A

PROJECT REPORT ON

**SHOPSY ECOMMERCE STORE**

DESIGNED AND DEVELOPED

BY

**Mr. MOHAMMAD RIYAZ SHAIKH**

AND

GUIDED BY

**Prof. Hasan Phudinawala**

SUBMITTED IN PARTIAL FULFILLMENT OF ACADEMIC PROJECT

BACHELOR OF SCIENCE [COMPUTER SCIENCE]

[UNIVERSITY OF MUMBAI]

**ACKNOWLEDGEMENT**

**ACKNOWLEDGEMENT**

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

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

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

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

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

**MOHAMMAD RIYAZ SHAIKH**

**DECLARATION**

**DECLARATION**

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

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

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

**MOHAMMAD RIYAZ SHAIKH**

**CERTIFICATE**



**Since 1989**

**DEPARTMENT OF COMPUTER SCIENCE**

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

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

**Certificate**

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

Signature of Project Guide

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

Signature of Examiner

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

H.O.D

Dept. of Computer Science

**ABOUT THE PROJECT**

**OBJECTIVES**

Reduce Management Cost

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

Providing a unique customer experience

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

Increasing the number of loyal customers

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

Boosting the efficiency of services

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

**INDEX**

**INDEX**

|  |  |  |
| --- | --- | --- |
| SR.  NO. | TOPIC | PAGE  NO. |
| I. | **Preliminary Investigation** |  |
| 1. | Present System and its advantage |  |
| 2. | System Requirements |  |
| 3. | Phase Title |  |
| 4. | Gantt Chart |  |
| II. | **System Analysis** |  |
| 1. | Event Table |  |
| 2. | Use Case Diagram |  |
| 3. | ERD |  |
| 4. | Class Diagram |  |
| III. | **System Design** |  |
| 1. | Sequence Diagram |  |
| IV. | **System Coding** |  |
| 1. | System Coding |  |
| V. | **Future Enhancement** |  |
| VI. | **Reference Requirement** |  |

**PRELIMINARY INVESTIGATION**

1. **PRELIMINARY INVESTIGATION**

Present System and advantages:

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

**System Requirements :**

**Hardware Requirements :**

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

**Software Requirements :**

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

**Node JS, Express JS, React-BootStrap5**

* **Framework : React JS**

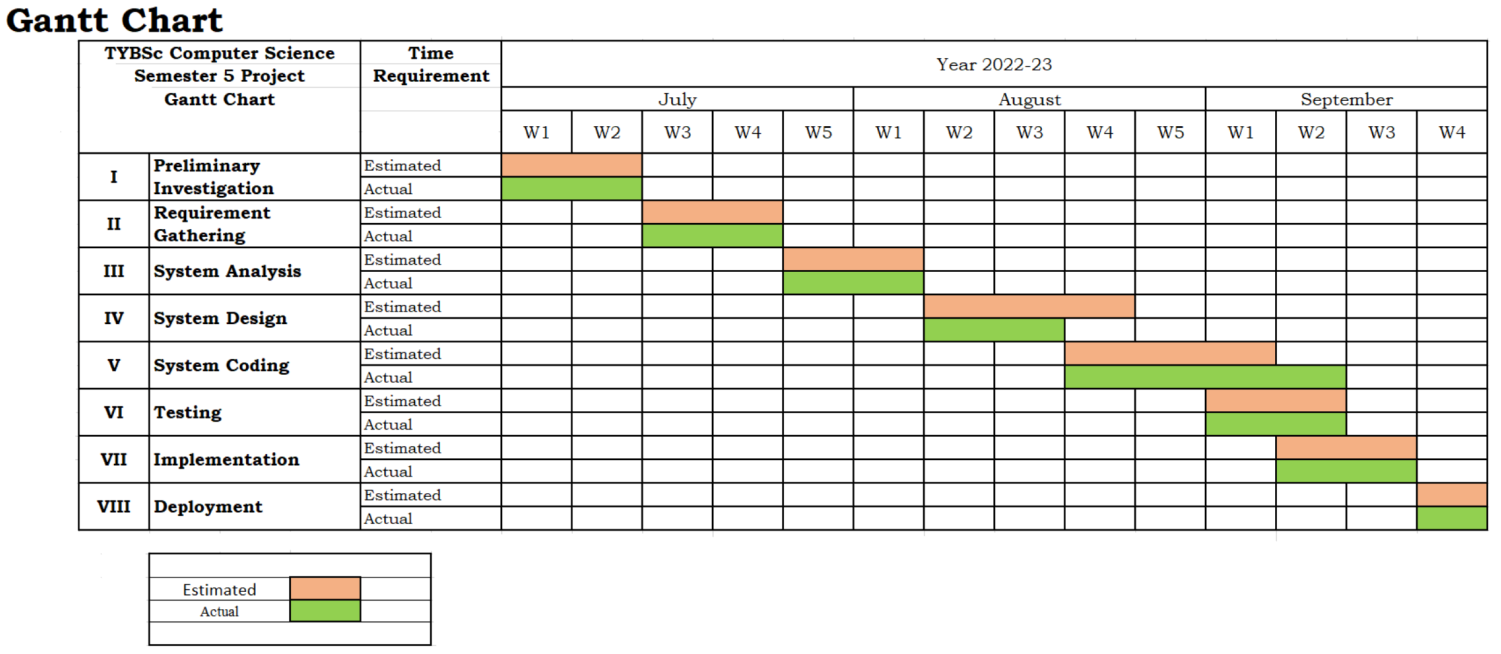
**PHASE TITLE:**

**TYBSC Computer Science Semester 5**

**2022 -2023**

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

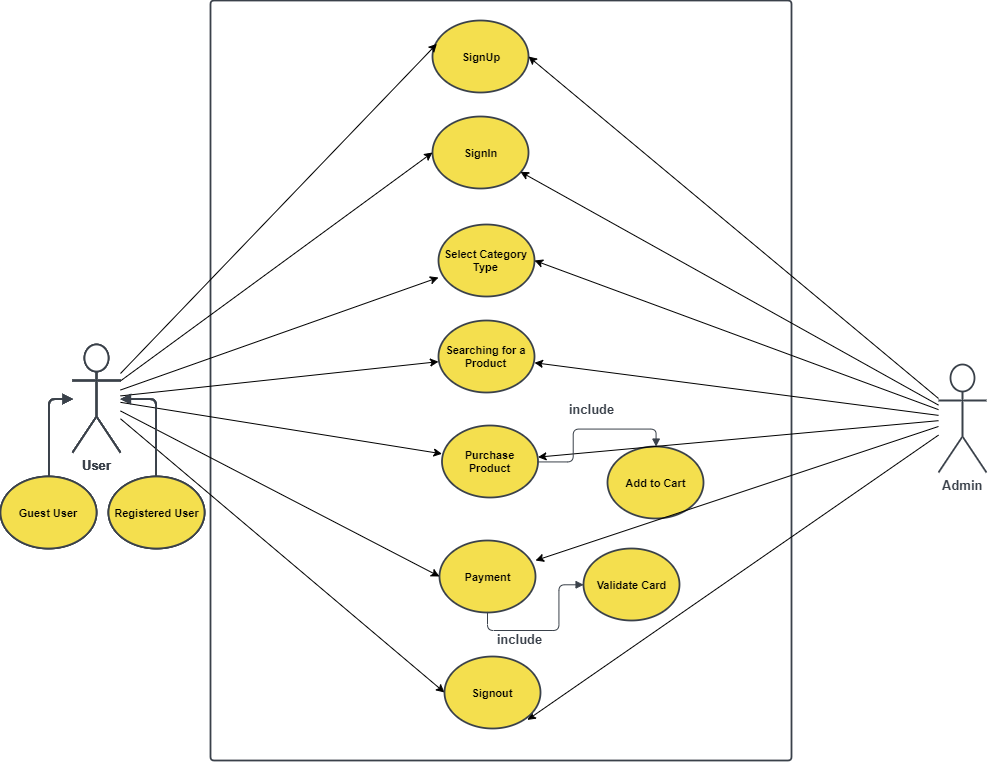
Student Name : MOHAMMAD RIYAZ SHAIKH Roll No : 41



