PROJECT SUBMITTED

TO THE UNIVERSITY OF MUMBAI

IN PARTIAL FULFILMENT TO THE DEGREE OF BACHELOR OF SCIENCE

SMT. CHANDIBAI HIMATHMAL MANSUKHANI COLLEGE
ULHASNAGAR – 421003

TY.B.Sc.(COMPUTER SCIENCE)

ACADEMIC YEAR 2022 – 2023

OOAD PROJECTION

ONLINE EXAMINATION SYSTEM

BY

MR. VINIT DATTARAM THOMBARE

PROJECT INCHARGE

MS. HARLEEN KAUR

Rejuvenate Healthcare

Rejuvenate Healthcare

Contents

Acknowledgement

It gives me great pleasure in presenting this project report. It's justification will never sound good if I do not express my vote of thanks to our C.H.M. College and respective Principal. I would also like to thank our HOD **Mrs. Ritika Sachdev** for her timely support in this completion of this project.

I thank our internal project guide **Miss Harleen Kaur**, who has done a lot to keep this project systematically and on schedule. I am thankful for their valuable guidance to every stage of the project. Finally, I would thank to all others, who give their extended support to complete this project.

A) System Documentation

1. Synposis

The Rejuvenate Healthcare is a web-based application. The main purpose of "Rejuvenate Healthcare Website" is to provide a convenient way for a customer look for Products and treatment methods and guides online.

The objective of this project is to develop a system that automates the processes and activities of a generic medical stores. The site will provide different categorized Product brands along with its description for easy to understand the quality and design for customers. It is tedious for a customer to get the proper Product and all the knowledge about it in a typical medical store. The project 'Rejuvenate Healthcare' is developed to replace the currently existing system, which helps in keeping records of all the Product required by a consumer

2. Hardware Requirements

- **CPU:** 1,6 GHz for web, 4 x 1,6 GHz CPUs for web and database hosting.
- **RAM:** minimum 4GB.
- **Minimum database space:** 10GB

3. Software Requirements

Frontend: HTML, CSS, JavaScript, ReactJS

Backend: MongoDB, NodeJS, ExpressJS

4. Description of Present System

In the present system a customer must approach various medical stores to find details of Product or Order a particular medicinal device. This often requires loads of time and effort. And for purchasing the Product as well many of sites are available but the current system is clustered, and it becomes difficult for people to choose their product.

5. Limitations of Present System

The customer does not get the product of a particular quality. Sometimes hidden the cost of Product often do not have a person to talk to when dealing with a problem Frauds in online sites design and quality Lack of significant discount for Ordering the Product You cannot bargain Misleading pictures. You don’t always get what you’re see online. The customer does not get the product of their choice. Sometimes hidden the cost of Product often do not have a person to talk to when dealing with a problem Frauds in online sites design and quality Lack of significant discount for Ordering the Product You cannot bargain Misleading pictures. You don’t always get what you’re see online

6. Proposed System and Its Advantages

The main advantage is to manage the detail of designed Products with its description, product and Products categories and its sub-categories, and build a website that will be user friendly customer can make reservation of designed Product and buy the product by just simply sitting in their house. There will be variety of travel Products available of their choice. The orders placed by the customer will be processed in the database. The user gets all the resources at a single place instead of wandering around for these. This system provides a graphical user interface to add or delete the information. This system can be implemented from anywhere at your fingertips.

7. Timeline

	Task Name	Planned Start Date	Planned Finish Date	Planned Duration (in days)	Actual Start Date	Actual Finish Date	Signature of the Guide
1	Analysis	Saturday December 24, 2022	Thursday December 29, 2022	5	Saturday December 24, 2022	Thursday December 29, 2022	
1.1	Defining Problem Statement of the Project and gathering Information	Saturday December 24, 2022	Monday December 26, 2022	2	Saturday December 24, 2022	Monday December 26, 2022	
1.2	Defining WBS (Work Breakdown Structure) of the Project	Tuesday December 27, 2022	Wednesday December 28, 2022	2	Tuesday December 27, 2022	Wednesday December 28, 2022	
1.3	Document Current System	Thursday December 29, 2022	Thursday December 29, 2022	1	Thursday December 29, 2022	Thursday December 29, 2022	
1.4	Analysis Complete	Thursday December 29, 2022	Thursday December 29, 2022	0	Thursday December 29, 2022	Thursday December 29, 2022	
2	Design	Friday December 30, 2022	Monday January 16, 2023	14	Friday December 30, 2022	Monday January 16, 2023	
2.1	Design Database	Friday December 30, 2022	Monday January 2, 2023	3	Friday December 30, 2022	Monday January 2, 2023	
2.2	Software Design	Tuesday January 3, 2023	Saturday January 7, 2023	5	Tuesday January 3, 2023	Saturday January 7, 2023	
2.3	Interface Design	Monday January 9, 2023	Monday January 9, 2023	1	Monday January 9, 2023	Monday January 9, 2023	

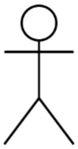
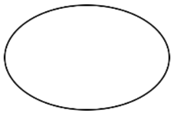
	Task Name	Planned Start Date	Planned Finish Date	Planned Duration (in days)	Actual Start Date	Actual Finish Date	Signature of the Guide
2.4	Create Design Specification	Tuesday January 10, 2023	Monday January 16, 2023	5	Tuesday January 10, 2023	Monday January 16, 2023	
2.5	Design Complete	Monday January 16, 2023	Monday January 16, 2023	0	Monday January 16, 2023	Monday January 16, 2023	
3	Development	Tuesday January 17, 2023	Thursday February 9, 2023	20	Tuesday January 17, 2023	Thursday February 9, 2023	
3.1	Develop System Module	Tuesday January 17, 2023	Saturday January 28, 2023	10	Tuesday January 17, 2023	Saturday January 28, 2023	
3.2	Integrate System Module	Monday January 30, 2023	Monday February, 2023	7	Monday January 30, 2023	Monday February, 2023	
3.3	Perform Initial Testing	Tuesday February 7, 2023	Thursday February 9, 2023	3	Tuesday February 7, 2023	Thursday February 9, 2023	
3.4	Development Complete	Thursday February 9, 2023	Thursday February 9, 2023	0	Thursday February 9, 2023	Thursday February 9, 2023	
4	Testing	Friday February 10, 2023	Wednesday March 1, 2023	16	Friday February 10, 2023	Wednesday March 1, 2023	
4.1	Perform System Testing	Friday February 10, 2023	Friday February 17, 2023	7	Friday February 10, 2023	Friday February 17, 2023	

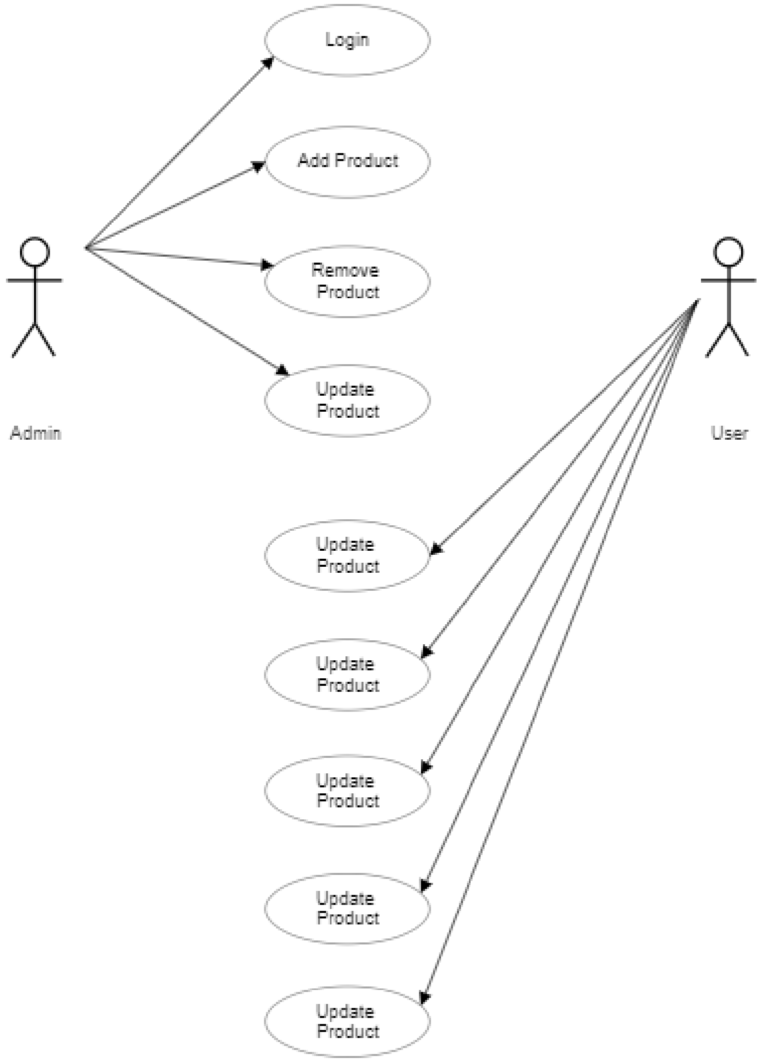
	Task Name	Planned Start Date	Planned Finish Date	Planned Duration (in days)	Actual Start Date	Actual Finish Date	Signature of the Guide
4.2	Documenting Issues Found	Monday February 20, 2023	Saturday February 25, 2023	6	Monday February 20, 2023	Saturday February 25, 2023	
4.3	Correcting/Resolving Issues Found	Monday February 27, 2023	Wednesday March 1, 2023	3	Monday February 27, 2023	Wednesday March 1, 2023	
4.4	Testing Complete	Wednesday March 1, 2023	Wednesday March 1, 2023	0	Wednesday March 1, 2023	Wednesday March 1, 2023	
5	Implementation	Thursday March 2, 2023	Monday March 6, 2023	4	Thursday March 2, 2023	Monday March 6, 2023	
5.1	On-Site/On-Campus Installation	Thursday March 2, 2023	Saturday March 4, 2023	3	Thursday March 2, 2023	Saturday March 4, 2023	
5.2	Demonstration	Monday March 6, 2023	Monday March 6, 2023	1	Monday March 6, 2023	Monday March 6, 2023	

8. UML Diagram

1. UseCase Diagram

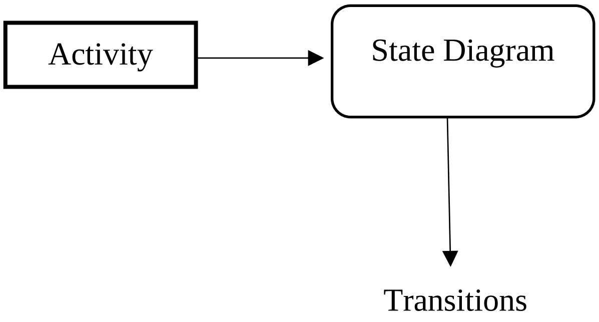
A use case diagram is a set of scenarios that describing an interaction between user and system. A use case diagram displays the relationship among actors & use cases. The 2 main components of use case diagram are use case and actor.
An actor represents a user, or another system will interact with the system that you are modelling. A use case is an external view of the system that represents some action that might perform in order to complete a task

NO	SYMBOLS	NAME	EXPLANATION
1.		**Actor**	**User which will interact with our system.**
2.		**Use case**	**It is the function to happen**
3.	________________	**Association**	**Association between use case and user**

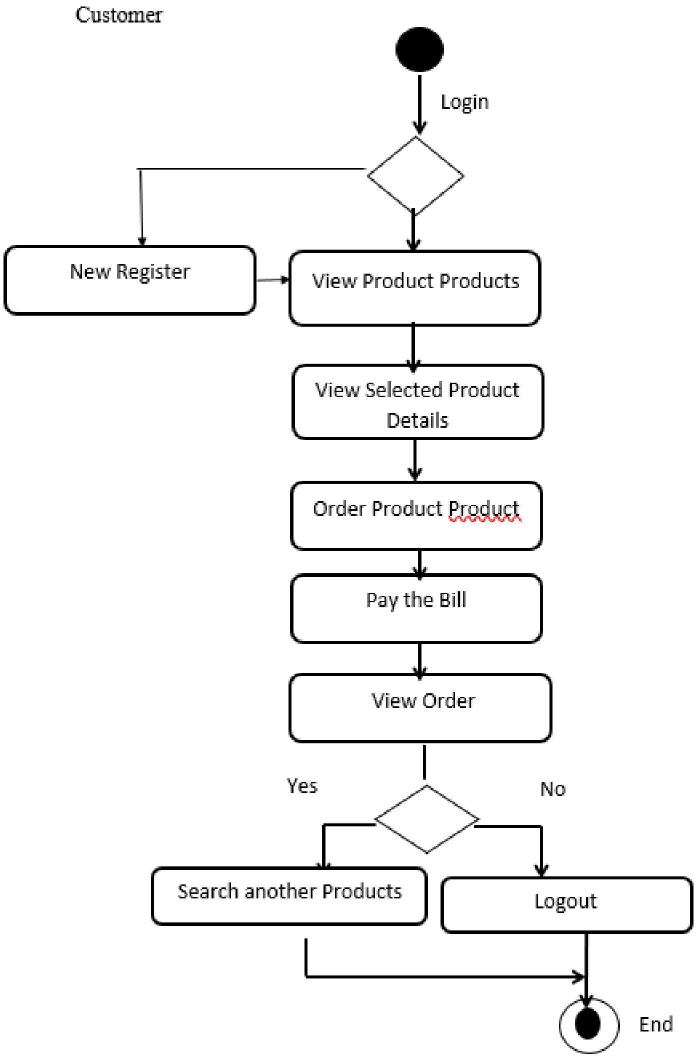

Login
Add Product
Remove Product
Update Product
Admin
User
Update Product
Update Product
Update Product
Update Product
Update Product

2. State Diagram

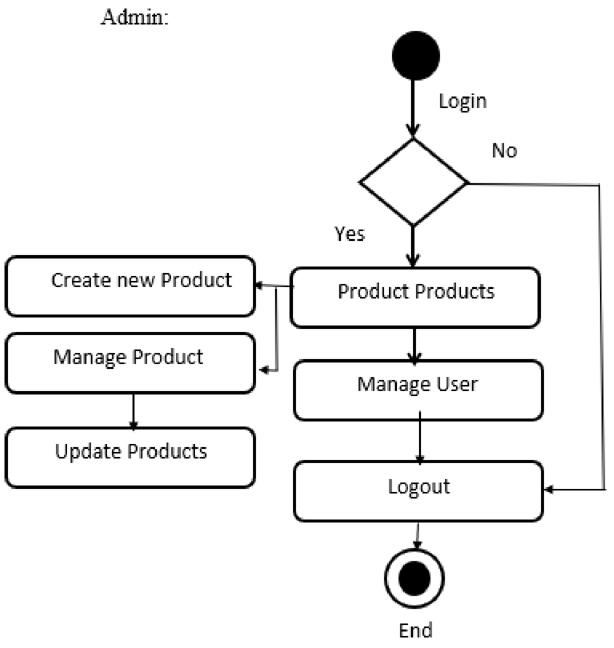
- State diagram are used to describe the behavior of the system.
- State diagram describe all of the possible state of an object as an event occurs.
- Each diagram usually represents objects of single class and track the different state of its object through the system.
- We use the state diagram to demonstrate the behavior of an object through many use cases of the system.
- We use the state diagram for classes where it is necessary to understand the behavior of the object through the entire system.



- **User:**

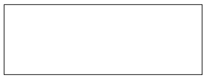
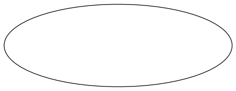
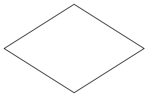
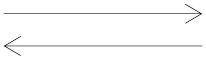
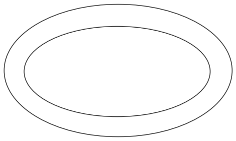
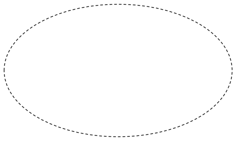
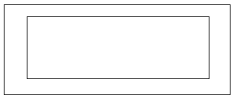
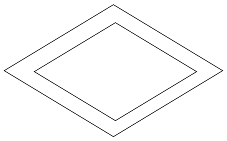

Customer
Login
New Register
View Product Products
View Selected Product Details
Order Product Product
Pay the Bill
View Order
Yes
No
Search another Products
Logout
End

- ★ **Admin**

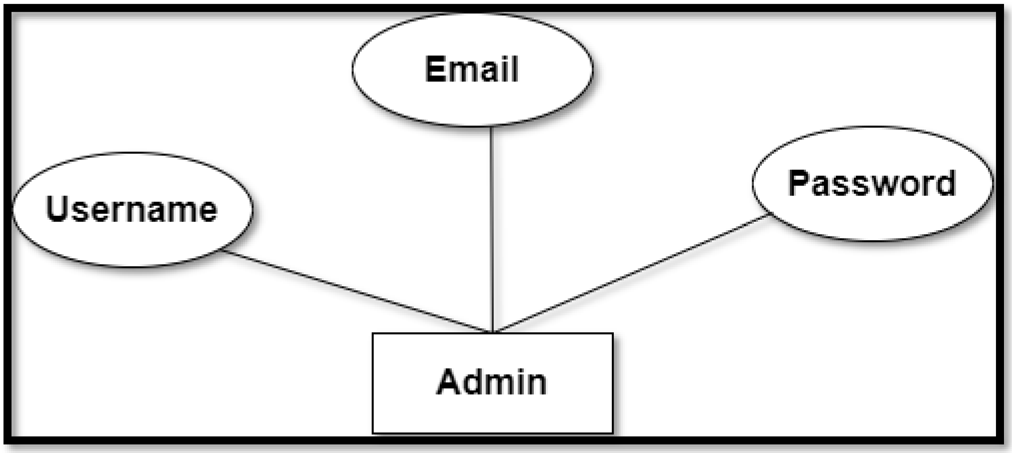

Admin:
Login
No
Yes
Create new Product
Product Products
Manage Product
Manage User
Update Products
Logout
End

3. ER Diagram

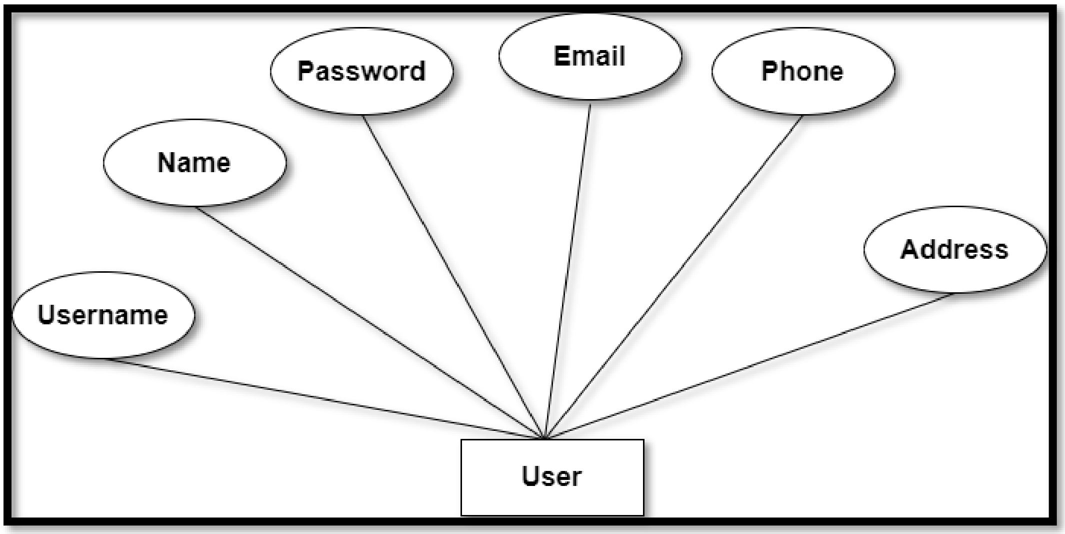
- Entity relationship diagram can express overall logical structure of database logically.
- ER Diagrams are simple and clear.
- ER Diagrams represents entities and tables and their relationship with one another.

