "No"}</td>

<td>

{order.isDelivered

? order.deliveredAt.substring(0, 10)

: "No"}

</td>

<td>

<Button

type="button"

variant="info"

onClick={() => {

navigate(`/order/${order.\_id}`);

}}

>

Details

</Button>

</td>

</tr>

))}

</tbody>

</table>

)}

</div>

);

};

export default OrderHistoryScreen;

**OrderScreen.js**

import axios from "axios";

import React, { useContext, useEffect, useReducer } from "react";

import { PayPalButtons, usePayPalScriptReducer } from "@paypal/react-paypal-js";

import { Helmet } from "react-helmet-async";

import { NavLink, useNavigate, useParams } from "react-router-dom";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import Button from "react-bootstrap/Button";

import ListGroup from "react-bootstrap/ListGroup";

import Card from "react-bootstrap/Card";

import { Link } from "react-router-dom";

import LoadingBox from "../components/LoadingBox";

import MessageBox from "../components/MessageBox";

import { Store } from "../Store";

import { getError } from "../utils";

import { toast } from "react-toastify";

const reducer = (state, action) => {

switch (action.type) {

case "FETCH\_REQUEST":

return { ...state, loading: true, error: "" };

case "FETCH\_SUCCESS":

return { ...state, loading: false, order: action.payload, error: "" };

case "FETCH\_FAIL":

return { ...state, loading: false, error: action.payload };

case "PAY\_REQUEST":

return { ...state, loadingPay: true };

case "PAY\_SUCCESS":

return { ...state, loadingPay: false, successPay: true };

case "PAY\_FAIL":

return { ...state, loadingPay: false };

case "PAY\_RESET":

return { ...state, loadingPay: false, successPay: false };

case "DELIVER\_REQUEST":

return { ...state, loadingDeliver: true };

case "DELIVER\_SUCCESS":

return { ...state, loadingDeliver: false, successDeliver: true };

case "DELIVER\_FAIL":

return { ...state, loadingDeliver: false };

case "DELIVER\_RESET":

return {

...state,

loadingDeliver: false,

successDeliver: false,

};

default:

return state;

}

};

const OrderScreen = () => {

const navigate = useNavigate();

const { state } = useContext(Store);

const { userInfo } = state;

const params = useParams();

const { id: orderId } = params;

const [

{

loading,

error,

order,

successPay,

loadingPay,

loadingDeliver,

successDeliver,

},

dispatch,

] = useReducer(reducer, {

loading: true,

order: {},

error: "",

successPay: false,

loadingPay: false,

});

const [{ isPending }, paypalDispatch] = usePayPalScriptReducer();

function createOrder(data, actions) {

return actions.order

.create({

purchase\_units: [

{

amount: { value: order.totalPrice },

},

],

})

.then((orderID) => {

return orderID;

});

}

function onApprove(data, actions) {

// if order is successfull then do this..

return actions.order.capture().then(async function (details) {

try {

dispatch({ type: "PAY\_REQUEST" });

const { data } = await axios.put(

`/api/orders/${order.\_id}/pay`,

details,

{

headers: { authorization: `Bearer ${userInfo.token}` },

}

);

dispatch({ type: "PAY\_SUCCESS", payload: data });

toast.success("Order is paid");

} catch (err) {

dispatch({ type: "PAY\_FAIL", payload: getError(err) });

toast.error(getError(err));

}

});

}

function onError(err) {

toast.error(getError(err));

}

return loading ? (

<LoadingBox></LoadingBox>

) : error ? (

<MessageBox variant="danger">{error}</MessageBox>

) : (

<div>

<Helmet>

<title>Order {orderId}</title>

</Helmet>

<h1 className="my-3">Order Id : {orderId}</h1>

<Row>

<Col md={8}>

<Card className="mb-3">

<Card.Body>

<Card.Title>Shipping</Card.Title>

<Card.Text>

<strong>Name:</strong> {order.shippingAddress.fullName} <br />

<strong>Address: </strong> {order.shippingAddress.address},

{order.shippingAddress.city}, {order.shippingAddress.postalCode}

,{order.shippingAddress.country}

</Card.Text>

</Card.Body>

</Card>

<Card className="mb-3">

<Card.Body>

<Card.Title>Payment</Card.Title>

<Card.Text>

<strong>Method:</strong> {order.paymentMethod}

</Card.Text>

</Card.Body>

</Card>

<Card className="mb-3">

<Card.Body>

<Card.Title>Items</Card.Title>

<ListGroup variant="flush">

{order.orderItems.map((item) => (

<ListGroup.Item key={item.\_id}>

<Row className="align-items-center">

<Col md={6}>

<img

src={item.image}

alt={item.name}

className="img-fluid rounded img-thumbnail"