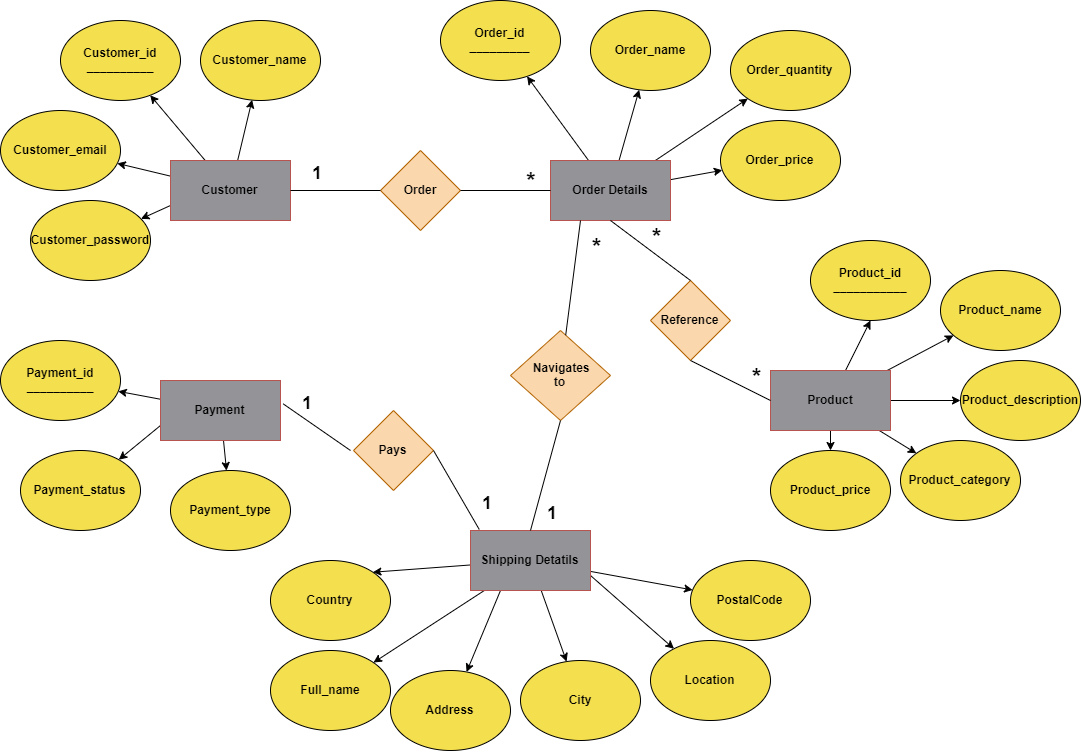
**SYSTEM ANALYSIS**

**Event Table**

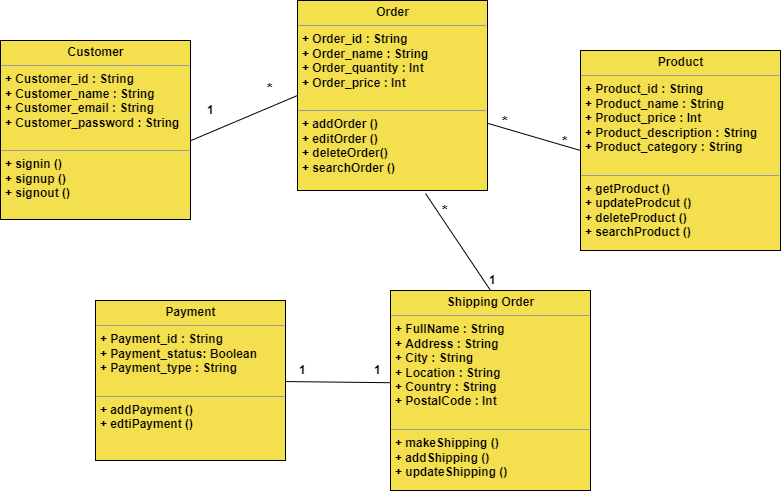
**USE CASE DIAGRAM**



**ENTITY RELATIONSHIP DIAGRAM**

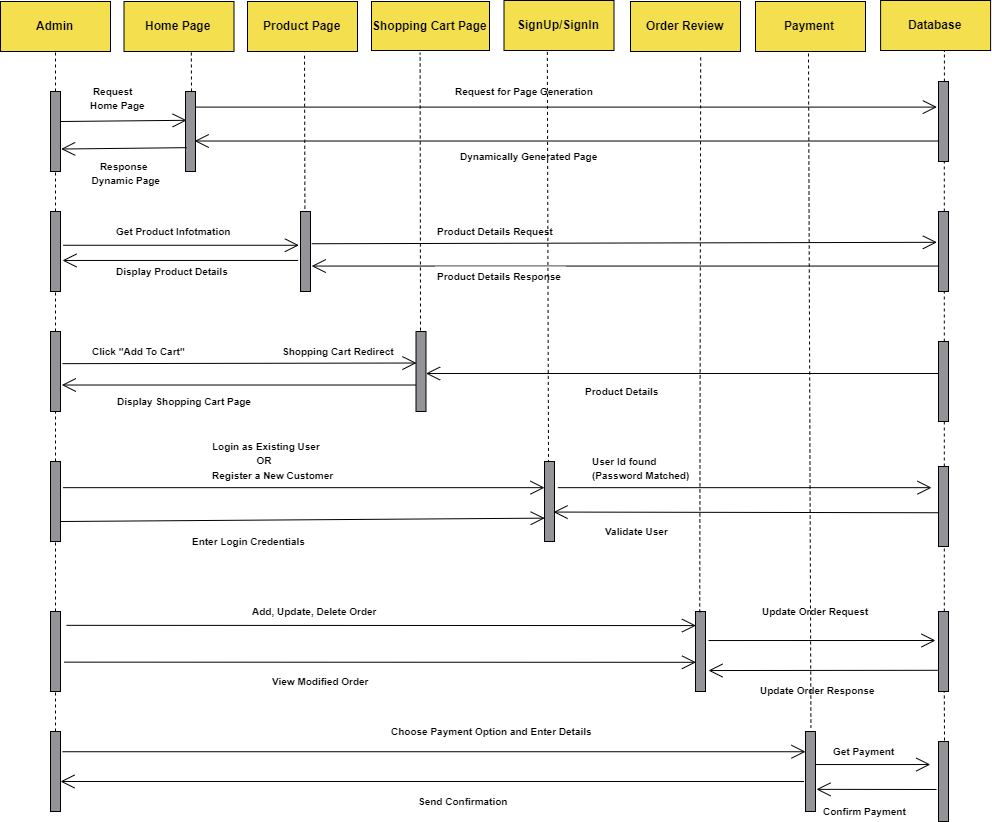