Sr. No.		Shape	Description
1	Rectangle		Represents entity set.
2	Ellipse		Represents attributes.
3	Diamond		Represents relationship.
4	Flow lines		Represents link between 2 entities set.
5	Double ellipse		Represents multivalve attributes.
6	Dashed ellipse		Denotes derived attributes.
7	Double Rectangle		Represents weak entity set.
8	Double Diamond		Represents relationship set for weak entity set.

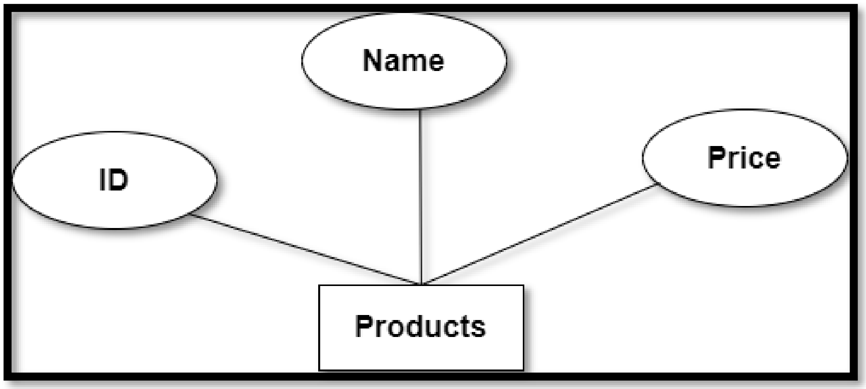
- **ER Diagram For Admin**



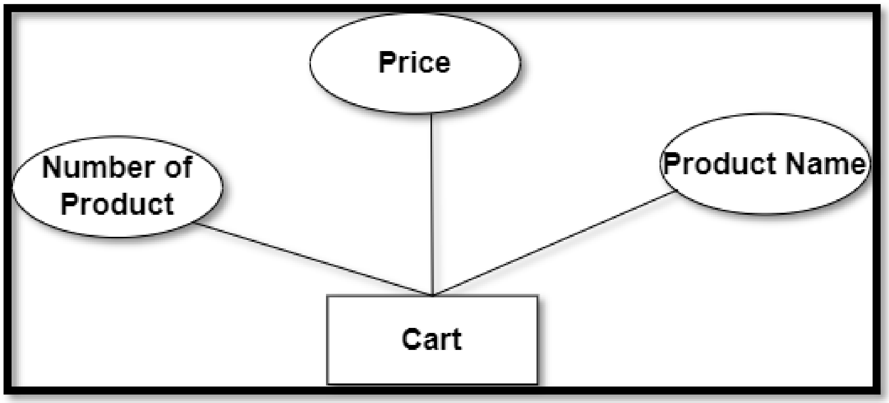
- **ER Diagram For User**



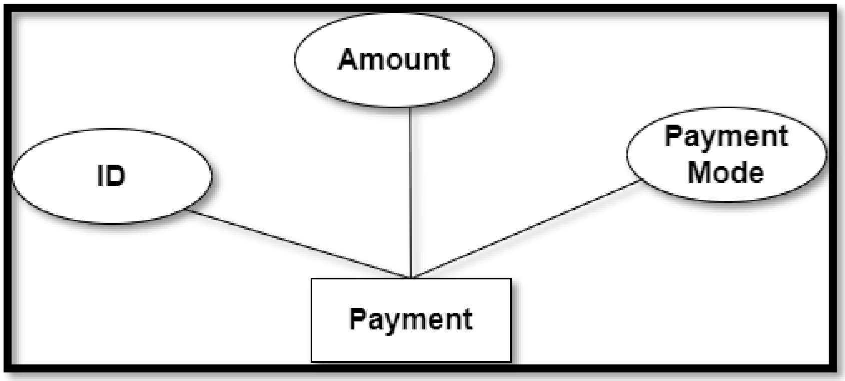
- **Products ER Diagram:**



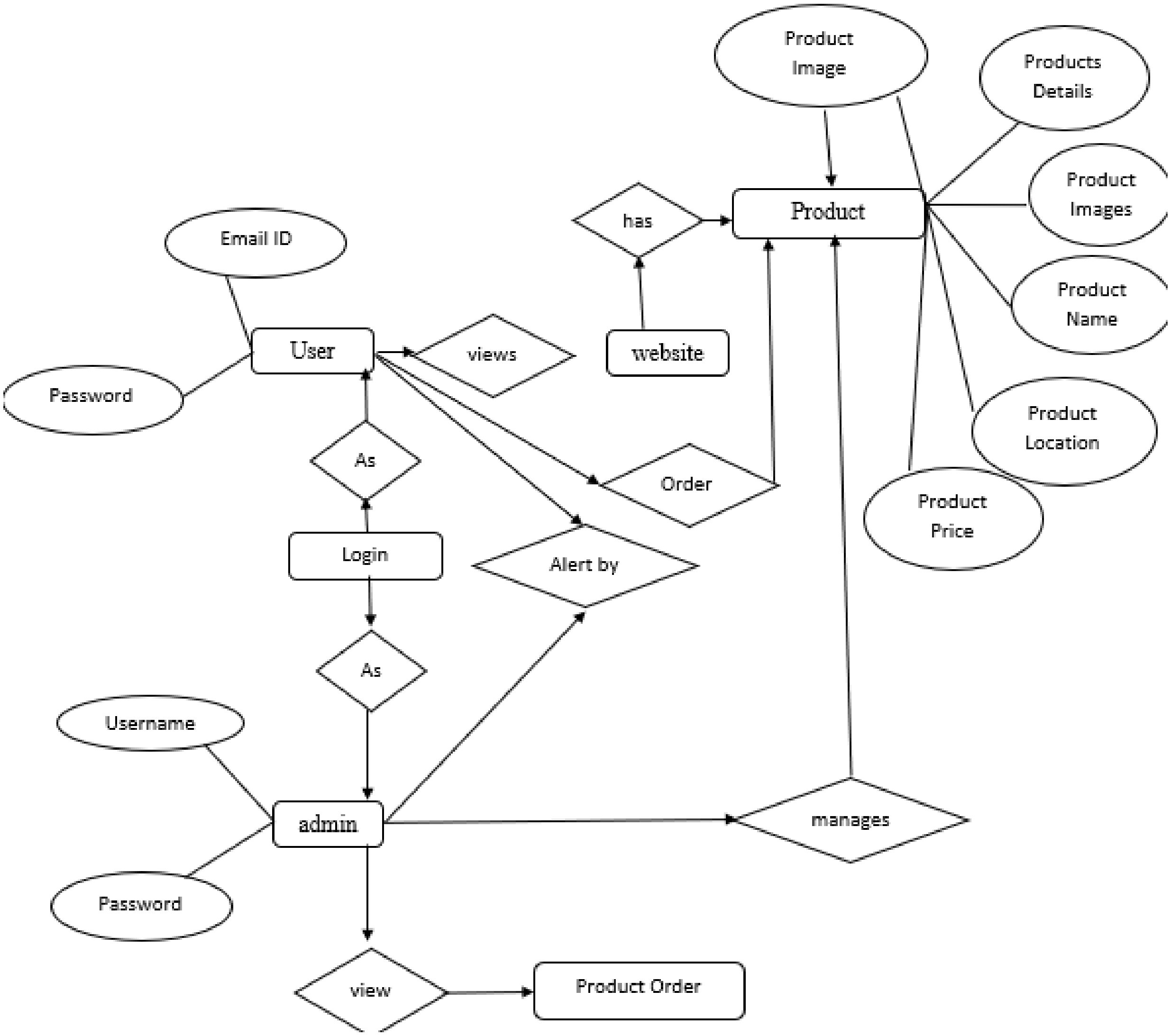
- **Cart ER Diagram:**



- **Payment ER Diagram:**



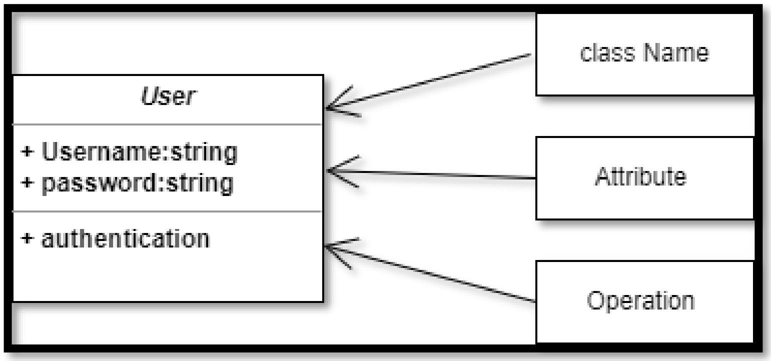
ER Diagram

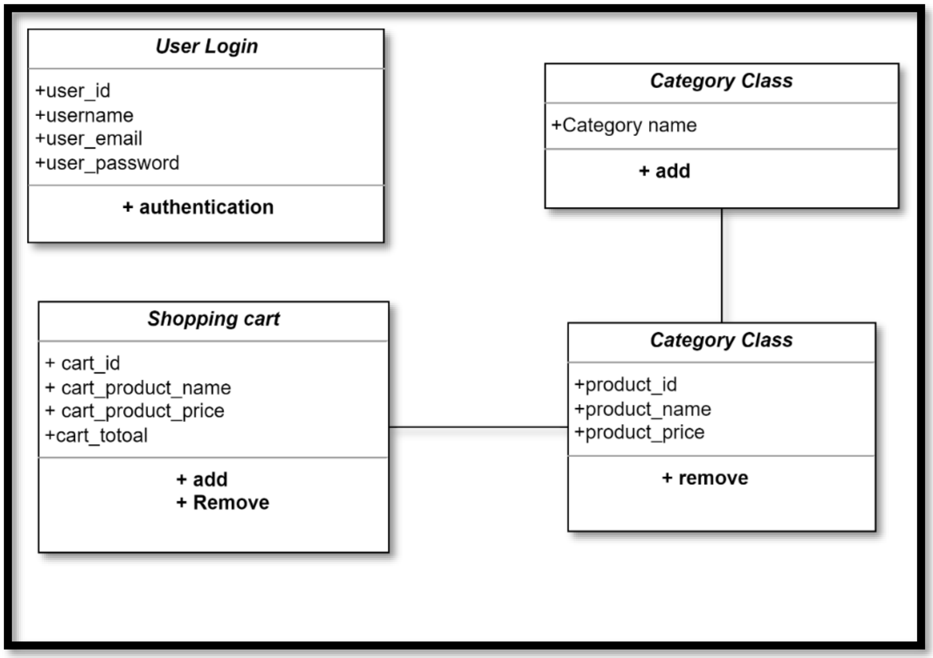

Product Image
Products Details
Product
has
Product Images
Email ID
Product Name
User
views
website
Password
Product Location
As
Order
Product Price
Login
Alert by
As
Username
admin
manages
Password
view
Product Order

4. Class Diagram

- Class diagrams are widely used to describe the types of objects used in system and their relationship. Class diagram models class structure and contents using design elements such as classes and packages and objects.
- Class diagram describes 3 different perspectives when designing a system. These perspectives become evident as the diagram is created and help solidify the design.
- Classes are composed of 3 things:
  - i. **Class name.**
  - ii. **Attributes and**
  - iii. **Operations.**

For Example:

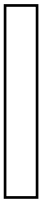
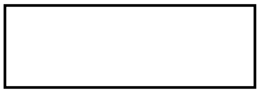
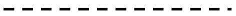
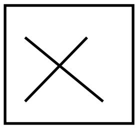


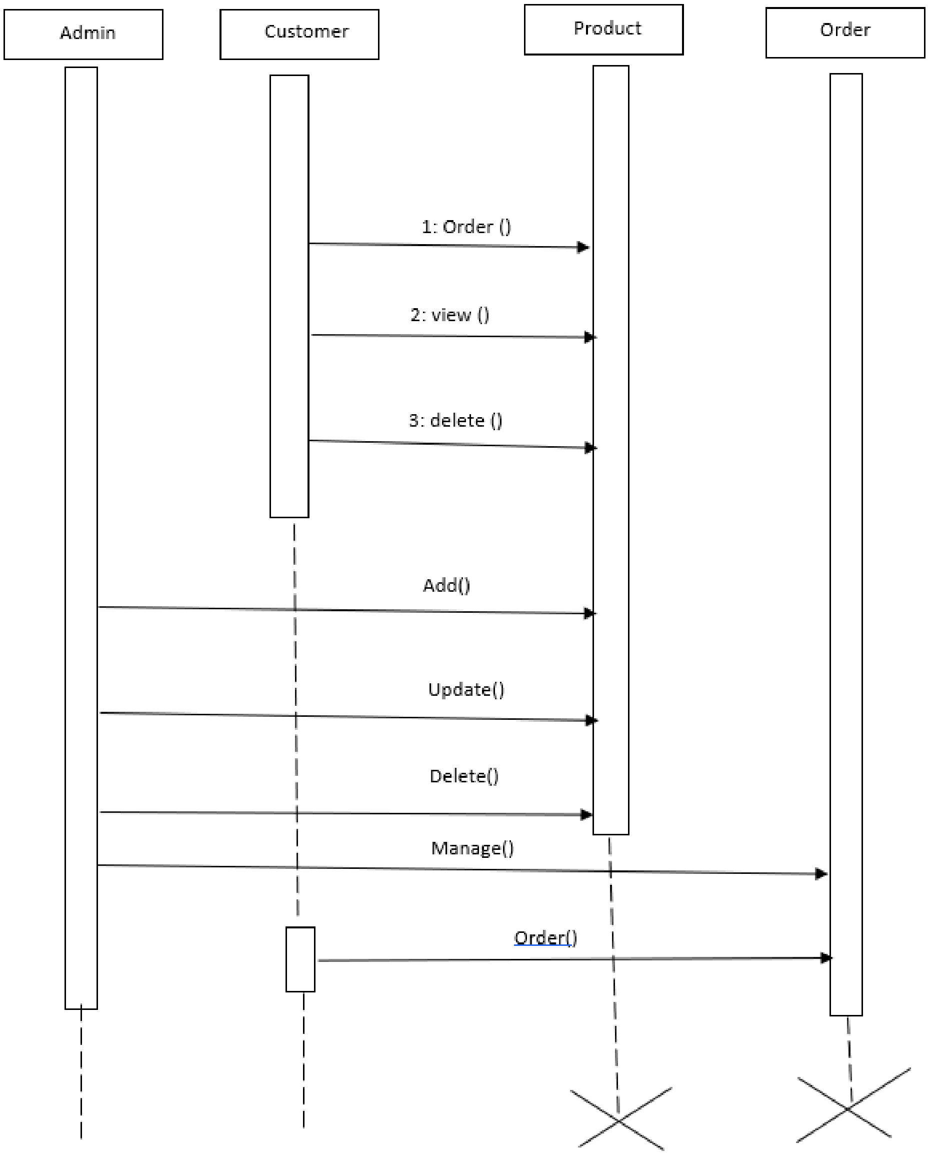


5. Sequence Diagram

- Sequence diagram demonstrates the behavior of the objects in a use- case by describing the objects and the messages they pass.
- The diagrams are read left to right & descending.

Components of Sequence Diagram:-

1		Represent object activation.
2		Represents the objects of case study.
3		Represents life of objects.
4		Represents end of objects.


Admin
Customer
Product
Order
1: Order ()
2: view ()
3: delete ()
Add()
Update()
Delete()
Manage()
Order()

9. Files Listing

- **Frontend**

1. Login.js
2. Register.js
3. ForgotPassword.js
4. About.js
5. Contact.js
6. Homepage.js
7. Cartpage.js
8. Categories.js
9. CategoryProduct.js
10. ProductDetails.js
11. Search.js
12. App.js
13. index.js
14. index.css

- Admin

1. AdminDashboard.js
2. AdminOrders.js
3. CreateCategory.js
4. CreateProduct.js
5. Product.js
6. updateProduct.js

- User

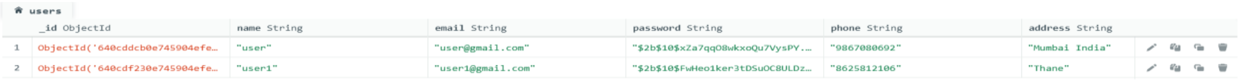
1. Users.js
2. UserDashboard.js
3. Orders.js
4. Profile.js

- **Backend**

1) .env
2) db.js
3) server.js
4) controllers :
    - ▪ authcontroller.js
    - ▪ categoryController.js
    - ▪ productController.js
5) models
    - categoryModel.js
    - orderModel.js
    - productModel.js
    - userModel.js
6) routes
    - ○ authRoute.js
    - ○ categoryRoutes.js
    - ○ productRoutes.js

10.Database

- User Table

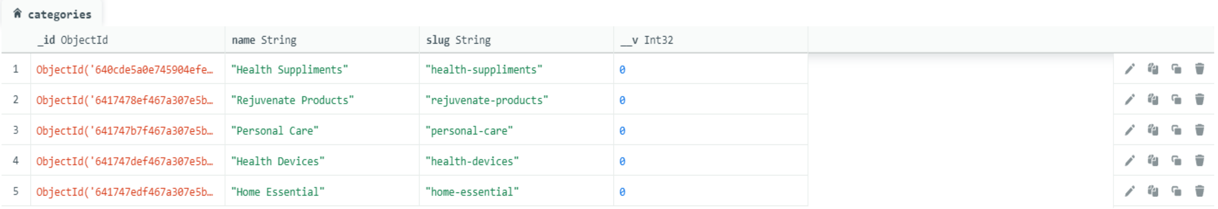


users

	_id ObjectId	name String	email String	password String	phone String	address String
1	ObjectId('640cddcb0e745904efe…	"user"	"user@gmail.com"	"$2b$10$xZa7qqO8wkxoQu7VysPY.…	"9867080692"	"Mumbai India"
2	ObjectId('640cdf230e745904efe…	"user1"	"user1@gmail.com"	"$2b$10$FwHeo1ker3tDSuOC8ULDz…	"8625812106"	"Thane"

Field Name	Datatypes	Description
id	int	Id of user
name	string	It is a string containing name
email	varchar	Email-id of user
password	varchar	Password of the user
phone	int	Contact details of user
address	varchar	Address of the user
answer	object	Security question of the user
role	object	Role of the user
createdAt	object	Date of user creation
updatedAt	object	User Updation date

- Categories

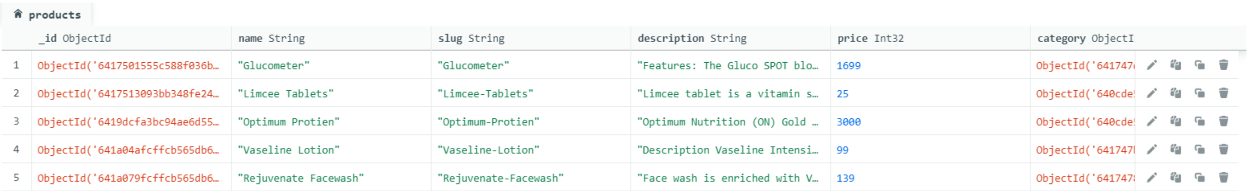


categories

	_id ObjectId	name String	slug String	__v Int32
1	ObjectId('640cde5a0e745904efe…	"Health Suppliments"	"health-suppliments"	0
2	ObjectId('6417478ef467a307e5b…	"Rejuvenate Products"	"rejuvenate-products"	0
3	ObjectId('641747b7f467a307e5b…	"Personal Care"	"personal-care"	0
4	ObjectId('641747def467a307e5b…	"Health Devices"	"health-devices"	0
5	ObjectId('641747edf467a307e5b…	"Home Essential"	"home-essential"	0