></img>{" "}

<Link

style={{ textDecoration: "none" }}

to={`/product/${item.slug}`}

>

{item.name}

</Link>

</Col>

<Col md={3}>

<span>{item.quantity}</span>

</Col>

<Col md={3}>${item.price}</Col>

</Row>

</ListGroup.Item>

))}

</ListGroup>

</Card.Body>

</Card>

</Col>

<Card className="mb-3">

<Card.Body>

<Card.Title>Items</Card.Title>

<ListGroup variant="flush">

{order.orderItems.map((item) => (

<ListGroup.Item key={item.\_id}>

<Row className="align-items-center">

<Col md={6}>

<img

src={item.image}

alt={item.name}

className="img-fluid rounded img-thumbnail"

></img>{" "}

<Link

style={{ textDecoration: "none" }}

to={`/product/${item.slug}`}

>

{item.name}

</Link>

</Col>

<Col md={3}>

<span>{item.quantity}</span>

</Col>

<Col md={3}>${item.price}</Col>

</Row>

</ListGroup.Item>

))}

</ListGroup>

</Card.Body>

</Card>

</Col>

</Row>

</ListGroup.Item>

</ListGroup.Item>

<p className="text-center">OR</p>

<Button variant="info bold">

<NavLink

style={{

textDecoration: "none",

color: "black",

fontWeight: "bold",

}}

exact

to="/cash"

>

Cash On Delivery

</NavLink>

</Button>

</ListGroup>

</Card.Body>

</Card>

</Col>

</Row>

</div>

);

};

export default OrderScreen;

**ProductScreen.js**

import React, { useReducer, useEffect, useContext } from "react";

import { useNavigate, useParams } from "react-router-dom";

import axios from "axios";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import ListGroup from "react-bootstrap/ListGroup";

import Badge from "react-bootstrap/Badge";

import Card from "react-bootstrap/Card";

import Rating from "../components/Rating";

import { Helmet } from "react-helmet-async";

import LoadingBox from "../components/LoadingBox";

import MessageBox from "../components/MessageBox";

import { getError } from "../utils";

import { Store } from "../Store";

const ProductScreen = () => {

return loading ? (

<LoadingBox />

) : error ? (

<MessageBox variant="danger">{error}</MessageBox>

) : (

<div>

<Row>

<Col md={6}>

<img className="img-large" src={product.image} alt={product.name} />

</Col>

<Col md={3}>

<ListGroup variant="flush">

<ListGroup.Item>

<Helmet>

<title>{product.name}</title>

</Helmet>

</ListGroup.Item>

<ListGroup.Item>

<Rating

rating={product.rating}

numReviews={product.numReviews}

></Rating>

</ListGroup.Item>

<ListGroup.Item>Price : ${product.price}</ListGroup.Item>

<ListGroup.Item>

Description:

<p>{product.description}</p>

</ListGroup.Item>

</ListGroup>

</Col>

</ListGroup.Item>

</ListGroup>

</Card.Body>

</Card>

</Col>

</Row>

</div>

);

};

export default ProductScreen;

**SearchScreen.js**

import React, { useEffect, useReducer, useState } from "react";

import { Link, useNavigate, useLocation } from "react-router-dom";

import axios from "axios";

import { toast } from "react-toastify";

import { getError } from "../utils";

import { Helmet } from "react-helmet-async";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

import Rating from "../components/Rating";

import LoadingBox from "../components/LoadingBox";

import MessageBox from "../components/MessageBox";

import Button from "react-bootstrap/Button";

import Product from "../components/Product";

import LinkContainer from "react-router-bootstrap/LinkContainer";