**CLASS DIAGRAM**

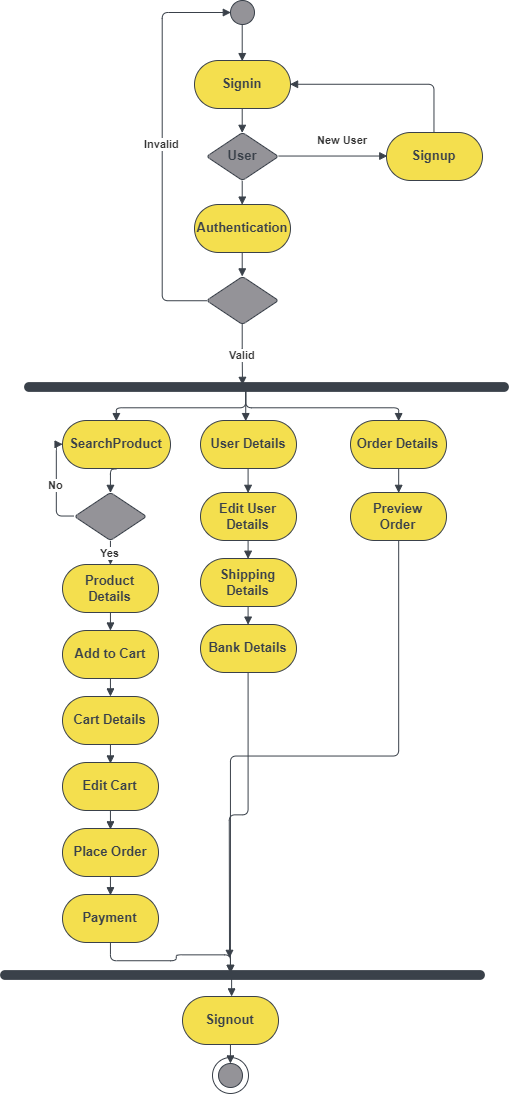


**SYSTEM DESIGN**

**SEQUENCE DIAGRAM**

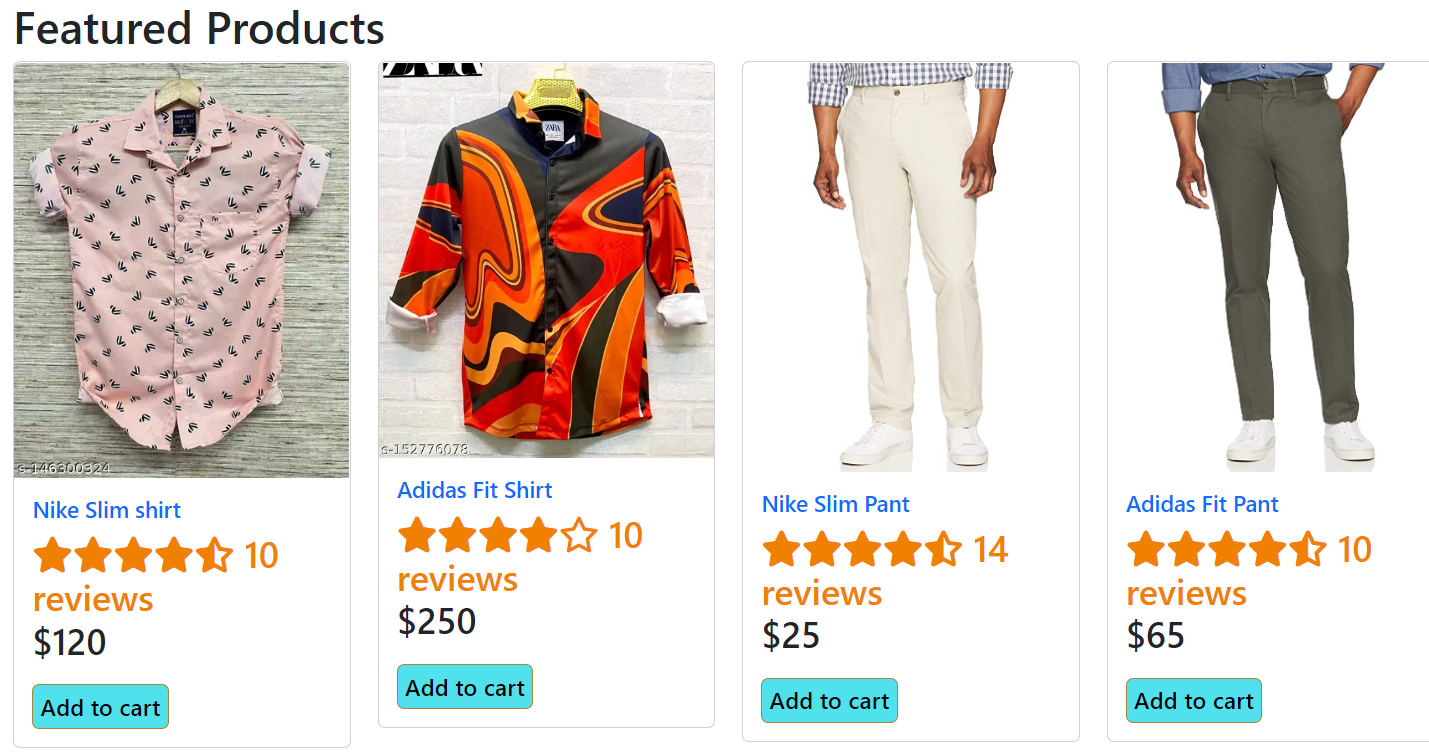
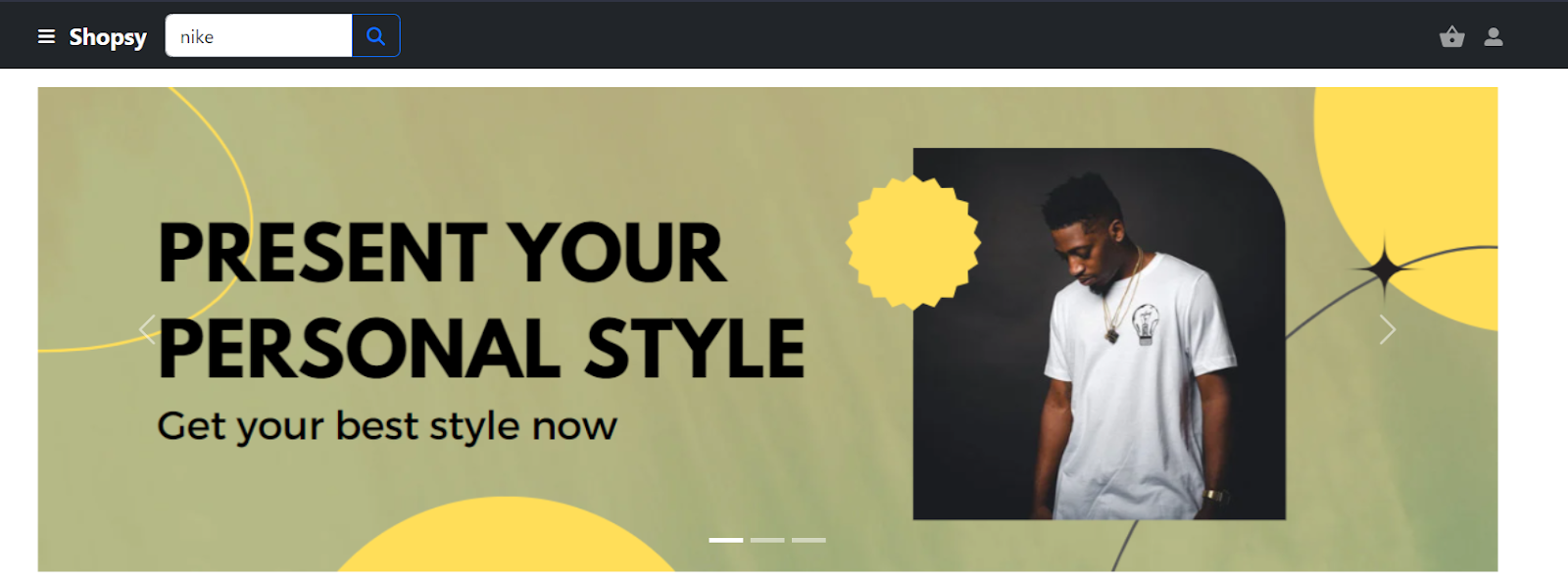