FieldName	DataTypes	Description
name	STRING	Category of product
SLUG	STRING	Name of user
_V	Int	Contact of user

- Products



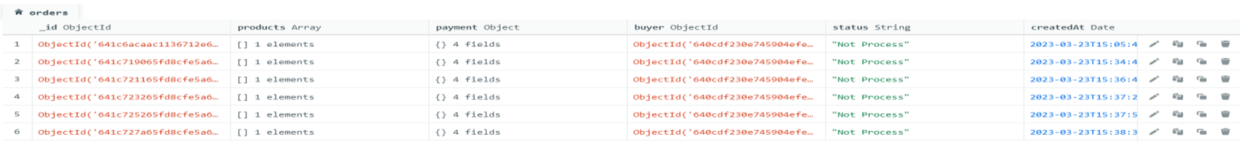
products

	_id ObjectId	name String	slug String	description String	price Int32	category ObjectI
1	ObjectId('6417501555c588f036b…	"Glucometer"	"Glucometer"	"Features: The Gluco SPOT blo…	1699	ObjectId('641747
2	ObjectId('6417513093bb348fe24…	"Limcee Tablets"	"Limcee-Tablets"	"Limcee tablet is a vitamin s…	25	ObjectId('640cde
3	ObjectId('6419dcfa3bc94ae6d55…	"Optimum Protien"	"Optimum-Protien"	"Optimum Nutrition (ON) Gold …	3000	ObjectId('640cde
4	ObjectId('641a04afcffcb565db6…	"Vaseline Lotion"	"Vaseline-Lotion"	"Description Vaseline Intensi…	99	ObjectId('641747
5	ObjectId('641a079fcffcb565db6…	"Rejuvenate Facewash"	"Rejuvenate-Facewash"	"Face wash is enriched with V…	139	ObjectId('641747

Field Name	Datatypes	Description
id	int	Unique id of Product
name	string	Name of Product
description	string	Description of the product
price	int	Price of Product
category	object	Category of the product
quantity	object	Object of the product

photo	object	Image of the product
createdAt	object	Product creation date
updateAt	object	Product updated at

- <u>Order Table</u>



orders

	_id ObjectId	products Array	payment Object	buyer ObjectId	status String	createdAt Date
1	ObjectId('641c6acaac1136712e6…	[] 1 elements	{} 4 fields	ObjectId('640cdf230e745904efe…	"Not Process"	2023-03-23T15:05:4
2	ObjectId('641c719065fd8cfe5a6…	[] 1 elements	{} 4 fields	ObjectId('640cdf230e745904efe…	"Not Process"	2023-03-23T15:34:4
3	ObjectId('641c721165fd8cfe5a6…	[] 1 elements	{} 4 fields	ObjectId('640cdf230e745904efe…	"Not Process"	2023-03-23T15:36:4
4	ObjectId('641c723265fd8cfe5a6…	[] 1 elements	{} 4 fields	ObjectId('640cdf230e745904efe…	"Not Process"	2023-03-23T15:37:2
5	ObjectId('641c725265fd8cfe5a6…	[] 1 elements	{} 4 fields	ObjectId('640cdf230e745904efe…	"Not Process"	2023-03-23T15:37:5
6	ObjectId('641c727a65fd8cfe5a6…	[] 1 elements	{} 4 fields	ObjectId('640cdf230e745904efe…	"Not Process"	2023-03-23T15:38:3

Field Name	**Datatypes**	**Description**
ID	Int	Product
products	Array	product
payment	object	Payment of the product
buyer	object	Name of User
status	string	Detail about product status
createdAt	object	Ordered at date
updatedAt	object	Order updated at

B) Project Documentation

1. Source Code

- **Frontend**

1. Login.js

```
import React, { useState } from "react";
import Layout from "./../../components/Layout/Layout";
import axios from "axios";
import { useNavigate, useLocation } from "react-router-dom";
import toast from "react-hot-toast";
import "../../styles/AuthStyles.css";
import { useAuth } from "../../context/auth";
const Login = () => {
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [auth, setAuth] = useAuth();

  const navigate = useNavigate();
  const location = useLocation();

  // form function
  const handleSubmit = async (e) => {
    e.preventDefault();
    try {
      const res = await axios.post("/api/v1/auth/login", {
        email,
        password,
      });
      if (res && res.data.success) {
        toast.success(res.data && res.data.message);
        setAuth({
          ...auth,
          user: res.data.user,
          token: res.data.token,
        });
        localStorage.setItem("auth", JSON.stringify(res.data));
        navigate(location.state || "/");
      } else {
        toast.error(res.data.message);
      }
    } catch (error) {
      console.log(error);
      toast.error("Something went wrong");
    }
  };
  return (
    <Layout title="Register - Ecommer App">
      <div className="form-container " style={{ minHeight: "90vh" }}>
        <form onSubmit={handleSubmit}>
          <h4 className="title">LOGIN FORM</h4>

          <div className="mb-3">
            <input
              type="email"
              autoFocus
              value={email}
              onChange={(e) => setEmail(e.target.value)}
              className="form-control"
              id="exampleInputEmail1"
              placeholder="Enter Your Email "
              required
            />
          </div>
          <div className="mb-3">
            <input
              type="password"
              value={password}
              onChange={(e) => setPassword(e.target.value)}
              className="form-control"
              id="exampleInputPassword1"
              placeholder="Enter Your Password"
              required
            />
          </div>
          <div className="mb-3">
            <button
              type="button"
              className="btn forgot-btn"
              onClick={() => {
                navigate("/forgot-password");
```

```
          }}
         >
          Forgot Password
         </button>
        </div>

        <button type="submit" className="btn btn-primary">
         LOGIN
        </button>
       </form>
      </div>
    </Layout>
  );
};

export default Login;
```

2. Register.js

```
import React, { useState } from "react";
import Layout from "./../../components/Layout/Layout";
import axios from "axios";
import { useNavigate } from "react-router-dom";
import toast from "react-hot-toast";
import "../../styles/AuthStyles.css";
const Register = () => {
  const [name, setName] = useState("");
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [phone, setPhone] = useState("");
  const [address, setAddress] = useState("");
  const [answer, setAnswer] = useState("");
  const navigate = useNavigate();

  // form function
  const handleSubmit = async (e) => {
    e.preventDefault();
    try {
      const res = await axios.post("/api/v1/auth/register", {
        name,
        email,
        password,
        phone,
        address,
        answer,
      });
      if (res && res.data.success) {
        toast.success(res.data && res.data.message);
        navigate("/login");
      } else {
        toast.error(res.data.message);
      }
    } catch (error) {
      console.log(error);
      toast.error("Something went wrong");
    }
  };

  return (
    <Layout title="Register - Ecommer App">
      <div
        className="form-container"
        style={{ minHeight: "90vh", backgroundColor: "red" }}
      >
        <form onSubmit={handleSubmit}>
          <h4 className="title">REGISTER FORM</h4>
          <div className="mb-3">
            <input
              type="text"
              value={name}
              onChange={(e) => setName(e.target.value)}
              className="form-control"
              id="exampleInputEmail1"
              placeholder="Enter Your Name"
              required
              autoFocus
            />
          </div>
          <div className="mb-3">
            <input
              type="email"
              value={email}
              onChange={(e) => setEmail(e.target.value)}
              className="form-control"
              id="exampleInputEmail1"
              placeholder="Enter Your Email "
              required
            />
          </div>
          <div className="mb-3">
            <input
              type="password"
              value={password}
              onChange={(e) => setPassword(e.target.value)}
              className="form-control"
              id="exampleInputPassword1"
              placeholder="Enter Your Password"
              required
            />
          </div>
          <div className="mb-3">
            <input
              type="text"
              value={phone}
              onChange={(e) => setPhone(e.target.value)}
              className="form-control"
              id="exampleInputEmail1"
              placeholder="Enter Your Phone"
              required
            />
          </div>
          <div className="mb-3">
            <input
              type="text"
              value={address}
              onChange={(e) => setAddress(e.target.value)}
              className="form-control"
              id="exampleInputEmail1"
              placeholder="Enter Your Address"
              required
            />
          </div>
```

```
          <div className="mb-3">
            <input
              type="text"
              value={answer}
              onChange={(e) => setAnswer(e.target.value)}
              className="form-control"
              id="exampleInputEmail1"
              placeholder="What is Your Favorite sports"
              required
            />
          </div>
          <button type="submit" className="btn btn-primary">
            REGISTER
          </button>
        </form>
      </div>
    </Layout>
  );
};

export default Register;
```

3. **ForgotPassword.js**

```
import React, { useState } from "react";
import Layout from "./../../components/Layout/Layout";
import axios from "axios";
import { useNavigate } from "react-router-dom";
import toast from "react-hot-toast";
import "../../styles/AuthStyles.css";

const ForgotPasssword = () => {
  const [email, setEmail] = useState("");
  const [newPassword, setNewPassword] = useState("");
  const [answer, setAnswer] = useState("");

  const navigate = useNavigate();

  // form function
  const handleSubmit = async (e) => {
    e.preventDefault();
    try {
      const res = await axios.post("/api/v1/auth/forgot-password", {
        email,
        newPassword,
        answer,
      });
      if (res && res.data.success) {
        toast.success(res.data && res.data.message);

        navigate("/login");
      } else {
        toast.error(res.data.message);
      }
    } catch (error) {
      console.log(error);
      toast.error("Something went wrong");
    }
  };
  return (
    <Layout title={"Forgot Password - Ecommerce APP"}>
      <div className="form-container ">
        <form onSubmit={handleSubmit}>
          <h4 className="title">RESET PASSWORD</h4>

          <div className="mb-3">
            <input
              type="email"
              value={email}
              onChange={(e) => setEmail(e.target.value)}
              className="form-control"
              id="exampleInputEmail1"
              placeholder="Enter Your Email "
              required
            />
          </div>
          <div className="mb-3">
            <input
              type="text"
              value={answer}
```

```
                onChange={(e) => setAnswer(e.target.value)}
                className="form-control"
                id="exampleInputEmail1"
                placeholder="Enter Your favorite Sport Name "
                required
              />
            </div>
            <div className="mb-3">
              <input
                type="password"
                value={newPassword}
                onChange={(e) => setNewPassword(e.target.value)}
                className="form-control"
                id="exampleInputPassword1"
                placeholder="Enter Your Password"
                required
              />
            </div>

            <button type="submit" className="btn btn-primary">
              RESET
            </button>
          </form>
        </div>
      </Layout>
    );
```

```
};


export default ForgotPassssword;;
```

4. About.js

```
import React from "react";
import Layout from "./../components/Layout/Layout";

const About = () => {
  return (
    <Layout title={"About us - Ecommer app"}>
      <div className="row contactus ">
        <div className="col-md-6 ">
          <img
            src="/images/about.jpeg"
            alt="contactus"
            style={{ width: "100%" }}
          />
        </div>
        <div className="col-md-4">
          <p className="text-justify mt-2">
            Rejuvenate Healthcare is a web based app made using mern stack.
          </p>
        </div>
      </div>
    </Layout>
  );
};


export default About;;
```

5. Policy.js

```
import React from "react";

import Layout from "./../components/Layout/Layout";

const Policy = () => {

  return (

    <Layout title={"Privacy Policy- 🪴 PlantParadise"}>

      <div

        className="row contactus "

        style={{minHeight: "90vh",paddingRight: "2%",marginTop: "5px",

boxShadow: "1px 2px 9px #F4AAB9",margin: "10 auto",padding: "1em",

backgroundImage: "url(/images/Picture2.png)",background: "cover",

backgroundRepeat: "no-repeat",backgroundSize: "cover",minHeight: "80vh",height: "100px",}}>

        <div

          className="col-md-6 "

          style={{paddingRight: "2%",marginTop: "5px",boxShadow: "1px 2px 9px
#F4AAB9",margin: "10 auto",padding: "1em",backgroundImage:
"url(/images/Picture2.png)",backgroundSize: "cover",backgroundRepeat: "no-repeat",  }}  >

          <h1 className="bg-dark p-2 text-white text-center">Privacy Policy</h1>

          <div className="col-md-4" style={{ textAlign: "center" }}>

            <p> privacy policy</p>

            <p> privacy policy</p>

            <p> privacy policy</p>

            <p> privacy policy</p>

            <p> privacy policy</p>
            <p> privacy policy</p>
            <p> privacy policy</p>
          </div></div></div></Layout>);};
export default Policy;
```

6. Contact.js

```
import React from "react";
import Layout from "./../components/Layout/Layout";
import { BiMailSend, BiPhoneCall, BiSupport } from "react-icons/bi";
const Contact = () => {
  return (
    <Layout title={"Contact us"}>
      <div className="row contactus ">
        <div className="col-md-6 ">
          <img
            src="/images/contactus.jpeg"
            alt="contactus"
            style={{ width: "75%", borderRadius: "25px" }}
          />
        </div>
        <div className="col-md-4">
          <h1 className="bg-dark p-2 text-white text-center">CONTACT US</h1>
          <p className="text-justify mt-2">Contact us for any help</p>
          <p className="mt-3">
            <BiMailSend /> : rejcarehelp@gmail.com
          </p>
          <p className="mt-3">
            <BiPhoneCall /> : 8625812106
          </p>
          <p className="mt-3">
            <BiSupport /> : 1800-0000-0000 (toll free)
          </p>
        </div>
      </div>
    </Layout>
  );
};

export default Contact;
```

7. Homepage.js

```
import React, { useState, useEffect } from "react";
import { useNavigate } from "react-router-dom";
import { Checkbox, Radio } from "antd";
import { Prices } from "../components/Prices";
import { useCart } from "../context/cart";
import axios from "axios";
import toast from "react-hot-toast";
import Layout from "./../components/Layout/Layout";
import { AiOutlineReload } from "react-icons/ai";
import "../styles/Homepage.css";
import ImageSlider from "./ImageSlider";

const slides = [
  { url: "/images/slider1.jpg" },
  { url: "/images/sliderhs.jpg" },
  { url: "/images/slider3.2.jpg" },
  { url: "/images/hdslider.jpg" },
  { url: "/images/slider4.4.jpg" },
];
const containerStyles = {
  width: "1500px",
  height: "300px",
  margin: "auto",
  marginBottom: "2rem",
};
```

```
const HomePage = () => {
  const navigate = useNavigate();
  const [cart, setCart] = useCart();
  const [products, setProducts] = useState([]);
  const [categories, setCategories] = useState([]);
  const [checked, setChecked] = useState([]);
  const [radio, setRadio] = useState([]);
  const [total, setTotal] = useState(0);
  const [page, setPage] = useState(1);
  const [loading, setLoading] = useState(false);

  //get all cat
  const getAllCategory = async () => {
    try {
      const { data } = await axios.get("/api/v1/category/get-category");
      if (data?.success) {
        setCategories(data?.category);
      }
    } catch (error) {
      console.log(error);
    }
  };

  useEffect(() => {
    getAllCategory();
  getTotal();
 }, []);
 //get products
 const getAllProducts = async () => {
  try {
   setLoading(true);
   const { data } = await axios.get(`/api/v1/product/product-list/${page}`);
   setLoading(false);
   setProducts(data.products);
  } catch (error) {
   setLoading(false);
   console.log(error);
  }
 };

 //getTOtal COunt
 const getTotal = async () => {
  try {
   const { data } = await axios.get("/api/v1/product/product-count");
   setTotal(data?.total);
  } catch (error) {
   console.log(error);
  }
 };

 useEffect(() => {
   if (page === 1) return;
   loadMore();
 }, [page]);
 //load more
 const loadMore = async () => {
   try {
     setLoading(true);
     const { data } = await axios.get(`/api/v1/product/product-list/${page}`);
     setLoading(false);
     setProducts([...products, ...data?.products]);
   } catch (error) {
     console.log(error);
     setLoading(false);
   }
 };

 // filter by cat
 const handleFilter = (value, id) => {
   let all = [...checked];
   if (value) {
     all.push(id);
   } else {
     all = all.filter((c) => c !== id);
   }
   setChecked(all);
 };
```

```
useEffect(() => {
  if (!checked.length || !radio.length) getAllProducts();
}, [checked.length, radio.length]);

useEffect(() => {
  if (checked.length || radio.length) filterProduct();
}, [checked, radio]);

//get filterd product
const filterProduct = async () => {
  try {
    const { data } = await axios.post("/api/v1/product/product-filters", {
      checked,
      radio,
    });
    setProducts(data?.products);
  } catch (error) {
    console.log(error);
  }
};
return (
  <Layout title={"ALL Products - Best offers "}>
    <div>
      <h1>Hello monsterlessons</h1>
      <div style={containerStyles}>
        <ImageSlider slides={slides} />
      </div>
    </div>
    {/* banner image */}
    {/* <img
      src="/images/banner.png"
      className="banner-img"
      alt="bannerimage"
      width={"100%"}
    /> */}
    {/* banner image */}
    <div
      className="container-fluid row mt-3 home-page"
      style={{
        backgroundColor: "#F0FFFF",
        borderRadius: "35px",
        marginLeft: "0.1rem",
        marginRight: "1rem",
        padding: "30px",
      }}
    >
      <div
        className="col-md-3 filters"
        // style={{
        //   backgroundColor: "#98AFC7",
        //   borderRadius: "25px",
        //   marginBottom: "5rem",
        //   borderBlockStyle: "solid",
        //   position: "static",
        //   border: "3px",
        // }}
      >
        <h4 className="text-center">Filter By Category</h4>
        <div className="d-flex flex-column">
          {categories?.map((c) => (
            <Checkbox
              key={c._id}
              onChange={(e) => handleFilter(e.target.checked, c._id)}
            >
              {c.name}
            </Checkbox>
          ))}
        </div>
        {/* price filter */}
        <h4 className="text-center mt-4">Filter By Price</h4>
        <div className="d-flex flex-column">
          <Radio.Group onChange={(e) => setRadio(e.target.value)}>
            {Prices?.map((p) => (
              <div key={p._id}>
                <Radio value={p.array}>{p.name}</Radio>
              </div>
            ))}
          </Radio.Group>
        </div>
        <div className="d-flex flex-column">
          <button
            className="btn btn-danger"
            onClick={() => window.location.reload()}
          >
            RESET FILTERS
          </button>
        </div>
      </div>
      <div className="col-md-9 ">
        <h1 className="text-center">All Products</h1>
        <div
          className="d-flex flex-wrap mx-5"
          style={{
            backgroundColor: "#CFECEC",
            borderRadius: "25px",
            marginRight: "5rem",
            borderBlockStyle: "solid",
            borderBlockWidth: "7px",
            borderBlockColor: "#307D7E",
          }}
        >
          {products?.map((p) => (
            <div className="card m-2" key={p._id}>
              <img
        src={`/api/v1/product/product-photo/${p._id}`}
        className="card-img-top"
        alt={p.name}
      />
      <div className="card-body">
        <div className="card-name-price">
          <h5 className="card-title">{p.name}</h5>
          <h5 className="card-title card-price">
            {p.price.toLocaleString("en-US", {
              style: "currency",
              currency: "INR",
            })}
          </h5>
        </div>
        <p className="card-text ">
          {p.description.substring(0, 60)}...
        </p>
        <div className="card-name-price">
          <button
            className="btn btn-info ms-1"
            onClick={() => navigate(`/product/${p.slug}`)}
          >
            More Details
          </button>
          <button
            className="btn btn-dark ms-1"
```

```
            onClick={() => {
              setCart([...cart, p]);
              localStorage.setItem(
                "cart",
                JSON.stringify([...cart, p])
              );
              toast.success("Item Added to cart");
            }}
          >
            ADD TO CART
          </button>
        </div>
      </div>
    </div>
  ))}
</div>
<div className="m-2 p-3">
  {products && products.length < total && (
    <button
      className="btn loadmore"
      onClick={(e) => {
        e.preventDefault();
        setPage(page + 1);
      }}
    >
      {loading ? (
                  "Loading ..."
                ) : (
                  <>
                    {" "}
                    Loadmore <AiOutlineReload />
                  </>
                )}
              </button>
            )}
          </div>
        </div>
      </div>
    </Layout>
  );
};

export default HomePage;;
```