export default function SearchScreen() {

return (

<div>

<Row>

<Col md={3}>

<h3>Department</h3>

<div>

<ul>

<li>

<Link

className={"all" === category ? "text-bold" : ""}

to={getFilterUrl({ category: "all" })}

>

Any

</Link>

</li>

</ul>

</div>

<div>

<h3>Price</h3>

<ul>

<li>

<Link

className={"all" === price ? "text-bold" : ""}

to={getFilterUrl({ price: "all" })}

>

Any

</Link>

</li>

</ul>

</div>

<div>

<h3>Avg. Customer Review</h3>

<ul>

<li>

<Link

to={getFilterUrl({ rating: "all" })}

className={rating === "all" ? "text-bold" : ""}

>

<Rating caption={" & up"} rating={0}></Rating>

</Link>

</li>

</ul>

</div>

</Col>

<Col md={9}>

>

<option value="newest">Newest Arrivals</option>

<option value="lowest">Price: Low to High</option>

<option value="highest">Price: High to Low</option>

<option value="toprated">Avg. Customer Reviews</option>

</select>

</Col>

</Row>

<Row>

{products.map((product) => (

<Col sm={6} lg={4} className="mb-3" key={product.\_id}>

<Product product={product}></Product>

</Col>

))}

</Row>

</>

</Col>

</Row>

</div>

);

}

Export default SearchScreen;

**MessageBox.js**

import Alert from "react-bootstrap/Alert";

export default function MessageBox(props) {

return <Alert variant={props.variant || "info"}>{props.children}</Alert>;

}

**SigninSceen.js**

import React, { useContext, useEffect, useState } from "react";

import axios from "axios";

import { Link, useLocation, useNavigate } from "react-router-dom";

import Form from "react-bootstrap/Form";

import { Helmet } from "react-helmet-async";

import { Store } from "../Store";

import { toast } from "react-toastify";

import { getError } from "../utils";

const SigninScreen = () => {

return (

<div>

<div className="login">

<div className="login\_container">

<h1 className="text-center">Sign In</h1>

<form onSubmit={submitHandler}>

<Form.Group className="mb-3" controlId="email">

<h5>Email</h5>

<input

className="login\_input"

type="email"

value={email}

onChange={(e) => {

setEmail(e.target.value);

}}

/>

</Form.Group>

</Form.Group>

<div className="mb-3">

<button type="submit" className="login\_signIn">

Sign in

</button>

</div>

<p>

By signing-in you Agree to Our Condition of Use & Sale. Please See

our Privacy Notice, our Cookies and our Interest-Based Ads Notice.{" "}

</p>

<div className="login\_register\_button">

New customer?{" "}

<Link

style={{ textDecoration: "none" }}

to={`/signup?redirect=${redirect}`}

>

Create your account

</Link>

</div>

</form>

</div>

</div>

</div>

);};

export default SigninScreen;

**SignupScreen.js**

import React from "react";

import axios from "axios";

import { Link, useLocation, useNavigate } from "react-router-dom";

import Form from "react-bootstrap/Form";

import { Helmet } from "react-helmet-async";

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

import { Store } from "../Store";

const SignupScreen = () => {

return (

<div className="login">

<div className="login\_container">

<h1 className="text-center">Sign Up</h1>

<Form onSubmit={submitHandler}>

<Form.Group className="mb-3" controlId="name">

<h5>Name</h5>

<input

className="login\_input"

type="text"

value={name}

onChange={(e) => {

setName(e.target.value);

}}

/>

</Form.Group>

<Form.Group className="mb-3" controlId="email">

<h5>Email</h5>

<input

className="login\_input"

type="email"

value={email}

onChange={(e) => {

setEmail(e.target.value);

}}

/>

</Form.Group>

<Form.Group className="mb-3" controlId="password">

<h5>Password</h5>

<input

className="login\_input"

type="password"

value={password}

onChange={(e) => {

setPassword(e.target.value);

}}

/>

<div className="mb-3">

<button type="submit" className="login\_signIn">

Sign Up

</button>

</div>

</div>);}:

export default SignupScreen;

**FUTURE ENHANCEMENT**

* Admin Dashboard
* Customer Care Chat Box
* Option of Online Payment
* Payments will be customized according to Customers
* New Items Can be Added by Admin

**REFERENCE AND BIBLIOGRAPHY**

* <https://stackoverflow.com>
* <https://web.learncodeonline.in>
* <https://www.researchgate.net>
* <https://www.diagrameditor.com>
* <https://www.lucidchart.com>