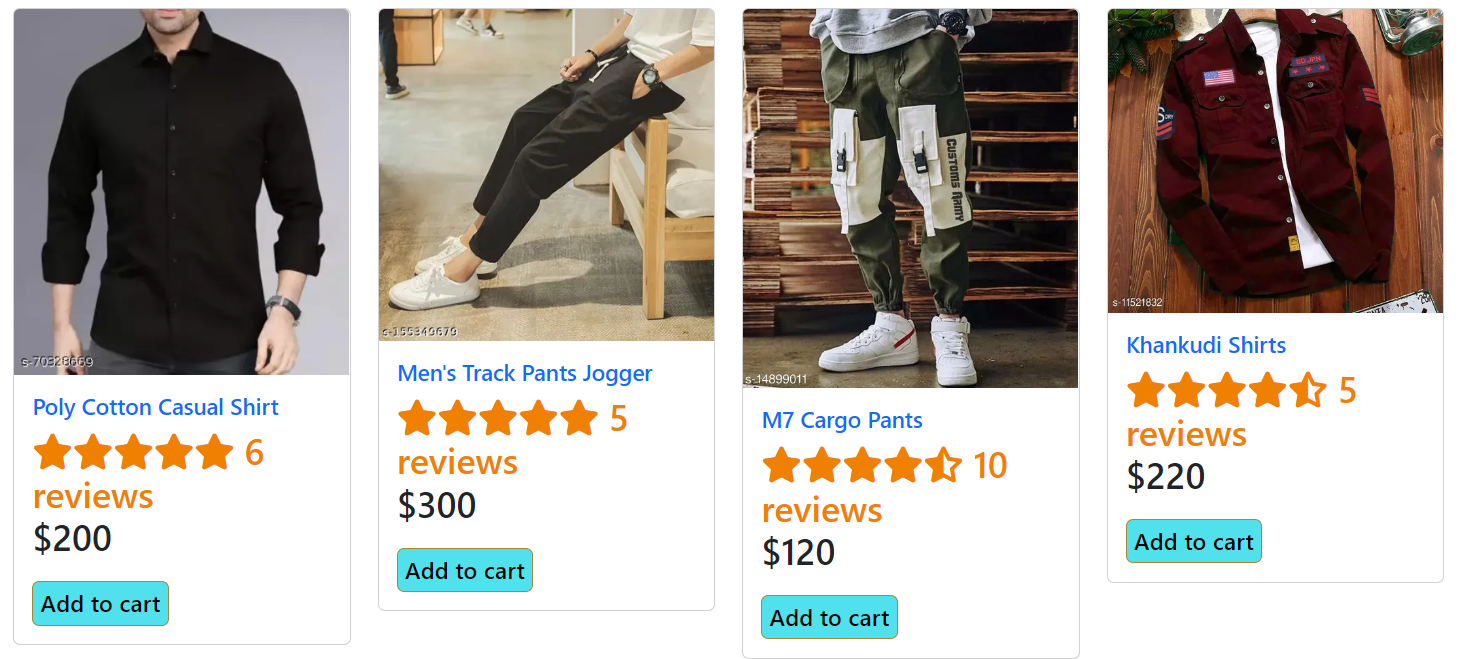
**ACTIVITY DIAGRAM**

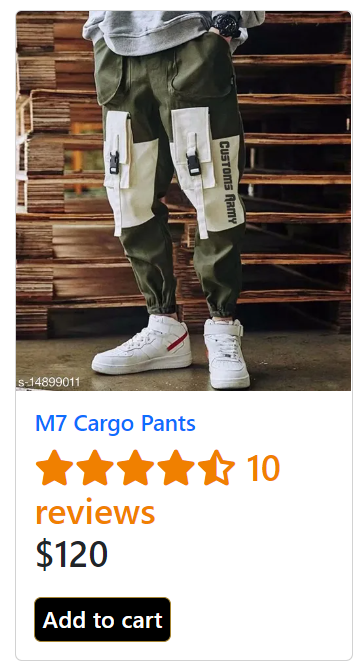


**SHOPSY ECOMMERCE STORE**

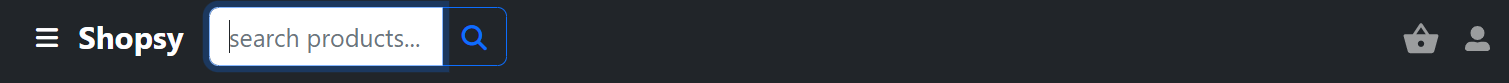
**HomeScreen**



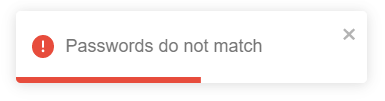
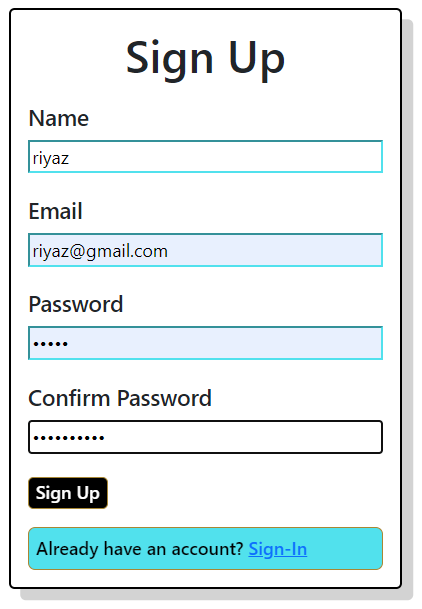




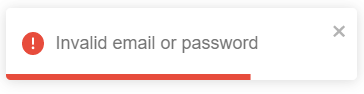
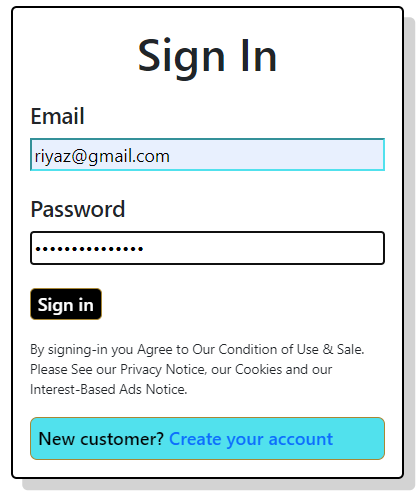
**CATEGORY**