8. Cartpage.js

```
import React, { useState, useEffect } from "react";
import Layout from "./../components/Layout/Layout";
import { useCart } from "../context/cart";
import { useAuth } from "../context/auth";
import { useNavigate } from "react-router-dom";
import DropIn from "braintree-web-drop-in-react";
import { AiFillWarning } from "react-icons/ai";
import axios from "axios";
import toast from "react-hot-toast";
import "../styles/CartStyles.css";

const CartPage = () => {
  const [auth, setAuth] = useAuth();
  const [cart, setCart] = useCart();
  const [clientToken, setClientToken] = useState("");
  const [instance, setInstance] = useState("");
  const [loading, setLoading] = useState(false);
  const navigate = useNavigate();

  const cost = () => {
    try {
      let total = 0;
      cart?.map((item) => {
        total = total + item.price;
      });
      return total;
    } catch (error) {
      console.log(error);
    }
  };

  //total price
  const totalPrice = () => {
    try {
```

```
    let total = 0;
    cart?.map((item) => {
      total = total + item.price;
    });
    return total.toLocaleString("en-US", {
      style: "currency",
      currency: "INR",
    });
  } catch (error) {
    console.log(error);
  }
};
//detele item
const removeCartItem = (pid) => {
  try {
    let myCart = [...cart];
    let index = myCart.findIndex((item) => item._id === pid);
    myCart.splice(index, 1);
    setCart(myCart);
    localStorage.setItem("cart", JSON.stringify(myCart));
  } catch (error) {
    console.log(error);
  }
};

//get payment gateway token
```

```
const getToken = async () => {
  try {
    const { data } = await axios.get("/api/v1/product/braintree/token");
    setClientToken(data?.clientToken);
  } catch (error) {
    console.log(error);
  }
};
useEffect(() => {
  getToken();
}, [auth?.token]);

//handle payments
const handlePayment = async () => {
  try {
    setLoading(true);
    const { nonce } = await instance.requestPaymentMethod();
    const { data } = await axios.post("/api/v1/product/braintree/payment", {
      nonce,
      cart,
    });
    setLoading(false);
    localStorage.removeItem("cart");
    setCart([]);
    navigate("/dashboard/user/orders");
    toast.success("Payment Completed Successfully ");
  } catch (error) {
    console.log(error);
    setLoading(false);
  }
};

const loadScript = (src) => {
  return new Promise((resolve) => {
    const script = document.createElement("script");
    script.src = src;

    script.onload = () => {
      resolve(true);
    };

    script.onerror = () => {
      resolve(false);
    };

    document.body.appendChild(script);
  });
};

const razorPay = async () => {
  const amount = cost();
  const res = await loadScript(
    "https://checkout.razorpay.com/v1/checkout.js"
  );

  if (!res) {
    alert("You are offline");
    return;
  }

  const options = {
    key: "rzp_test_0IYFXNw795iFmU",
    currency: "INR",
    amount: amount * 100,
    name: "Code with the himmy",
    description: "Thanks for purchasing",
    image:
      "https://cdn.pixabay.com/photo/2015/04/23/22/00/tree-736885__480.jpg",

    handler: function (response) {
      // alert(response.raporpay_payment_id);
      // alert("Payment Successful");
      // alert("Your Order has been placed");
    },
    prefill: {
      name: "code with himmy",
    },
  };
```

```
  const paymentObject = new window.Razorpay(options);
  paymentObject.open();
};

const handleRazorpay = async () => {
  try {
    setLoading(true);
    razorPay();
    const res = await axios.post("/api/v1/product/braintree/payment", {
      products: cart,
      buyer: {
        _id: {
          $oid: "640cddcb0e745904efe650b7",
        },
        name: "user",
        email: "user@gmail.com",
        password:
          "$2b$10$xZa7qqO8wkxoQu7VysPY.ux4BhMKyd6ctbv6edXIoGqi.4L81OIqq",
        phone: "9867080692",
        address: "Mumbai India",
        answer: "Football",
        role: 1,
        createdAt: {
          $date: {
            $numberLong: "1678564811529",
          },
        },
        updatedAt: {
          $date: {
            $numberLong: "1678564811529",
          },
        },
        __v: 0,
      },
    });
    setLoading(false);
    localStorage.removeItem("cart");
    setCart([]);
    navigate("/dashboard/user/orders");
    toast.success("Payment Completed Successfully ");
  } catch (error) {
    console.log(error);
    setLoading(false);
  }
};
return (
  <Layout>
    <div className=" cart-page">
      <div className="row">
        <div className="col-md-12">
          <h1 className="text-center bg-light p-2 mb-1">
            {!auth?.user
        ? "Hello Guest"
        : `Hello  ${auth?.token && auth?.user?.name}`}
      <p className="text-center">
        {cart?.length
          ? `You Have ${cart.length} items in your cart ${
              auth?.token ? "" : "please login to checkout !"
            }`
          : " Your Cart Is Empty"}
      </p>
    </h1>
  </div>
</div>
<div className="container ">
  <div className="row ">
    <div className="col-md-7  p-0 m-0">
      {cart?.map((p) => (
        <div className="row card flex-row" key={p._id}>
          <div className="col-md-4">
            <img
              src={`/api/v1/product/product-photo/${p._id}`}
              className="card-img-top"
              alt={p.name}
              width="100%"
              height={"130px"}
            />
          </div>
          <div className="col-md-4">
            <p>{p.name}</p>
            <p>{p.description.substring(0, 30)}</p>
            <p>Price : {p.price}</p>
          </div>
          <div className="col-md-4 cart-remove-btn">
            <button
              className="btn btn-danger"
              onClick={() => removeCartItem(p._id)}
            >
              Remove
            </button>
          </div>
        </div>
      ))}
    </div>
    <div className="col-md-5 cart-summary ">
      <h2>Cart Summary</h2>
      <p>Total | Checkout | Payment</p>
      <hr />
      <h4>Total : {totalPrice()} </h4>
      {auth?.user?.address ? (
        <>
          <div className="mb-3">
            <h4>Current Address</h4>
            <h5>{auth?.user?.address}</h5>
            <button
              className="btn btn-outline-warning"
              onClick={() => navigate("/dashboard/user/profile")}
            >
              Update Address
            </button>
          </div>
        </>
      ) : (
        <div className="mb-3">
          {auth?.token ? (
            <button
              className="btn btn-outline-warning"
              onClick={() => navigate("/dashboard/user/profile")}
            >
              Update Address
            </button>
          ) : (
            <button
              className="btn btn-outline-warning"
              onClick={() =>
                navigate("/login", {
                  state: "/cart",
                })
              }
            >
              Plase Login to checkout
            </button>
          )}
        </div>
      )}
      <div className="mt-2">
        {!clientToken || !auth?.token || !cart?.length ? (
          ""
        ) : (
          <>
            <DropIn
              options={{
                authorization: clientToken,
                // paypal: {
                //   flow: "vault",
                // },
              }}
              onInstance={(instance) => setInstance(instance)}
            />

            <button
              className="btn btn-primary"
              onClick={handlePayment}
              disabled={loading || !instance || !auth?.user?.address}
            >
              {loading ? "Processing ...." : "Pay with card"}
```

```
          </button>
          <button
            className="m-4 btn btn-success"
            onClick={handleRazorpay}
          >
            Pay with Razorpay
          </button>
        </>
      )}
    </div>
   </div>
  </div>
 </div>
</div>
</Layout>
);
};

export default CartPage;
```

9. Categories.js

```
import React, { useState, useEffect } from "react";
import { Link } from "react-router-dom";
import useCategory from "../hooks/useCategory";
import Layout from "../components/Layout/Layout";
const Categories = () => {
  const categories = useCategory();
  return (
    <Layout title={"All Categories"}>
      <div className="container" style={{ marginTop: "100px" }}>
        <div className="row container">
          {categories.map((c) => (
            <div className="col-md-4 mt-5 mb-3 gx-3 gy-3" key={c._id}>
              <div className="card">
                <Link to={`/category/${c.slug}`} className="btn cat-btn">
                  {c.name}
                </Link>
              </div>
            </div>
          ))}
        </div>
      </div>
    </Layout>
  );
};
```

```
export default Categories;;
```

10.ProductDetails.js

```
import React, { useState, useEffect } from "react";
import Layout from "./../components/Layout/Layout";
import axios from "axios";
import { useParams, useNavigate } from "react-router-dom";
import "../styles/ProductDetailsStyles.css";
import toast from "react-hot-toast";
import { useCart } from "../context/cart";

const ProductDetails = () => {
  const params = useParams();
  const navigate = useNavigate();
  const [product, setProduct] = useState({});
  const [relatedProducts, setRelatedProducts] = useState([]);
  const [cart, setCart] = useCart();

  //initalp details
  useEffect(() => {
    if (params?.slug) getProduct();
  }, [params?.slug]);
  //getProduct
  const getProduct = async () => {
    try {
      const { data } = await axios.get(
        `/api/v1/product/get-product/${params.slug}`
      );
      setProduct(data?.product);
      getSimilarProduct(data?.product._id, data?.product.category._id);
    } catch (error) {
      console.log(error);
    }
  };
  //get similar product
  const getSimilarProduct = async (pid, cid) => {
    try {
      const { data } = await axios.get(
        `/api/v1/product/related-product/${pid}/${cid}`
      );
      setRelatedProducts(data?.products);
    } catch (error) {
      console.log(error);
    }
  };
  return (
    <Layout>
      <div className="row container product-details">
        <div className="col-md-6">
          <img
            src={`/api/v1/product/product-photo/${product._id}`}
            className="card-img-top"
            alt={product.name}
            height="300"
            width={"350px"}
          />
        </div>
        <div className="col-md-6 product-details-info">
          <h1 className="text-center">Product Details</h1>
          <hr />
          <h6>Name : {product.name}</h6>
          <h6>Description : {product.description}</h6>
          <h6>
            Price :
            {product?.price?.toLocaleString("en-US", {
              style: "currency",
              currency: "INR",
            })}
          </h6>
          <h6>Category : {product?.category?.name}</h6>
          <button
            class="btn btn-secondary ms-1"
            onClick={() => {
              setCart([...cart, product]);
              localStorage.setItem("cart", JSON.stringify([...cart, product]));
              toast.success("Item Added to cart");
            }}
          >
            ADD TO CART
          </button>
        </div>
      </div>
      <hr />
      <div className="row container similar-products">
        <h4>Similar Products ➡️</h4>
        {relatedProducts.length < 1 && (
          <p className="text-center">No Similar Products found</p>
        )}
        <div className="d-flex flex-wrap">
          {relatedProducts?.map((p) => (
            <div className="card m-2" key={p._id}>
              <img
                src={`/api/v1/product/product-photo/${p._id}`}
                className="card-img-top"
                alt={p.name}
              />
              <div className="card-body">
                <div className="card-name-price">
                  <h5 className="card-title">{p.name}</h5>
                  <h5 className="card-title card-price">
                    {p.price.toLocaleString("en-US", {
                      style: "currency",
                      currency: "INR",
                    })}
                  </h5>
                </div>
                <p className="card-text ">
                  {p.description.substring(0, 60)}...
                </p>
                <div className="card-name-price">
                  <button
                    className="btn btn-info ms-1"
                    onClick={() => navigate(`/product/${p.slug}`)}
                  >
                    More Details
                  </button>
                  {/* <button
                  className="btn btn-dark ms-1"
                  onClick={() => {
                    setCart([...cart, p]);
                    localStorage.setItem(
                      "cart",
                      JSON.stringify([...cart, p])
                    );
                    toast.success("Item Added to cart");
                  }}
                >
                  ADD TO CART
                </button> */}
                </div>
              </div>
```

```
          </div>
        ))}
      </div>
    </div>
  </Layout>
 );
};

export default ProductDetails;s;
```

11.Search.js

```
import React from "react";
import Layout from "./../components/Layout/Layout";
import { useSearch } from "../context/search";
const Search = () => {
 const [values, setValues] = useSearch();
 return (
  <Layout title={"Search results"}>
   <div className="container">
    <div className="text-center">
     <h1>Search Resuts</h1>
     <h6>
      {values?.results.length < 1
       ? "No Products Found"
       : `Found ${values?.results.length}`}
     </h6>
```

```
          <div className="d-flex flex-wrap mt-4">
            {values?.results.map((p) => (
              <div className="card m-2" style={{ width: "18rem" }}>
                <img
                  src={`/api/v1/product/product-photo/${p._id}`}
                  className="card-img-top"
                  alt={p.name}
                />
                <div className="card-body">
                  <h5 className="card-title">{p.name}</h5>
                  <p className="card-text">
                    {p.description.substring(0, 30)}...
                  </p>
                  <p className="card-text"> $ {p.price}</p>
                  <button class="btn btn-primary ms-1">More Details</button>
                  <button class="btn btn-secondary ms-1">ADD TO CART</button>
                </div>
              </div>
            ))}
          </div>
        </div>
      </div>
    </Layout>
  );
};
```

```
export default Search;
```

12.App.js

```
import { Routes, Route } from "react-router-dom";
import HomePage from "./pages/HomePage";
import About from "./pages/About";
import Contact from "./pages/Contact";
import Policy from "./pages/Policy";
import Pagenotfound from "./pages/Pagenotfound";
import Register from "./pages/Auth/Register";
import Login from "./pages/Auth/Login";
import Dashboard from "./pages/user/Dashboard";
import PrivateRoute from "./components/Routes/Private";
import ForgotPasssword from "./pages/Auth/ForgotPasssword";
import AdminRoute from "./components/Routes/AdminRoute";
import AdminDashboard from "./pages/Admin/AdminDashboard";
import CreateCategory from "./pages/Admin/CreateCategory";
import CreateProduct from "./pages/Admin/CreateProduct";
import Users from "./pages/Admin/Users";
import Orders from "./pages/user/Orders";
import Profile from "./pages/user/Profile";
import Products from "./pages/Admin/Products";
import UpdateProduct from "./pages/Admin/UpdateProduct";
import Search from "./pages/Search";
import ProductDetails from "./pages/ProductDetails";
import Categories from "./pages/Categories";
import CategoryProduct from "./pages/CategoryProduct";
import CartPage from "./pages/CartPage";
import AdminOrders from "./pages/Admin/AdminOrders";
function App() {
  return (
    <>
      <Routes>
        <Route path="/" element={<HomePage />} />
        <Route path="/product/:slug" element={<ProductDetails />} />
        <Route path="/categories" element={<Categories />} />
        <Route path="/cart" element={<CartPage />} />
        <Route path="/category/:slug" element={<CategoryProduct />} />
        <Route path="/search" element={<Search />} />
        <Route path="/dashboard" element={<PrivateRoute />}>
          <Route path="user" element={<Dashboard />} />
          <Route path="user/orders" element={<Orders />} />
          <Route path="user/profile" element={<Profile />} />
        </Route>
        <Route path="/dashboard" element={<AdminRoute />}>
          <Route path="admin" element={<AdminDashboard />} />
          <Route path="admin/create-category" element={<CreateCategory />} />
          <Route path="admin/create-product" element={<CreateProduct />} />
          <Route path="admin/product/:slug" element={<UpdateProduct />} />
          <Route path="admin/products" element={<Products />} />
          <Route path="admin/users" element={<Users />} />
          <Route path="admin/orders" element={<AdminOrders />} />
        </Route>
        <Route path="/register" element={<Register />} />
        <Route path="/forgot-password" element={<ForgotPasssword />} />
        <Route path="/login" element={<Login />} />
        <Route path="/about" element={<About />} />
        <Route path="/contact" element={<Contact />} />
        <Route path="/policy" element={<Policy />} />
        <Route path="*" element={<Pagenotfound />} />
      </Routes>
    </>
  );
}

export default App;
```

13.Index.js

```
import React from "react";
import ReactDOM from "react-dom/client";
import "./index.css";
import App from "./App";
import reportWebVitals from "./reportWebVitals";
import { BrowserRouter } from "react-router-dom";
import { AuthProvider } from "./context/auth";
import { SearchProvider } from "./context/search";
import { CartProvider } from "./context/cart";
import "antd/dist/reset.css";

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

```
root.render(
  <AuthProvider>
    <SearchProvider>
      <CartProvider>
        <BrowserRouter>
          <App />
        </BrowserRouter>
      </CartProvider>
    </SearchProvider>
  </AuthProvider>
);


// If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();
```

13.index.css

```
@import url("https://fonts.googleapis.com/css2?family=Playfair+Display:wght@700&family=Poppins:wght@300;400&display=swap");


* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}
```

```
/* font-family: 'Poppins', sans-serif; */
/* font-family: 'Playfair Display', serif; */

/* //navbar css */
.navbar {
  font-family: "Poppins", sans-serif;
  font-size: 17px;
  line-height: 26px;
  text-transform: uppercase;
  box-shadow: 0 8px 6px -6px gray;
  --webkit-box-shadow: 0 8px 6px -6px gray;
  border-bottom: solid gray !important;
}

.nav-link {
  font-weight: 300 !important;
}

.active {
  border-bottom: 2px solid black;
}

.navbar-brand {
  font-weight: 700;
  font-family: "roboto", sans-serif;
  letter-spacing: 3px;
}

.search-form {
  margin-right: 10px;
  margin-top: 4px;
}
.search-form input {
  border: none;
  border-radius: 0;
}
.search-form button {
  background-color: #000000;
  border-radius: 0;
  color: white;
}

/* =============================
==========footer============= */

.footer {
  color: white;
  padding: 25px;
  background: #000000; /* fallback for old browsers */
  background: -webkit-linear-gradient(
    to right,
    #434343,
    #000000
  ); /* Chrome 10-25, Safari 5.1-6 */
  background: linear-gradient(to right, #434343, #000000);
}

.footer a {
  text-decoration: none;
  color: white;
  padding: 10px;
}

.footer a:hover {
  color: #ffefba;
  border-bottom: 1px solid #ffefba;
}
/* ==================================
======== page not found css ======= */
.pnf {
  display: flex;
  min-height: 65vh;
  flex-direction: column;
  align-items: center;
  justify-content: center;
}
```

```
.pnf-title {
  font-size: 100px;
  font-weight: 700;
}

.pnf-heading {
  font-weight: normal;
}

.pnf-btn {
  color: black;
  border: 1px solid black;
  text-decoration: none;
  padding: 10px;
  margin-top: 10px;
}
.pnf-btn:hover {
  background-color: black;
  color: white;
}
/* ========================================= */
/* =========contact us ========= */
.contactus {
  margin: 0;
  padding: 0;
  height: 70vh;
  display: flex;
  align-items: center;
  justify-content: center;
}

/* ===================== */
.product-link {
  text-decoration: none !important;
  color: black !important;
}

.cat-btn {
  padding: 40px 0px;
  font-size: 24px;
  text-transform: uppercase;
}
.cat-btn:hover {
  background-color: #434343;
  color: white;
}
/* ================
====dashboard
=================== */

.dashboard {
  margin-top: 100px !important;
```

```
}

.dashboard-menu h4 {
  background-color: #434343 !important;
  color: white;
  padding: 20px 0px;
```

- }**Admin**

  AdminDashboard.js

  ```
  import React from "react";
  import AdminMenu from "../../components/Layout/AdminMenu";
  import Layout from "./../../components/Layout/Layout";
  import { useAuth } from "../../context/auth";
  const AdminDashboard = () => {
    const [auth] = useAuth();
    return (
      <Layout>
        <div className="container-fluid m-3 p-3 dashboard">
          <div className="row">
            <div className="col-md-3">
              <AdminMenu />
            </div>
            <div className="col-md-9">
              <div className="card w-75 p-3">
                <h3> Admin Name : {auth?.user?.name}</h3>
                <h3> Admin Email : {auth?.user?.email}</h3>
  ```

```
          <h3> Admin Contact : {auth?.user?.phone}</h3>
         </div>
        </div>
       </div>
      </div>
     </Layout>
    );
   };

   export default AdminDashboard;
```

AdminOrders.js

```
import React, { useState, useEffect } from "react";
import axios from "axios";
import toast from "react-hot-toast";
import AdminMenu from "../../components/Layout/AdminMenu";
import Layout from "../../components/Layout/Layout";
import { useAuth } from "../../context/auth";
import moment from "moment";
import { Select } from "antd";
const { Option } = Select;

const AdminOrders = () => {
  const [status, setStatus] = useState([
    "Not Process",
    "Processing",
    "Shipped",
    "deliverd",
    "cancel",
  ]);
  const [changeStatus, setCHangeStatus] = useState("");
  const [orders, setOrders] = useState([]);
  const [auth, setAuth] = useAuth();
  const getOrders = async () => {
    try {
      const { data } = await axios.get("/api/v1/auth/all-orders");
      setOrders(data);
    } catch (error) {
      console.log(error);
    }
  };

  useEffect(() => {
    if (auth?.token) getOrders();
  }, [auth?.token]);

  const handleChange = async (orderId, value) => {
    try {
      const { data } = await axios.put(`/api/v1/auth/order-status/${orderId}`, {
        status: value,
      });
      getOrders();
    } catch (error) {
      console.log(error);
    }
  };
  return (
    <Layout title={"All Orders Data"}>
      <div className="row dashboard">
        <div className="col-md-3">
          <AdminMenu />
        </div>
        <div className="col-md-9">
          <h1 className="text-center">All Orders</h1>
          {orders?.map((o, i) => {
            return (
              <div className="border shadow">
                <table className="table">
                  <thead>
                    <tr>
                      <th scope="col">#</th>
                      <th scope="col">Status</th>
                      <th scope="col">Buyer</th>
                      <th scope="col"> date</th>
                      <th scope="col">Payment</th>
                      <th scope="col">Quantity</th>
                    </tr>
                  </thead>
                  <tbody>
      <tr>
        <td>{i + 1}</td>
        <td>
          <Select
            bordered={false}
            onChange={(value) => handleChange(o._id, value)}
            defaultValue={o?.status}
          >
            {status.map((s, i) => (
              <Option key={i} value={s}>
                {s}
              </Option>
            ))}
          </Select>
        </td>
        <td>{o?.buyer?.name}</td>
        <td>{moment(o?.createAt).fromNow()}</td>
        <td>{o?.payment.success ? "Success" : "Success"}</td>
        <td>{o?.products?.length}</td>
      </tr>
    </tbody>
  </table>
  <div className="container">
    {o?.products?.map((p, i) => (
      <div className="row mb-2 p-3 card flex-row" key={p._id}>
        <div className="col-md-4">
```

```
                        <img
                          src={`/api/v1/product/product-photo/${p._id}`}
                          className="card-img-top"
                          alt={p.name}
                          width="100px"
                          height={"100px"}
                        />
                      </div>
                      <div className="col-md-8">
                        <p>{p.name}</p>
                        <p>{p.description.substring(0, 30)}</p>
                        <p>Price : {p.price}</p>
                      </div>
                    </div>
                  ))}
                </div>
              </div>
            );
          })}
        </div>
      </div>
    </Layout>
  );
};

export default AdminOrders;
```

1. **CreateCategory.js**

```
import React, { useEffect, useState } from "react";
import Layout from "./../../components/Layout/Layout";
import AdminMenu from "./../../components/Layout/AdminMenu";
import toast from "react-hot-toast";
import axios from "axios";
import CategoryForm from "../../components/Form/CategoryForm";
import { Modal } from "antd";
const CreateCategory = () => {
  const [categories, setCategories] = useState([]);
  const [name, setName] = useState("");
  const [visible, setVisible] = useState(false);
  const [selected, setSelected] = useState(null);
  const [updatedName, setUpdatedName] = useState("");
  //handle Form
  const handleSubmit = async (e) => {
    e.preventDefault();
    try {
      const { data } = await axios.post("/api/v1/category/create-category", {
        name,
      });
      if (data?.success) {
        toast.success(`${name} is created`);
        getAllCategory();
      } else {
        toast.error(data.message);
      }
    } catch (error) {
      console.log(error);
      // toast.error("somthing went wrong in input form");
    }
  };

  //get all cat
  const getAllCategory = async () => {
    try {
      const { data } = await axios.get("/api/v1/category/get-category");
      if (data?.success) {
        setCategories(data?.category);
      }
    } catch (error) {
      console.log(error);
      toast.error("Something wwent wrong in getting catgeory");
    }
  };

  useEffect(() => {
    getAllCategory();
  }, []);

  //update category
  const handleUpdate = async (e) => {
    e.preventDefault();
    try {
      const { data } = await axios.put(
        `/api/v1/category/update-category/${selected._id}`,
        { name: updatedName }
      );
      if (data?.success) {
        toast.success(`${updatedName} is updated`);
        setSelected(null);
        setUpdatedName("");
        setVisible(false);
        getAllCategory();
      } else {
        toast.error(data.message);
      }
    } catch (error) {
      console.log(error);
    }
  };
  //delete category
  const handleDelete = async (pId) => {
    try {
      const { data } = await axios.delete(
        `/api/v1/category/delete-category/${pId}`
      );
      if (data.success) {
        toast.success(`category is deleted`);
        getAllCategory();
      } else {
        toast.error(data.message);
      }
    } catch (error) {
      toast.error("Somtihing went wrong");
    }
  };
  return (
    <Layout title={"Dashboard - Create Category"}>
      <div className="container-fluid m-3 p-3 dashboard">
        <div className="row">
          <div className="col-md-3">
            <AdminMenu />
          </div>
          <div className="col-md-9">
            <h1>Manage Category</h1>
            <div className="p-3 w-50">
              <CategoryForm
                handleSubmit={handleSubmit}
                value={name}
                setValue={setName}
              />
            </div>
            <div className="w-75">
              <table className="table">
                <thead>
                  <tr>
                    <th scope="col">Name</th>
                    <th scope="col">Actions</th>
                  </tr>
                </thead>
                <tbody>
                  {categories?.map((c) => (
                    <>
                      <tr>
                        <td key={c._id}>{c.name}</td>
                        <td>
                          <button
                            className="btn btn-primary ms-2"
```

```
                            onClick={() => {
                              setVisible(true);
                              setUpdatedName(c.name);
                              setSelected(c);
                            }}
                          >
                            Edit
                          </button>
                          <button
                            className="btn btn-danger ms-2"
                            onClick={() => {
                              handleDelete(c._id);
                            }}
                          >
                            Delete
                          </button>
                        </td>
                      </tr>
                    </>
                  ))}
                </tbody>
              </table>
            </div>
            <Modal
              onCancel={() => setVisible(false)}
              footer={null}
              visible={visible}
            >
              <CategoryForm
                value={updatedName}
                setValue={setUpdatedName}
                handleSubmit={handleUpdate}
              />
            </Modal>
          </div>
        </div>
      </div>
    </Layout>
  );
};
```

```
export default CreateCategory;
```

- User

1. User.js

```
import React from "react";
import AdminMenu from "../../components/Layout/AdminMenu";
import Layout from "./../../components/Layout/Layout";

const Users = () => {
  return (
    <Layout title={"Dashboard - All Users"}>
      <div className="container-fluid m-3 p-3">
        <div className="row">
          <div className="col-md-3">
            <AdminMenu />
          </div>
          <div className="col-md-9">
            <h1>All Users</h1>
          </div>
        </div>
      </div>
    </Layout>
  );
};

export default Users;
```

2. Orders.js

```
import React, { useState, useEffect } from "react";
import UserMenu from "../../components/Layout/UserMenu";
import Layout from "./../../components/Layout/Layout";
import axios from "axios";
import { useAuth } from "../../context/auth";
import moment from "moment";

const Orders = () => {
  const [orders, setOrders] = useState([]);
  const [auth, setAuth] = useAuth();
  const getOrders = async () => {
    try {
      const { data } = await axios.get("/api/v1/auth/orders");
      setOrders(data);
    } catch (error) {
      console.log(error);
    }
  };

  useEffect(() => {
    if (auth?.token) getOrders();
  }, [auth?.token]);
  return (
    <Layout title={"Your Orders"}>
```

```
<div className="container-flui p-3 m-3 dashboard">
  <div className="row">
    <div className="col-md-3">
      <UserMenu />
    </div>
    <div className="col-md-9">
      <h1 className="text-center">All Orders</h1>
      {orders?.map((o, i) => {
        return (
          <div className="border shadow">
            <table className="table">
              <thead>
                <tr>
                  <th scope="col">#</th>
                  <th scope="col">Status</th>
                  <th scope="col">Buyer</th>
                  <th scope="col"> date</th>
                  <th scope="col">Payment</th>
                  <th scope="col">Quantity</th>
                </tr>
              </thead>
              <tbody>
                <tr>
                  <td>{i + 1}</td>
                  <td>{o?.status}</td>
                  <td>{o?.buyer?.name}</td>
                  <td>{moment(o?.createAt).fromNow()}</td>
                  <td>{o?.payment.success ? "Success" : "Success"}</td>
                  <td>{o?.products?.length}</td>
                </tr>
              </tbody>
            </table>
            <div className="container">
              {o?.products?.map((p, i) => (
                <div className="row mb-2 p-3 card flex-row" key={p._id}>
                  <div className="col-md-4">
                    <img
                      src={`/api/v1/product/product-photo/${p._id}`}
                      className="card-img-top"
                      alt={p.name}
                      width="100px"
                      height={"100px"}
                    />
                  </div>
                  <div className="col-md-8">
                    <p>{p.name}</p>
                    <p>{p.description.substring(0, 30)}</p>
                    <p>Price : {p.price}</p>
                  </div>
                </div>
              ))}
            </div>
```

```
        </div>
       );
      })}
     </div>
    </div>
   </div>
  </Layout>
 );
};

export default Orders;
```

controllers :

- authcontroller.js

```
import userModel from "../models/userModel.js";
import orderModel from "../models/orderModel.js";

import { comparePassword, hashPassword } from "./../helpers/authHelper.js";
import JWT from "jsonwebtoken";

export const registerController = async (req, res) => {
 try {
  const { name, email, password, phone, address, answer } = req.body;
  //validations
  if (!name) {
   return res.send({ error: "Name is Required" });
  }
  if (!email) {
```

```
  return res.send({ message: "Email is Required" });
}
if (!password) {
  return res.send({ message: "Password is Required" });
}
if (!phone) {
  return res.send({ message: "Phone no is Required" });
}
if (!address) {
  return res.send({ message: "Address is Required" });
}
if (!answer) {
  return res.send({ message: "Answer is Required" });
}
//check user
const exisitingUser = await userModel.findOne({ email });
//exisiting user
if (exisitingUser) {
  return res.status(200).send({
    success: false,
    message: "Already Register please login",
  });
}
//register user
const hashedPassword = await hashPassword(password);
//save
```

```
    const user = await new userModel({
      name,
      email,
      phone,
      address,
      password: hashedPassword,
      answer,
    }).save();

    res.status(201).send({
      success: true,
      message: "User Register Successfully",
      user,
    });
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      message: "Errro in Registeration",
      error,
    });
  }
};

//POST LOGIN
export const loginController = async (req, res) => {
```

```
try {
  const { email, password } = req.body;
  //validation
  if (!email || !password) {
    return res.status(404).send({
      success: false,
      message: "Invalid email or password",
    });
  }
  //check user
  const user = await userModel.findOne({ email });
  if (!user) {
    return res.status(404).send({
      success: false,
      message: "Email is not registerd",
    });
  }
  const match = await comparePassword(password, user.password);
  if (!match) {
    return res.status(200).send({
      success: false,
      message: "Invalid Password",
    });
  }
  //token
  const token = await JWT.sign({ _id: user._id }, process.env.JWT_SECRET, {
      expiresIn: "7d",
    });
    res.status(200).send({
      success: true,
      message: "login successfully",
      user: {
        _id: user._id,
        name: user.name,
        email: user.email,
        phone: user.phone,
        address: user.address,
        role: user.role,
      },
      token,
    });
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      message: "Error in login",
      error,
    });
  }
};

//forgotPasswordController
```

```
export const forgotPasswordController = async (req, res) => {
  try {
    const { email, answer, newPassword } = req.body;
    if (!email) {
      res.status(400).send({ message: "Emai is required" });
    }
    if (!answer) {
      res.status(400).send({ message: "answer is required" });
    }
    if (!newPassword) {
      res.status(400).send({ message: "New Password is required" });
    }
    //check
    const user = await userModel.findOne({ email, answer });
    //validation
    if (!user) {
      return res.status(404).send({
        success: false,
        message: "Wrong Email Or Answer",
      });
    }
    const hashed = await hashPassword(newPassword);
    await userModel.findByIdAndUpdate(user._id, { password: hashed });
    res.status(200).send({
      success: true,
      message: "Password Reset Successfully",
    });
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      message: "Something went wrong",
      error,
    });
  }
};

//test controller
export const testController = (req, res) => {
  try {
    res.send("Protected Routes");
  } catch (error) {
    console.log(error);
    res.send({ error });
  }
};

//update prfole
export const updateProfileController = async (req, res) => {
  try {
    const { name, email, password, address, phone } = req.body;
    const user = await userModel.findById(req.user._id);
    //password
    if (password && password.length < 6) {
      return res.json({ error: "Passsword is required and 6 character long" });
    }
    const hashedPassword = password ? await hashPassword(password) : undefined;
    const updatedUser = await userModel.findByIdAndUpdate(
      req.user._id,
      {
        name: name || user.name,
        password: hashedPassword || user.password,
        phone: phone || user.phone,
        address: address || user.address,
      },
      { new: true }
    );
    res.status(200).send({
      success: true,
      message: "Profile Updated SUccessfully",
      updatedUser,
    });
  } catch (error) {
    console.log(error);
    res.status(400).send({
      success: false,
      message: "Error WHile Update profile",
      error,
    });
  }
};

//orders
export const getOrdersController = async (req, res) => {
  try {
    const orders = await orderModel
      .find({ buyer: req.user._id })
      .populate("products", "-photo")
      .populate("buyer", "name");
    res.json(orders);
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      message: "Error WHile Geting Orders",
      error,
    });
  }
};
//orders
export const getAllOrdersController = async (req, res) => {
  try {
    const orders = await orderModel
```

```
      .find({})
      .populate("products", "-photo")
      .populate("buyer", "name")
      .sort({ createdAt: "-1" });
    res.json(orders);
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      message: "Error WHile Geting Orders",
      error,
    });
  }
};

//order status
export const orderStatusController = async (req, res) => {
  try {
    const { orderId } = req.params;
    const { status } = req.body;
    const orders = await orderModel.findByIdAndUpdate(
      orderId,
      { status },
      { new: true }
    );
    res.json(orders);
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      message: "Error While Updateing Order",
      error,
    });
  }
};
```

- **categoryController.js**

```
import categoryModel from "../models/categoryModel.js";
import slugify from "slugify";
export const createCategoryController = async (req, res) => {
  try {
    const { name } = req.body;
    if (!name) {
      return res.status(401).send({ message: "Name is required" });
    }
    const existingCategory = await categoryModel.findOne({ name });
    if (existingCategory) {
      return res.status(200).send({
        success: false,
        message: "Category Already Exisits",
      });
    }
    const category = await new categoryModel({
      name,
      slug: slugify(name),
    }).save();
    res.status(201).send({
      success: true,
      message: "new category created",
      category,
    });
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      errro,
      message: "Errro in Category",
    });
  }
};

//update category
export const updateCategoryController = async (req, res) => {
  try {
    const { name } = req.body;
    const { id } = req.params;
    const category = await categoryModel.findByIdAndUpdate(
      id,
      { name, slug: slugify(name) },
      { new: true }
    );
    res.status(200).send({
      success: true,
      messsage: "Category Updated Successfully",
      category,
    });
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      error,
      message: "Error while updating category",
    });
  }
};

// get all cat
export const categoryControlller = async (req, res) => {
  try {
    const category = await categoryModel.find({});
    res.status(200).send({
      success: true,
      message: "All Categories List",
      category,
    });
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      error,
      message: "Error while getting all categories",
    });
  }
};
```

```
// single category
export const singleCategoryController = async (req, res) => {
  try {
    const category = await categoryModel.findOne({ slug: req.params.slug });
    res.status(200).send({
      success: true,
      message: "Get SIngle Category SUccessfully",
      category,
    });
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      error,
      message: "Error While getting Single Category",
    });
  }
};

//delete category
export const deleteCategoryCOntroller = async (req, res) => {
  try {
    const { id } = req.params;
    await categoryModel.findByIdAndDelete(id);
    res.status(200).send({
      success: true,
      message: "Categry Deleted Successfully",
    });
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      message: "error while deleting category",
      error,
    });
  }
}
```

- ;**productController.js**

```
import productModel from "../models/productModel.js";
import categoryModel from "../models/categoryModel.js";
import orderModel from "../models/orderModel.js";

import fs from "fs";
import slugify from "slugify";
import braintree from "braintree";
import dotenv from "dotenv";

dotenv.config();

// payment gateway
var gateway = new braintree.BraintreeGateway({
  environment: braintree.Environment.Sandbox,
  merchantId: process.env.BRAINTREE_MERCHANT_ID,
  publicKey: process.env.BRAINTREE_PUBLIC_KEY,
  privateKey: process.env.BRAINTREE_PRIVATE_KEY,
});

export const createProductController = async (req, res) => {
  try {
    const { name, description, price, category, quantity, shipping } =
      req.fields;
    const { photo } = req.files;
    //alidation
    switch (true) {
      case !name:
        return res.status(500).send({ error: "Name is Required" });
      case !description:
        return res.status(500).send({ error: "Description is Required" });
      case !price:
        return res.status(500).send({ error: "Price is Required" });
      case !category:
        return res.status(500).send({ error: "Category is Required" });
      case !quantity:
        return res.status(500).send({ error: "Quantity is Required" });
      case photo && photo.size > 1000000:
        return res
          .status(500)
          .send({ error: "photo is Required and should be less then 1mb" });
    }

    const products = new productModel({ ...req.fields, slug: slugify(name) });
    if (photo) {
      products.photo.data = fs.readFileSync(photo.path);
      products.photo.contentType = photo.type;
    }
    await products.save();
    res.status(201).send({
      success: true,
      message: "Product Created Successfully",
      products,
    });
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      error,
      message: "Error in crearing product",
    });
  }
};

//get all products
export const getProductController = async (req, res) => {
  try {
    const products = await productModel
      .find({})
      .populate("category")
      .select("-photo")
      .limit(12)
      .sort({ createdAt: -1 });
    res.status(200).send({
      success: true,
      counTotal: products.length,
      message: "ALlProducts ",
      products,
    });
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      message: "Erorr in getting products",
      error: error.message,
    });
  }
};
// get single product
export const getSingleProductController = async (req, res) => {
  try {
    const product = await productModel
```

```
      .findOne({ slug: req.params.slug })
      .select("-photo")
      .populate("category");
    res.status(200).send({
      success: true,
      message: "Single Product Fetched",
      product,
    });
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      message: "Eror while getitng single product",
      error,
    });
  }
};

// get photo
export const productPhotoController = async (req, res) => {
  try {
    const product = await productModel.findById(req.params.pid).select("photo");
    if (product.photo.data) {
      res.set("Content-type", product.photo.contentType);
      return res.status(200).send(product.photo.data);
    }
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      message: "Erorr while getting photo",
      error,
    });
  }
};