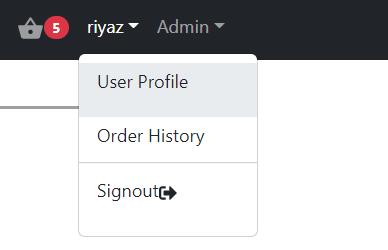
**SIGNUP**

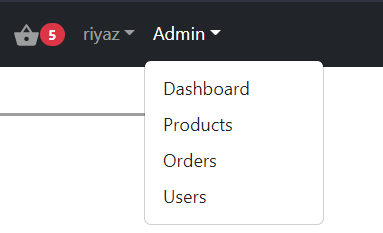


**SIGNIN**

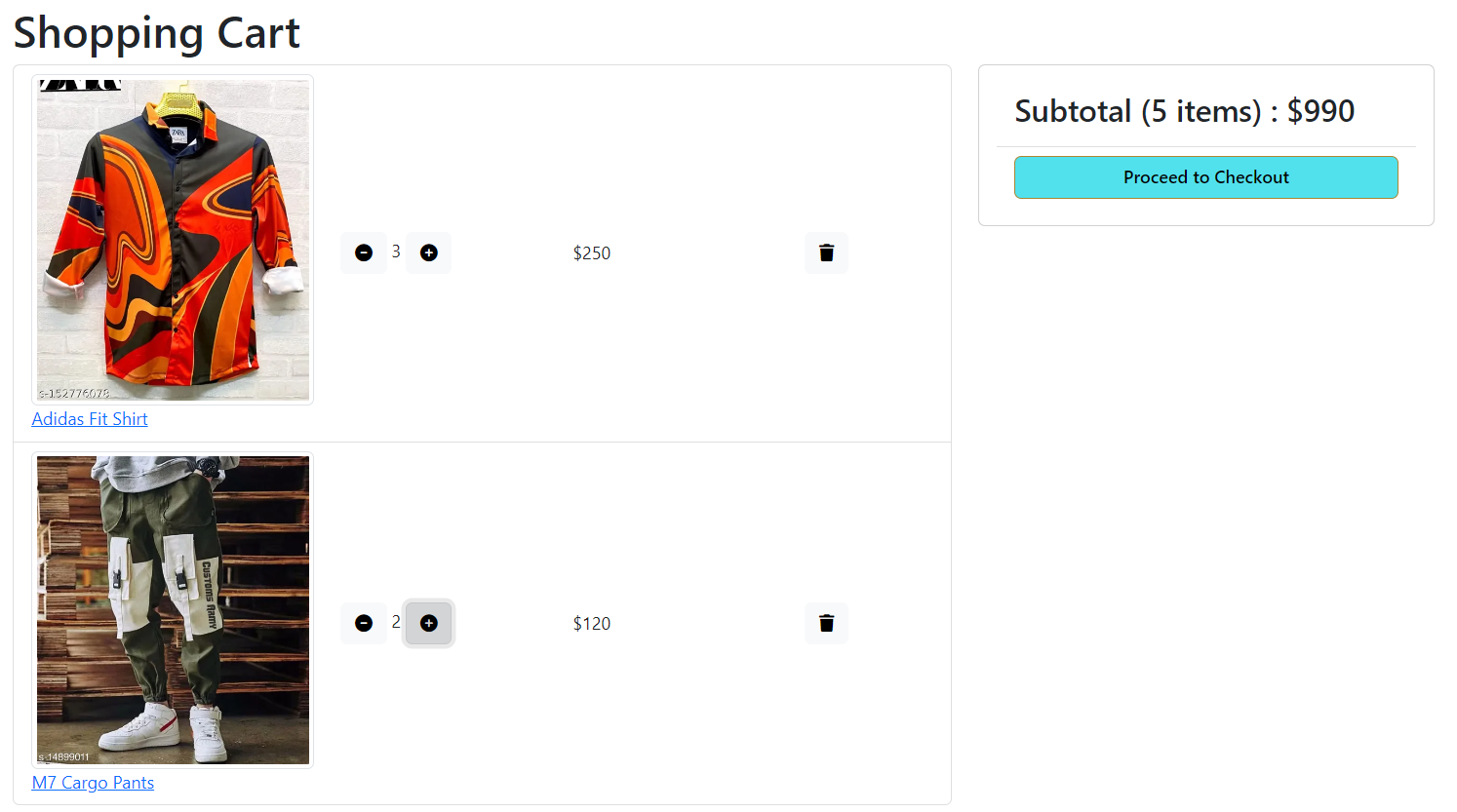