//delete controller
export const deleteProductController = async (req, res) => {
  try {
    await productModel.findByIdAndDelete(req.params.pid).select("-photo");
    res.status(200).send({
      success: true,
      message: "Product Deleted successfully",
    });
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      message: "Error while deleting product",
      error,
    });
  }
```

```
};

//upate producta
export const updateProductController = async (req, res) => {
  try {
    const { name, description, price, category, quantity, shipping } =
      req.fields;
    const { photo } = req.files;
    //alidation
    switch (true) {
      case !name:
        return res.status(500).send({ error: "Name is Required" });
      case !description:
        return res.status(500).send({ error: "Description is Required" });
      case !price:
        return res.status(500).send({ error: "Price is Required" });
      case !category:
        return res.status(500).send({ error: "Category is Required" });
      case !quantity:
        return res.status(500).send({ error: "Quantity is Required" });
      case photo && photo.size > 1000000:
        return res
          .status(500)
          .send({ error: "photo is Required and should be less then 1mb" });
    }
    const products = await productModel.findByIdAndUpdate(
      req.params.pid,
      { ...req.fields, slug: slugify(name) },
      { new: true }
    );
    if (photo) {
      products.photo.data = fs.readFileSync(photo.path);
      products.photo.contentType = photo.type;
    }
    await products.save();
    res.status(201).send({
      success: true,
      message: "Product Updated Successfully",
      products,
    });
  } catch (error) {
    console.log(error);
    res.status(500).send({
      success: false,
      error,
      message: "Error in Updte product",
    });
  }
};

// filters
```

```
export const productFiltersController = async (req, res) => {
  try {
    const { checked, radio } = req.body;
    let args = {};
    if (checked.length > 0) args.category = checked;
    if (radio.length) args.price = { $gte: radio[0], $lte: radio[1] };
    const products = await productModel.find(args);
    res.status(200).send({
      success: true,
      products,
    });
  } catch (error) {
    console.log(error);
    res.status(400).send({
      success: false,
      message: "Error WHile Filtering Products",
      error,
    });
  }
};

// product count
export const productCountController = async (req, res) => {
  try {
    const total = await productModel.find({}).estimatedDocumentCount();
    res.status(200).send({
      success: true,
      total,
    });
  } catch (error) {
    console.log(error);
    res.status(400).send({
      message: "Error in product count",
      error,
      success: false,
    });
  }
};x`
export const createOrder = async (req, res) => {
  const { products, buyer } = req.body;
  const payment = {
    success: true,
  };
  if (products && buyer) {
    const order = new orderModel({
      products: products,
      buyer: buyer,
      payment: payment,
    });
    res.status(200).json({ ok: true });
  }
};
```

1. **models**
    - **orderModel.js**

```
import mongoose from "mongoose";
const orderSchema = new mongoose.Schema(
{
products: [ {type: mongoose.ObjectId, ref: "Products",},],
payment: {},
buyer: { type: mongoose.ObjectId, ref: "users",},
status: { type: String, default: "Not Process",
enum: ["Not Process", "Processing", "Shipped", "deliverd", "cancel"], },},
{ timestamps: true } );
export default mongoose.model("Order", orderSchema);
```

- **productModel.js**

```
import mongoose from "mongoose";
const productSchema = new mongoose.Schema(
{
name: {type: String,required: true,},
slug: {type: String,required: true,},
description: {type: String,required: true,},
price: {type: Number,required: true,},
category: {type: mongoose.ObjectId,ref: "Category",required: true,},
quantity: {type: Number,required: true,},
photo: {data: Buffer,contentType: String,},
shipping: {type: Boolean,},},
{ timestamps: true });
export default mongoose.model("Products", productSchema);
```

2. **routes**
    - **authRoute.js**

```
import express from "express";
import {
registerController, loginController, testController, forgotPasswordController,
updateProfileController, getOrdersController, getAllOrdersController, orderStatusController,
} from "../controllers/authController.js";
import { isAdmin, requireSignIn } from "../middlewares/authMiddleware.js";

//router object
const router = express.Router();
//routing
//REGISTER || METHOD POST
router.post("/register", registerController);
//LOGIN || POST
router.post("/login", loginController);
//Forgot Password || POST
router.post("/forgot-password", forgotPasswordController);
//test routes
router.get("/test", requireSignIn, isAdmin, testController);
//protected User route auth
router.get("/user-auth", requireSignIn, (req, res) => {
res.status(200).send({ ok: true });
});
//protected Admin route auth
router.get("/admin-auth", requireSignIn, isAdmin, (req, res) => {
res.status(200).send({ ok: true }); });
```

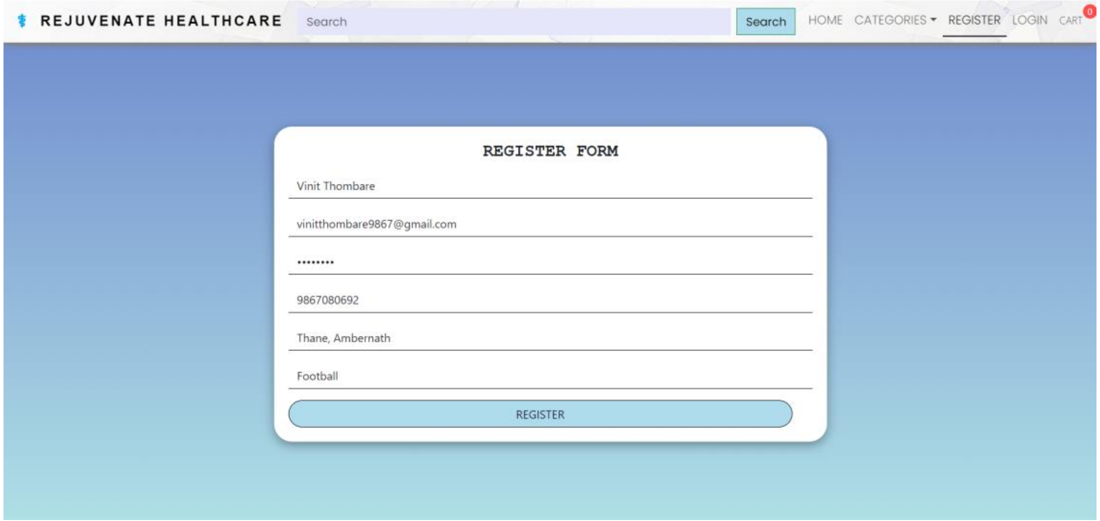
```
//update profile

router.put("/profile", requireSignIn, updateProfileController);

//orders

router.get("/orders", requireSignIn, getOrdersController);


//all orders

router.get("/all-orders", requireSignIn, isAdmin, getAllOrdersController);


// order status update

router.put(

"/order-status/:orderId", requireSignIn, isAdmin,orderStatusController);

export default router;
```

- **productRoutes.js**

```
import express from "express";
import formidable from "express-formidable";
import {
brainTreePaymentController, braintreeTokenController, createProductController,
deleteProductController, getProductController, getSingleProductController,
productCategoryController, productCountController, productFiltersController,
productListController, productPhotoController, realtedProductController,
searchProductController, updateProductController,} from "../controllers/productController.js";
import { isAdmin, requireSignIn } from "../middlewares/authMiddleware.js";
const router = express.Router();
//routes
router.post(
"/create-product", requireSignIn, isAdmin, formidable(), createProductController );
//routes
router.put(
"/update-product/:pid", requireSignIn, isAdmin, formidable(), updateProductController );

//get products
router.get("/get-product", getProductController);

//single product
```

```
router.get("/get-product/:slug", getSingleProductController);

//get photo
router.get("/product-photo/:pid", productPhotoController);

//delete rproduct
router.delete("/delete-product/:pid", deleteProductController);

//filter product
router.post("/product-filters", productFiltersController);

//product count
router.get("/product-count", productCountController);

//product per page
router.get("/product-list/:page", productListController);

//search product
router.get("/search/:keyword", searchProductController);

//similar product
router.get("/related-product/:pid/:cid", realtedProductController);

//category wise product
router.get("/product-category/:slug", productCategoryController);

//payments routes
//token
router.get("/braintree/token", braintreeTokenController);

//payments
router.post("/braintree/payment", requireSignIn, brainTreePaymentController);