**NAVBAR**

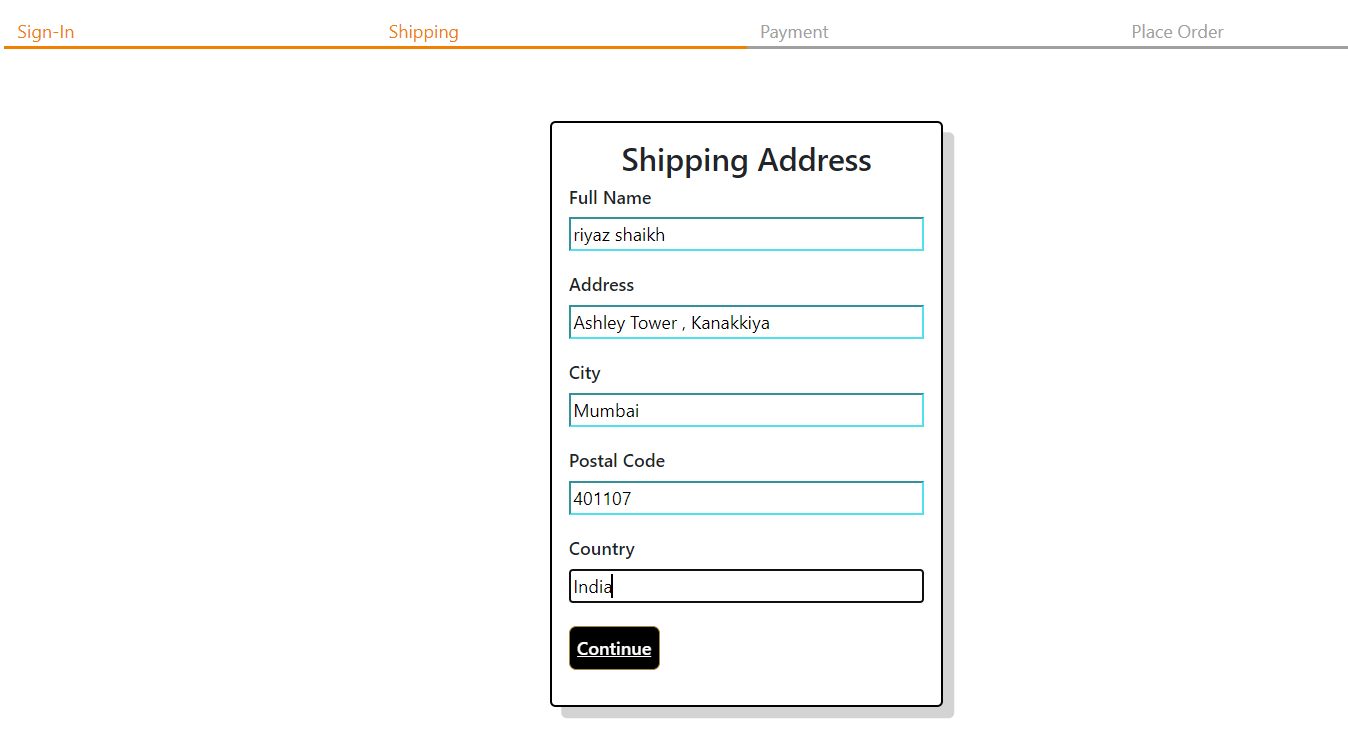




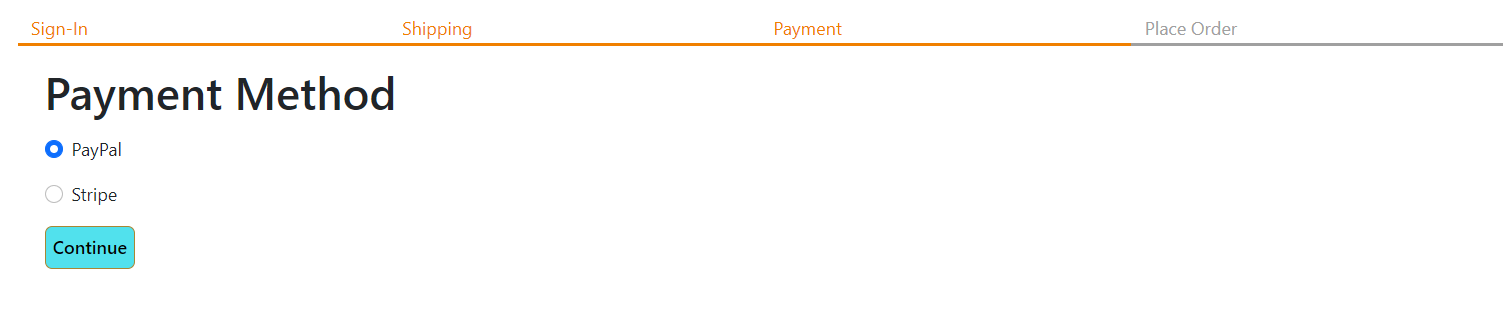
**SHOPPING CART**



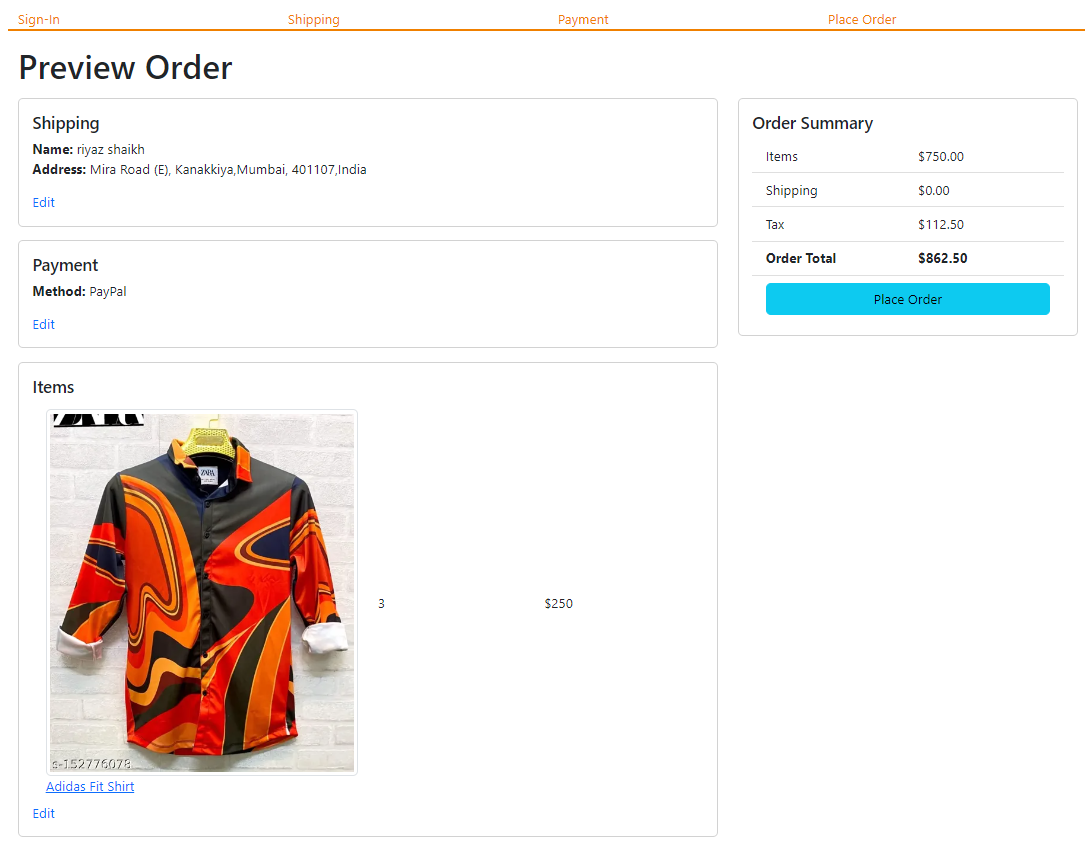