export default router;
```

2. Data Entry Screens



REGISTER FORM

Vinit Thombare

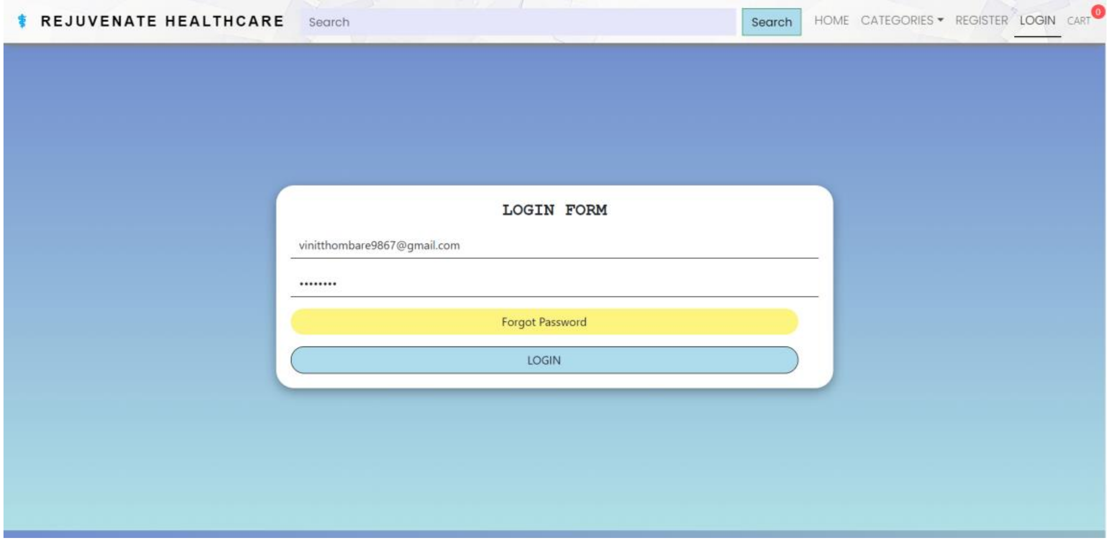
vinitthombare9867@gmail.com

••••••••

9867080692

Thane, Ambernath

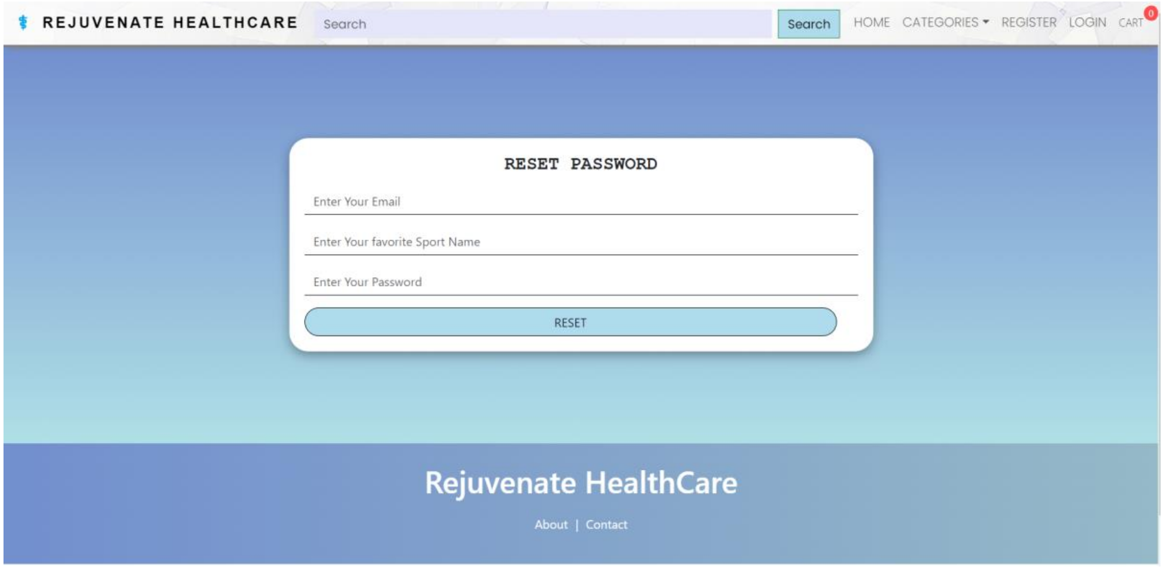
Football



LOGIN FORM

vinitthombare9867@gmail.com

••••••••



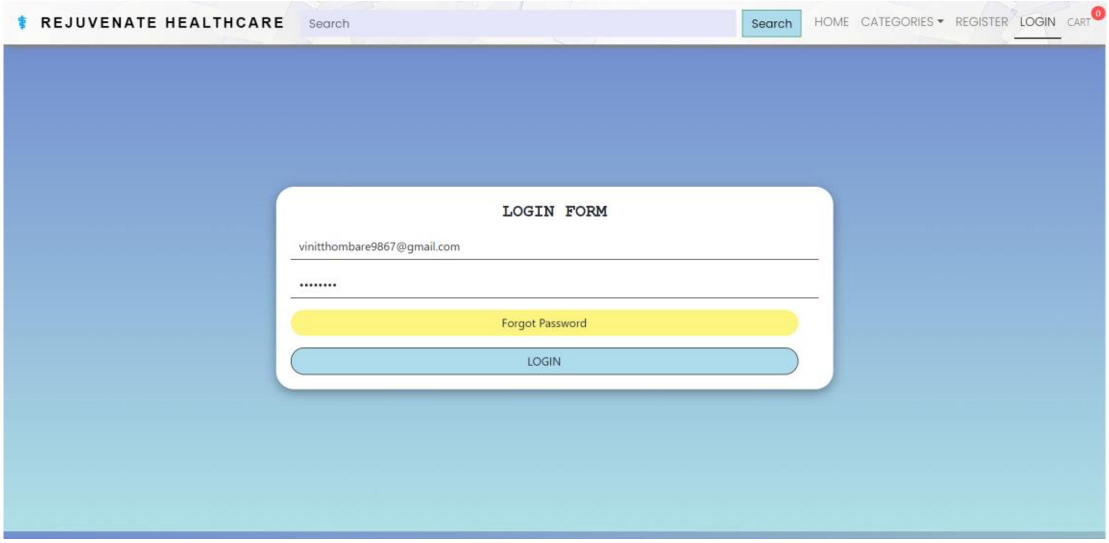
RESET PASSWORD

Enter Your Email

Enter Your favorite Sport Name

Enter Your Password

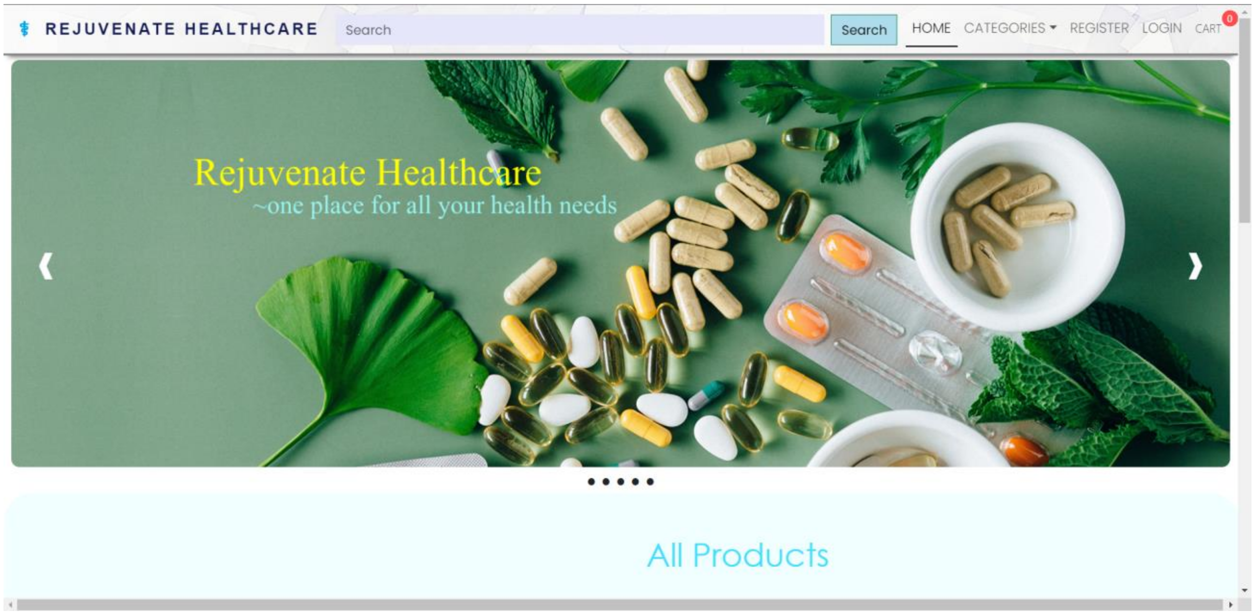
3. User input Pages

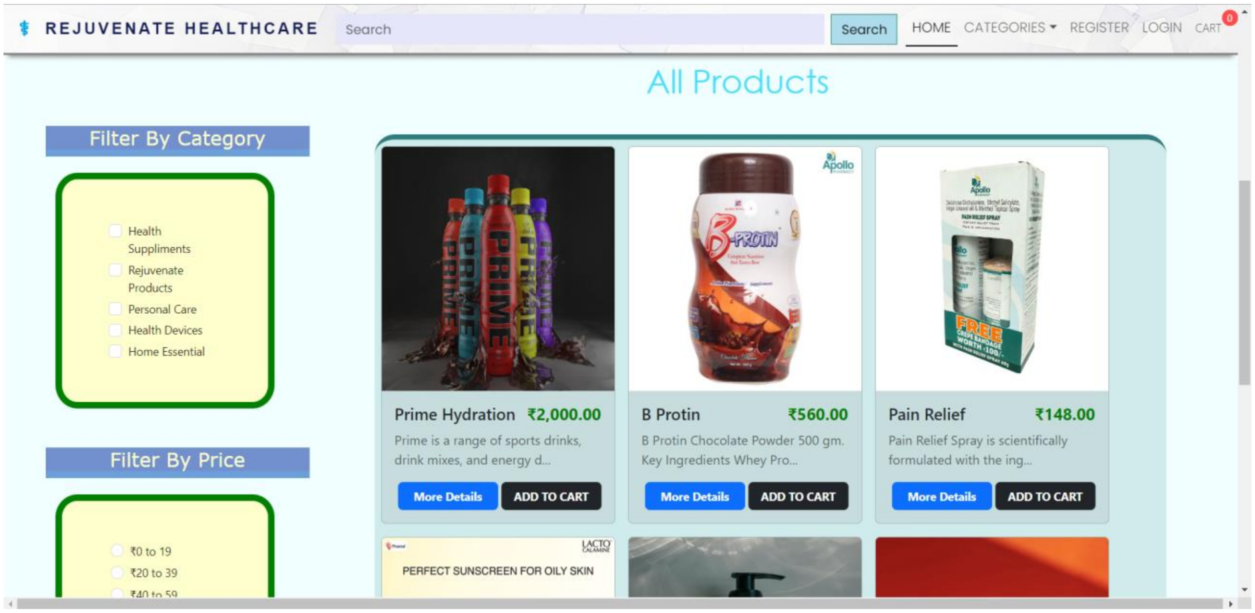


LOGIN FORM

vinitthombare9867@gmail.com

••••••••





All Products

Filter By Category

□ Health Suppliments
□ Rejuvenate Products
□ Personal Care
□ Health Devices
□ Home Essential

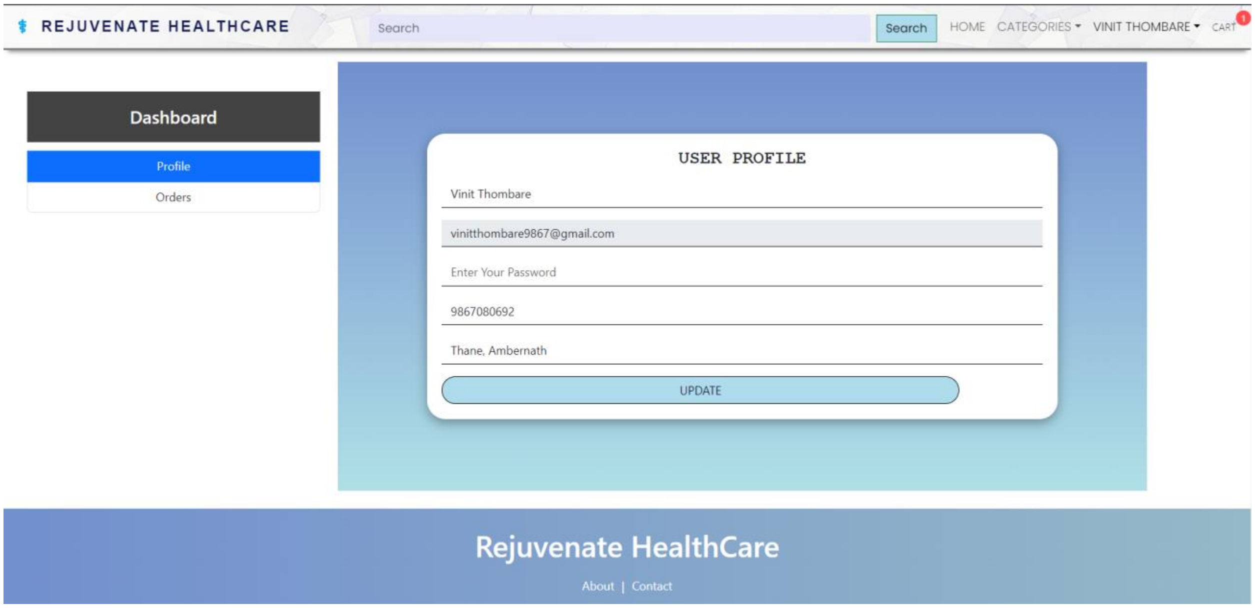
Filter By Price

○ ₹0 to 19
○ ₹20 to 39
○ ₹40 to 59

Prime Hydration ₹2,000.00
Prime is a range of sports drinks, drink mixes, and energy d...

B Protin ₹560.00
B Protin Chocolate Powder 500 gm. Key Ingredients Whey Pro...

Pain Relief ₹148.00
Pain Relief Spray is scientifically formulated with the ing...



USER PROFILE

Vinit Thombare

vinitthombare9867@gmail.com

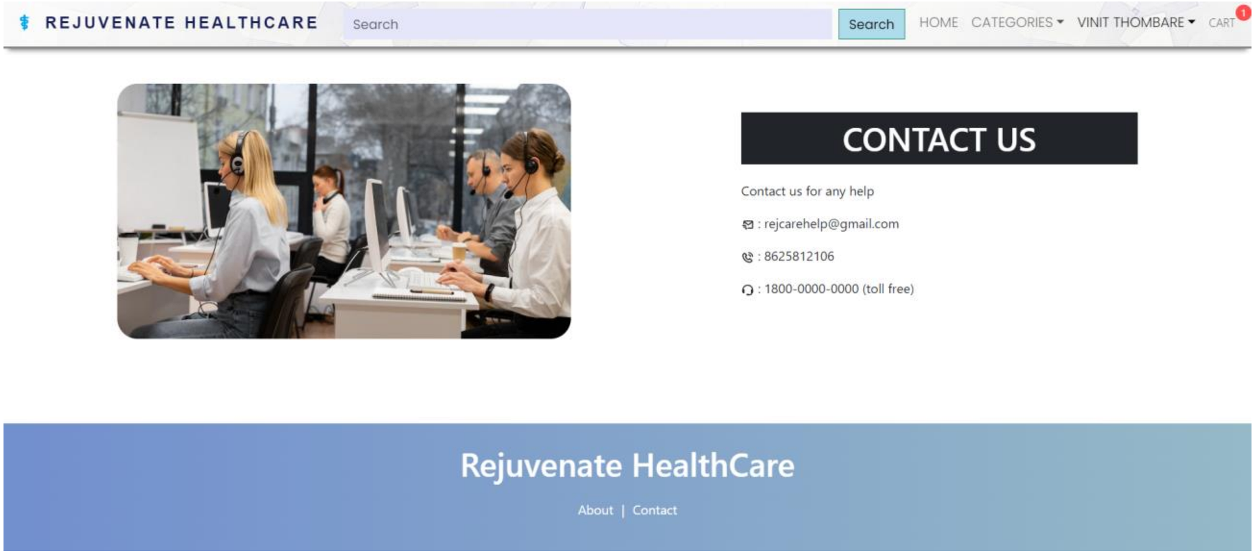
Enter Your Password

9867080692

Thane, Ambernath

Rejuvenate HealthCare

About | Contact



CONTACT US

Contact us for any help

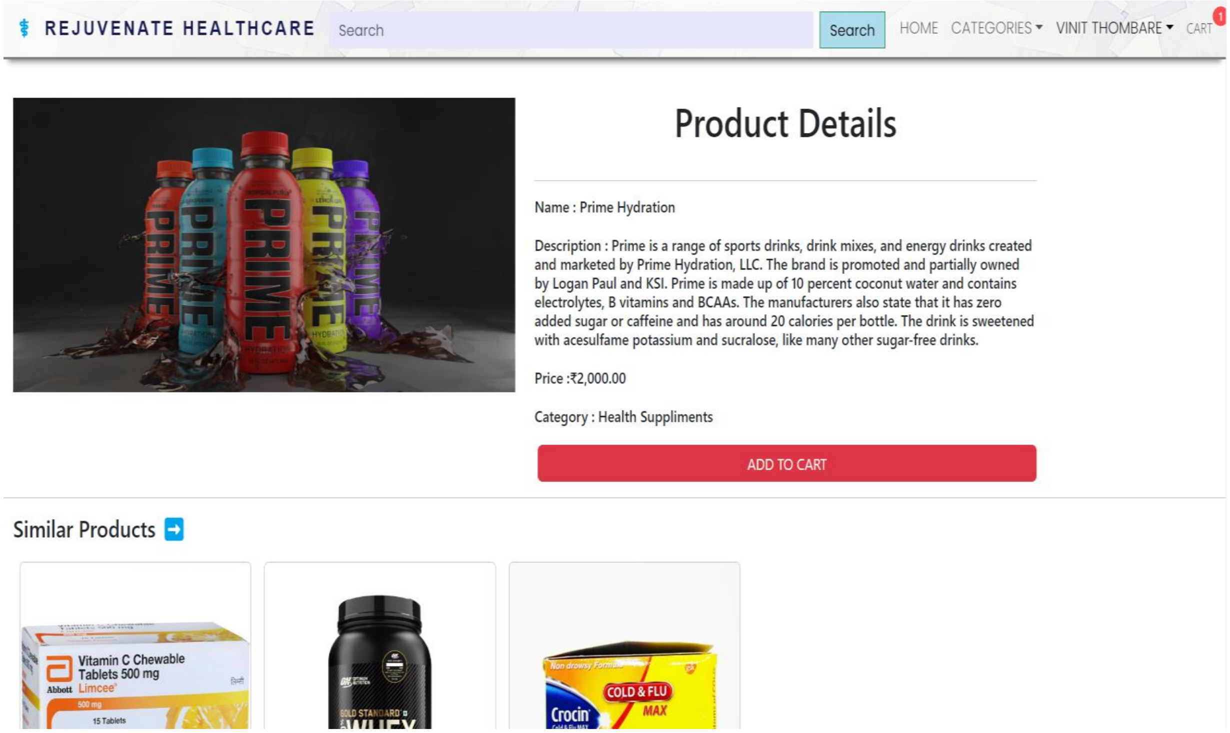
🖂 : rejcarehelp@gmail.com

☎ : 8625812106

🎧 : 1800-0000-0000 (toll free)

Rejuvenate HealthCare

About | Contact



Product Details

Name : Prime Hydration

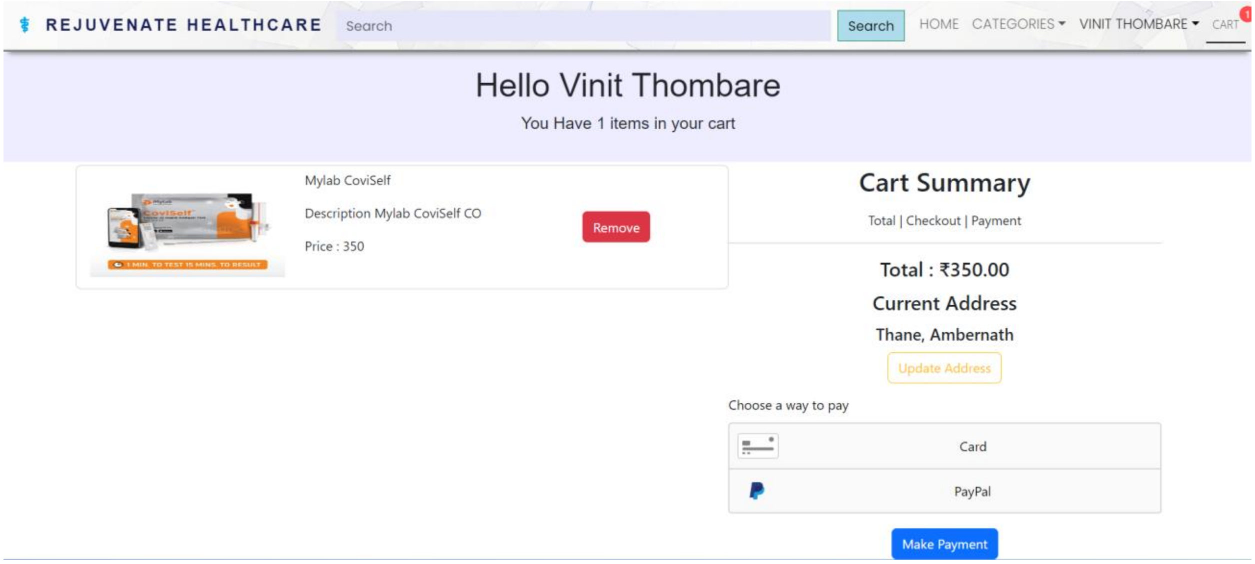
Description : Prime is a range of sports drinks, drink mixes, and energy drinks created and marketed by Prime Hydration, LLC. The brand is promoted and partially owned by Logan Paul and KSI. Prime is made up of 10 percent coconut water and contains electrolytes, B vitamins and BCAAs. The manufacturers also state that it has zero added sugar or caffeine and has around 20 calories per bottle. The drink is sweetened with acesulfame potassium and sucralose, like many other sugar-free drinks.

Price :₹2,000.00

Category : Health Suppliments

ADD TO CART

Similar Products



Hello Vinit Thombare

You Have 1 items in your cart

Mylab CoviSelf

Description Mylab CoviSelf CO

Price : 350

Cart Summary

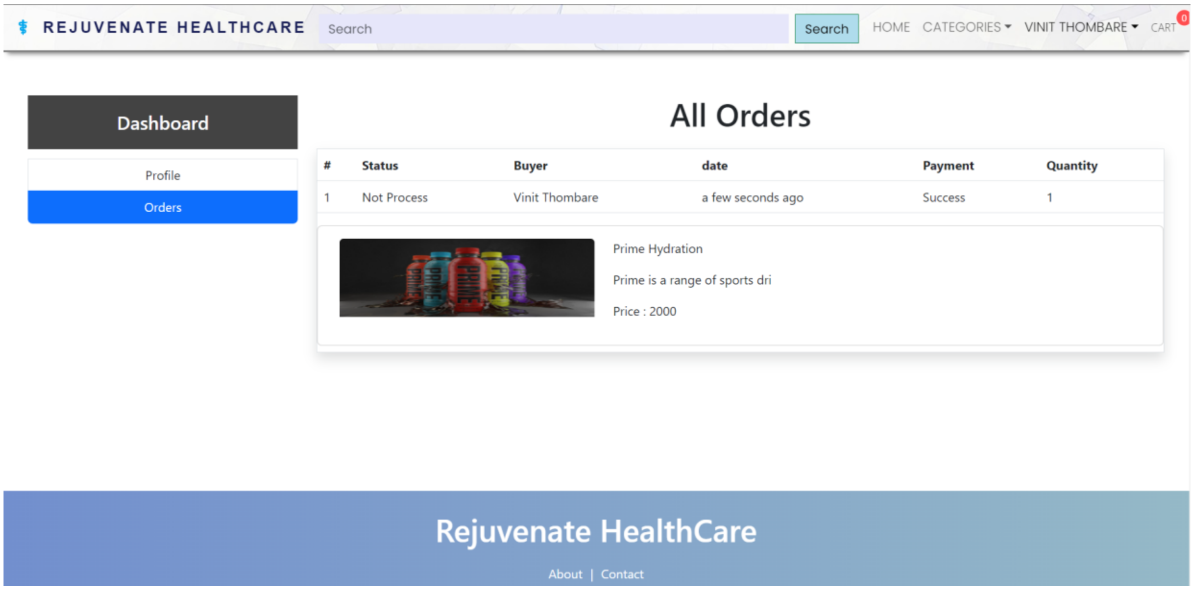
Total | Checkout | Payment

Total : ₹350.00

Current Address

Thane, Ambernath

Choose a way to pay



All Orders

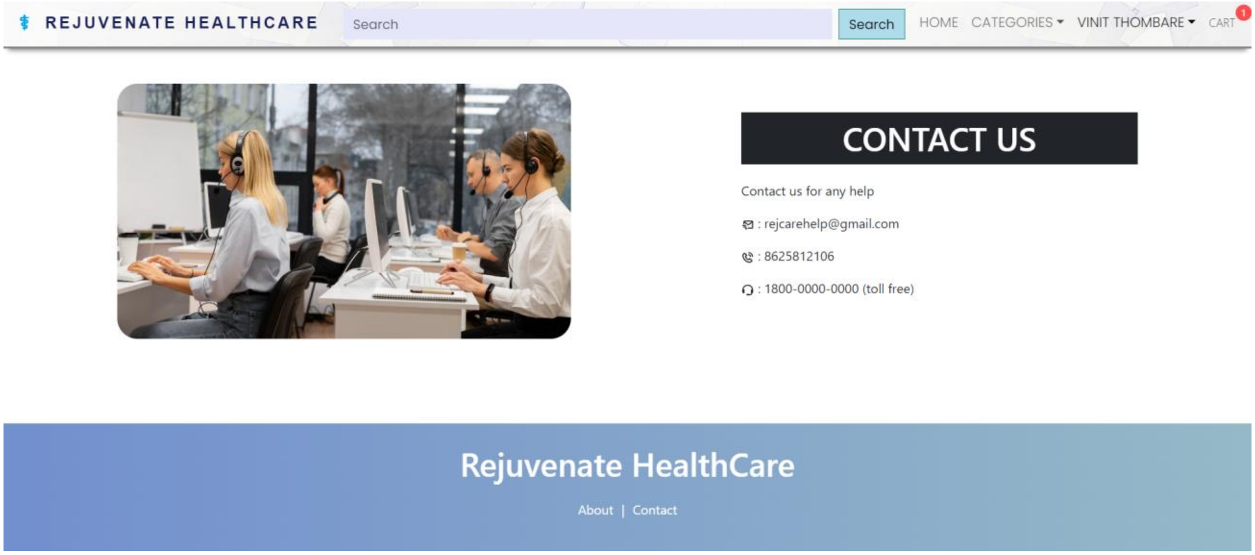
#	Status	Buyer	date	Payment	Quantity
1	Not Process	Vinit Thombare	a few seconds ago	Success	1

Prime Hydration

Prime is a range of sports dri

Price : 2000

Rejuvenate HealthCare



CONTACT US

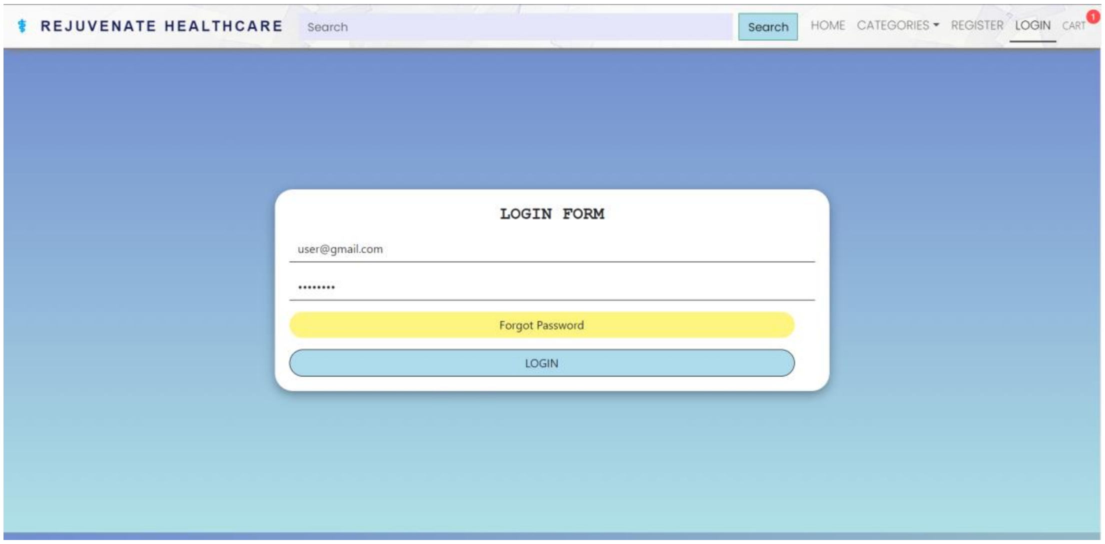
Contact us for any help

✉ : rejcarehelp@gmail.com

☎ : 8625812106

🎧 : 1800-0000-0000 (toll free)

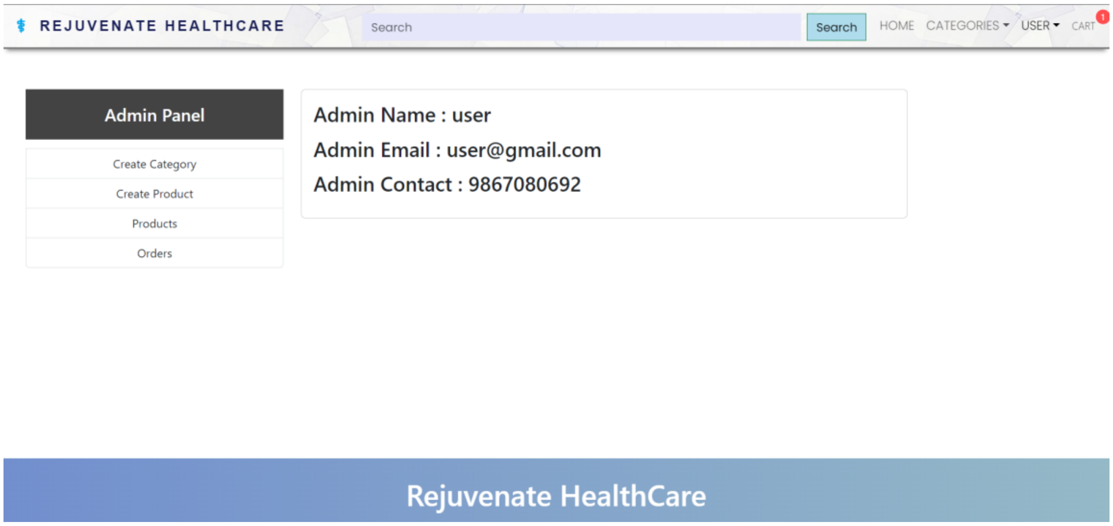
4. Admin Input Pages



LOGIN FORM

user@gmail.com

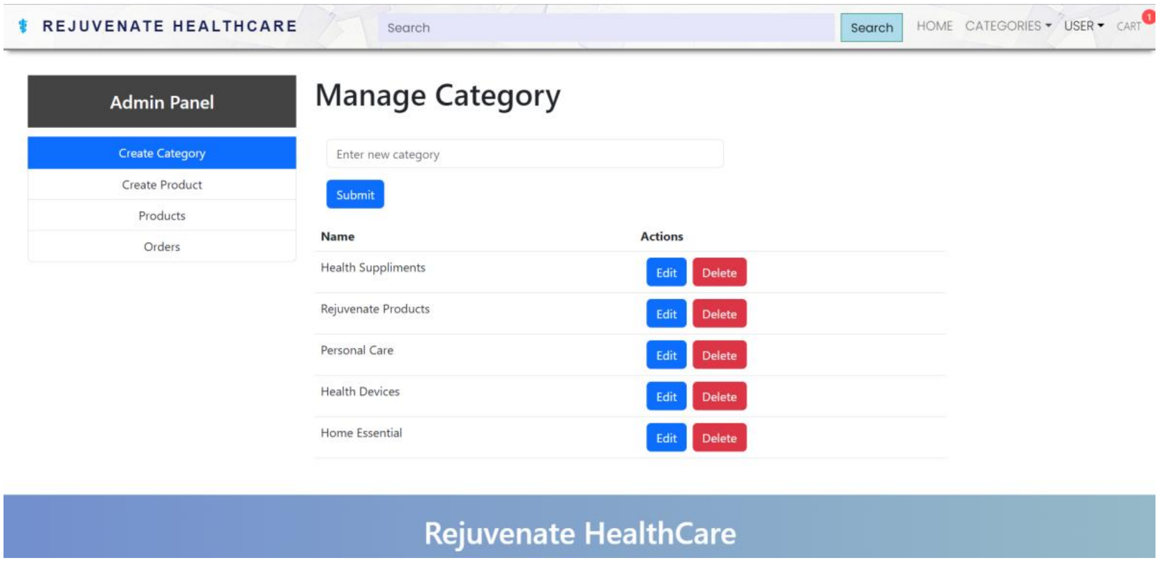
••••••••



Admin Name : user

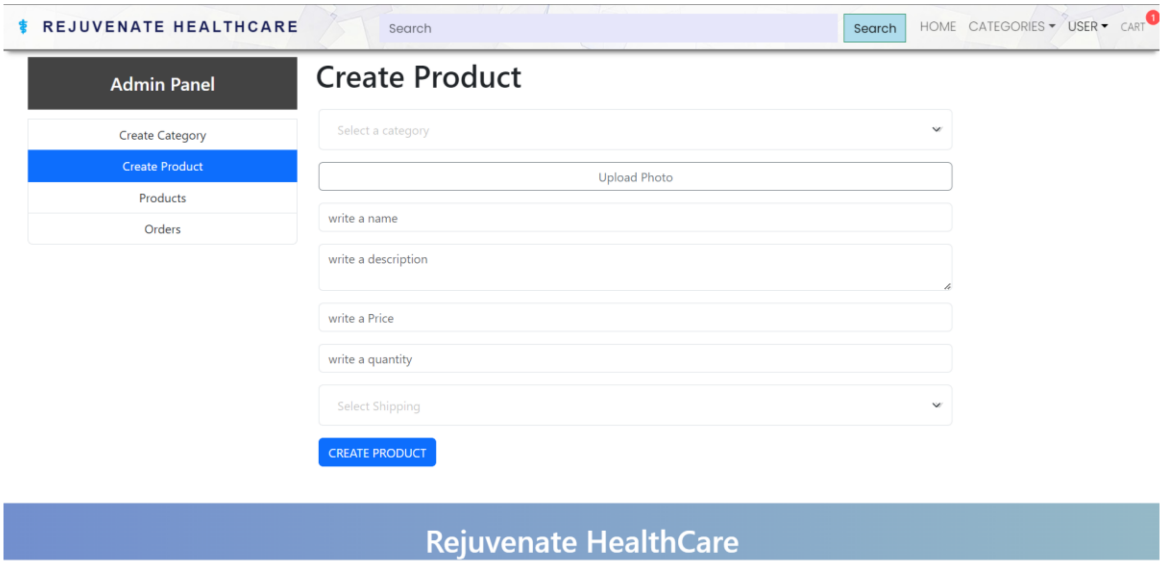
Admin Email : user@gmail.com

Admin Contact : 9867080692

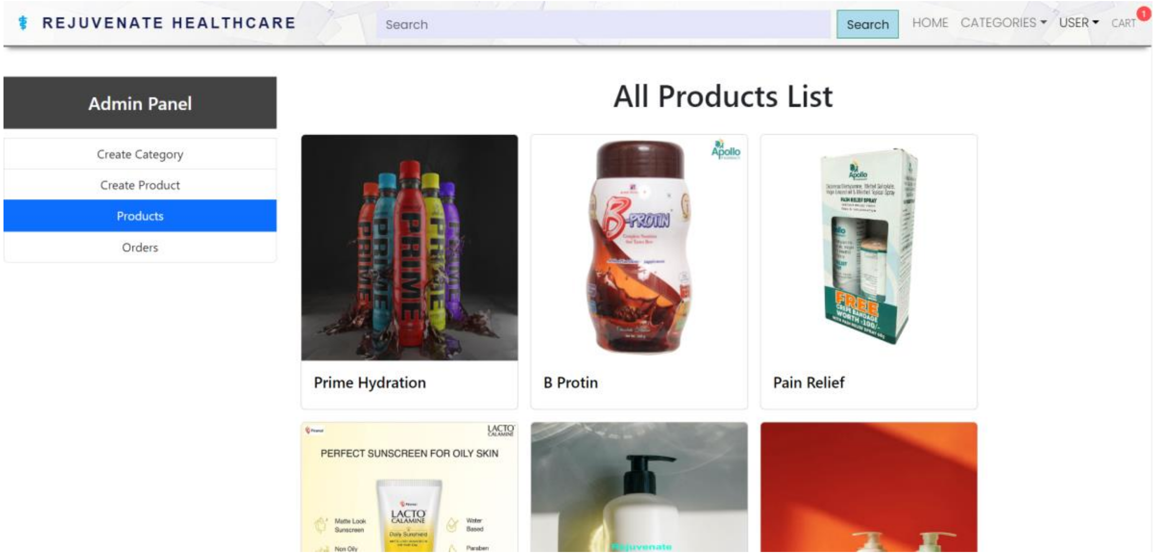


Manage Category

Name	Actions
Health Suppliments	Edit Delete
Rejuvenate Products	Edit Delete
Personal Care	Edit Delete
Health Devices	Edit Delete
Home Essential	Edit Delete



Create Product

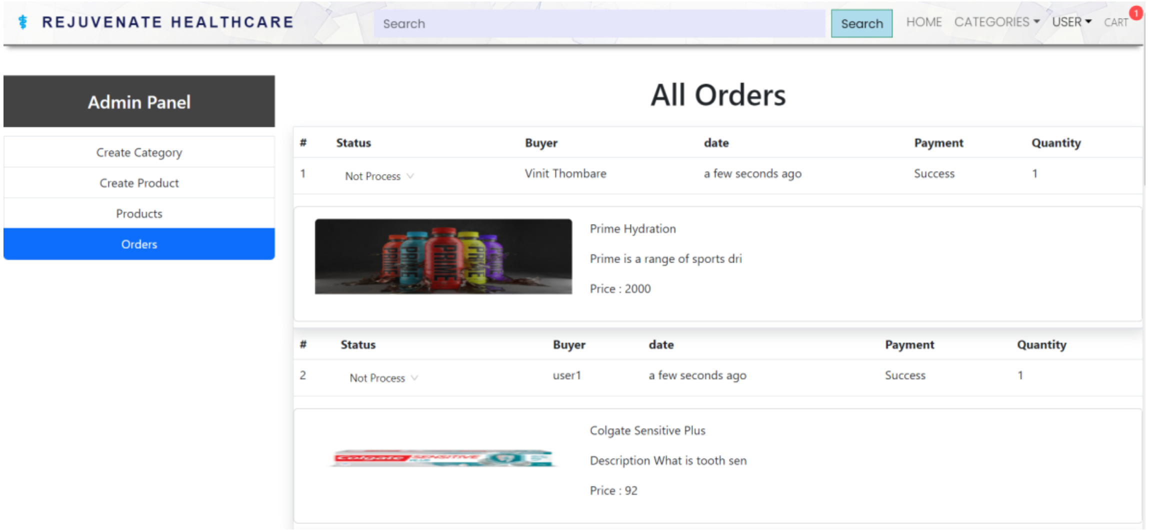


All Products List

Prime Hydration

B Protin

Pain Relief



All Orders

#	Status	Buyer	date	Payment	Quantity
1	Not Process	Vinit Thombare	a few seconds ago	Success	1

Prime Hydration

Prime is a range of sports dri

Price : 2000

#	Status	Buyer	date	Payment	Quantity
2	Not Process	user1	a few seconds ago	Success	1

Colgate Sensitive Plus

Description What is tooth sen

Price : 92

C) Validation

1. Test Cases

What is Test Case? “A Test Case has a component that describes an input, action or event expected response, to determine if a feature of an application is working correctly.” Software testing can be stated as the process of validating and verifying that a computer program/application/product:

- Meets the requirements that guided its design and development.
- Works as expected
- Can be implemented with the same characters.
- And satisfies the needs of Stakeholders.

Why we Write Test Case? A Test Case in Software Engineering is a set of conditions or variables under which a tester will determine whether an application, software system or one of it’s feature is working as it was originally established for it to do. The basic objective to write test case is to validate the testing coverage for the application. Test Cases bring some sort of standardization and minimize the ad-hoc approach in testing.

2. Test Data

List of variables and possible values used in the test case. You can list specific values or describe value ranges. The test case should be performed once for each combination Of values. These values written in set notation one per line. E.g.; Login ID {Valid login, invalid login, login empty}

1) Login ID {Valid login, invalid login, login empty}
2) Name {Name cannot be blank, Name should be a string}
3) Email {Invalid email id, Email id is a compulsory attribute}
4) Age {Age is a required attribute, Specify age with respect to range}
5) Validation Summary-Show pop ups at the top of the page **Steps to carry out the test.**

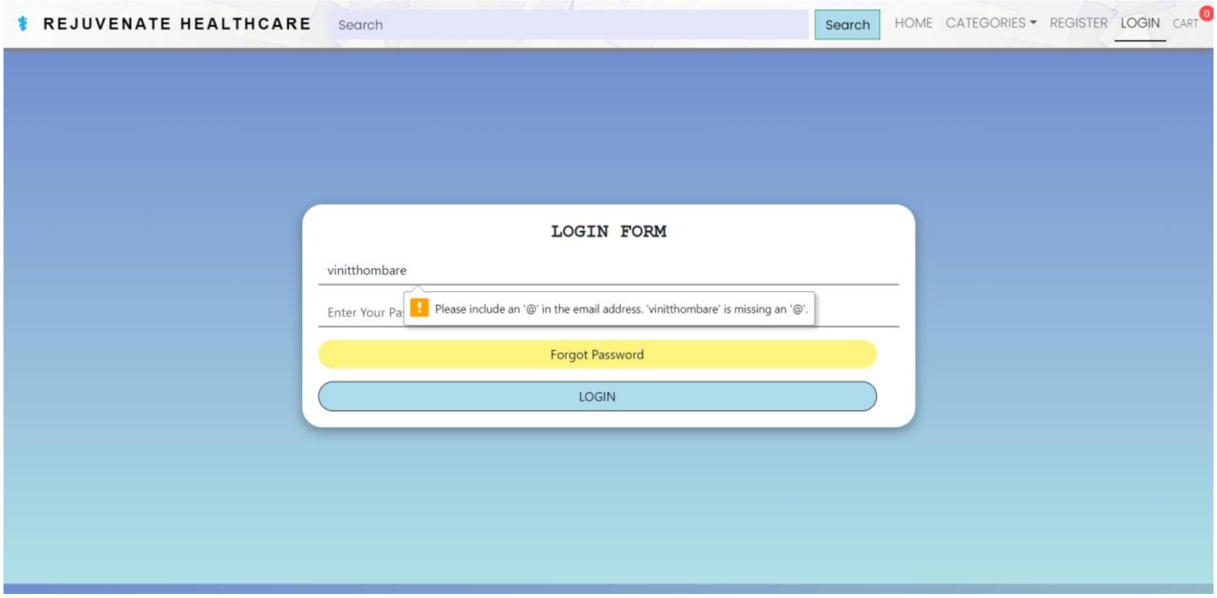
See step formatting rules below:

1. Visit Login page
2. Enter User Name
3. Enter Password Click Login **Actual Results:**

….Empty username and empty password Here you have to paste your empty user name and password field.. along with message box in alert type message box………. After than invalid username and password image along with msgbox…. (Total there should be 4 images)….

Comments:

If username and password are correctly entered only then you will be taken to next page.



LOGIN FORM

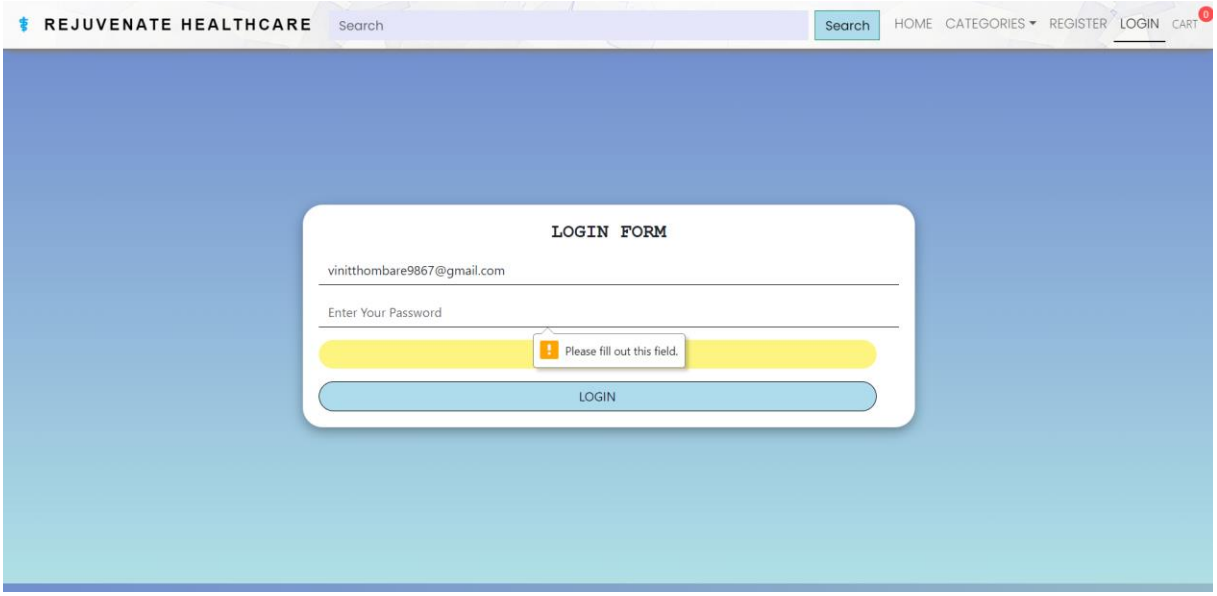
vinitthombare

Enter Your Pa

Please include an '@' in the email address. 'vinitthombare' is missing an '@'.

Forgot Password

LOGIN



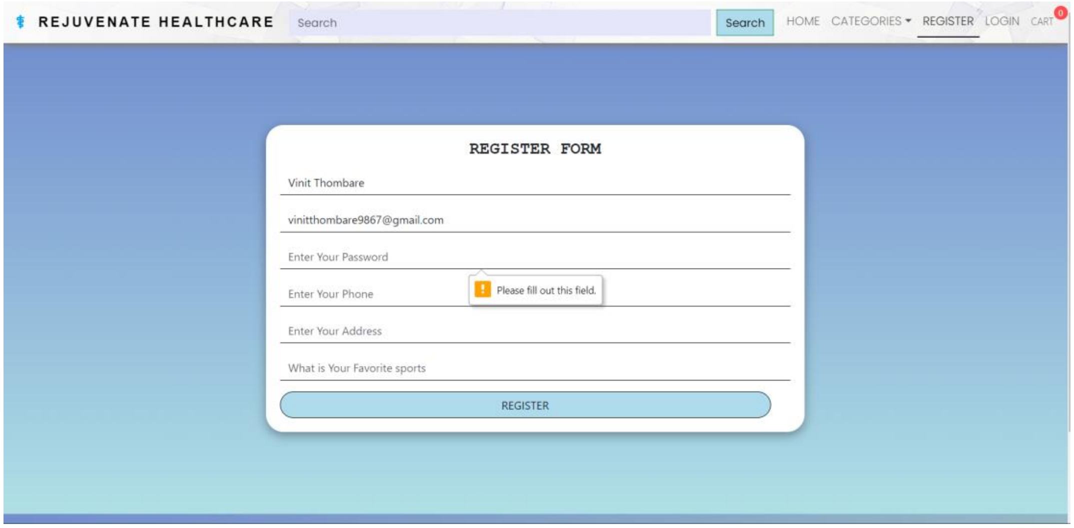
LOGIN FORM

vinitthombare9867@gmail.com

Enter Your Password

Please fill out this field.

LOGIN



REGISTER FORM

Vinit Thombare

vinitthombare9867@gmail.com

Enter Your Password

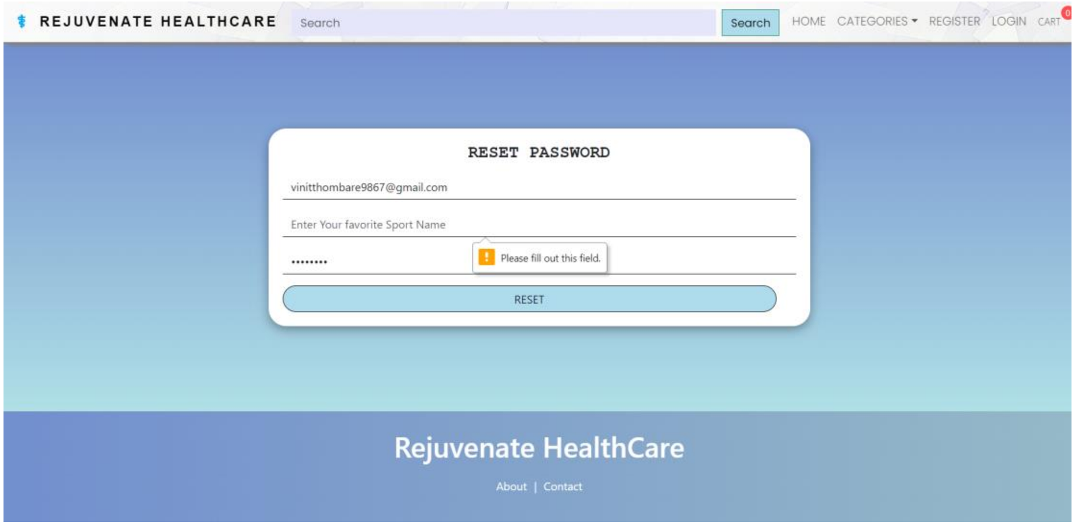
Enter Your Phone

Please fill out this field.

Enter Your Address

What is Your Favorite sports

REGISTER



RESET PASSWORD

vinitthombare9867@gmail.com

Enter Your favorite Sport Name

••••••••

Please fill out this field.

RESET

D) References and Bibliography

I. https://stackoverflow.com/

II. https://www.geeksforgeeks.org/reactjs-tutorials/

III. https://fullstackopen.com/en/

IV. https://jsfiddle.net/

V. https://reactrouter.com/en/main