**SHOPPING ADDRESS**



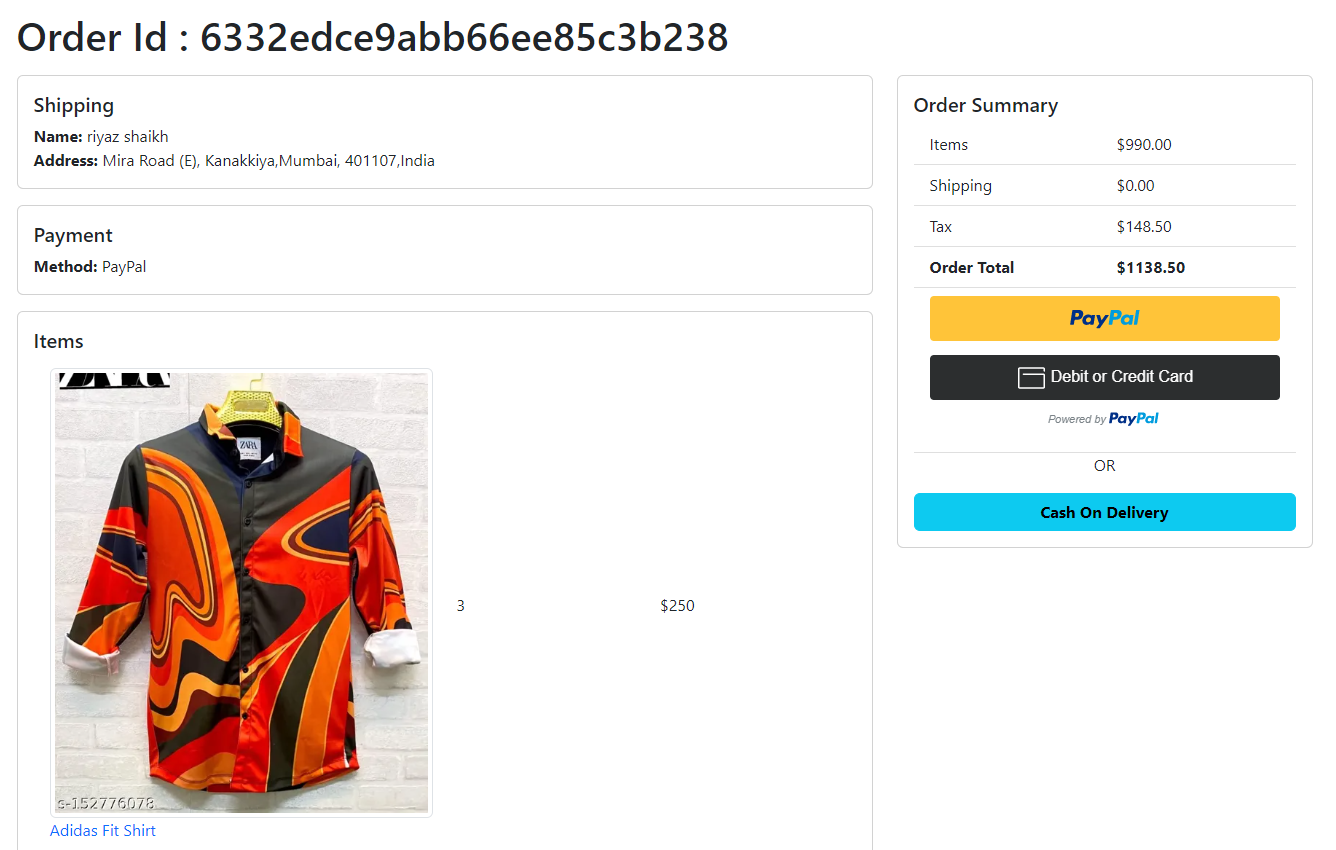
**PAYMENT METHOD**



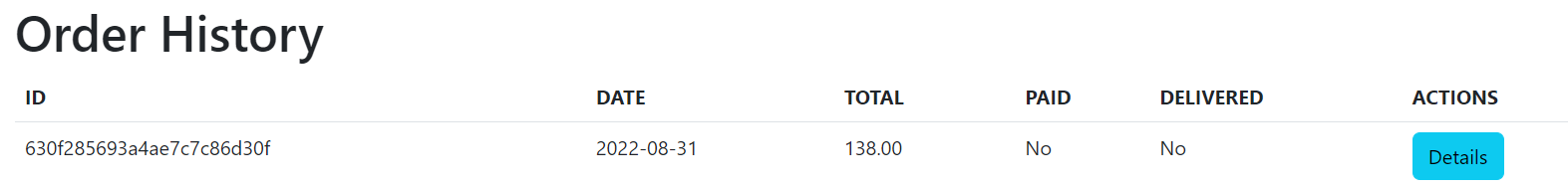
**PLACE ORDER**



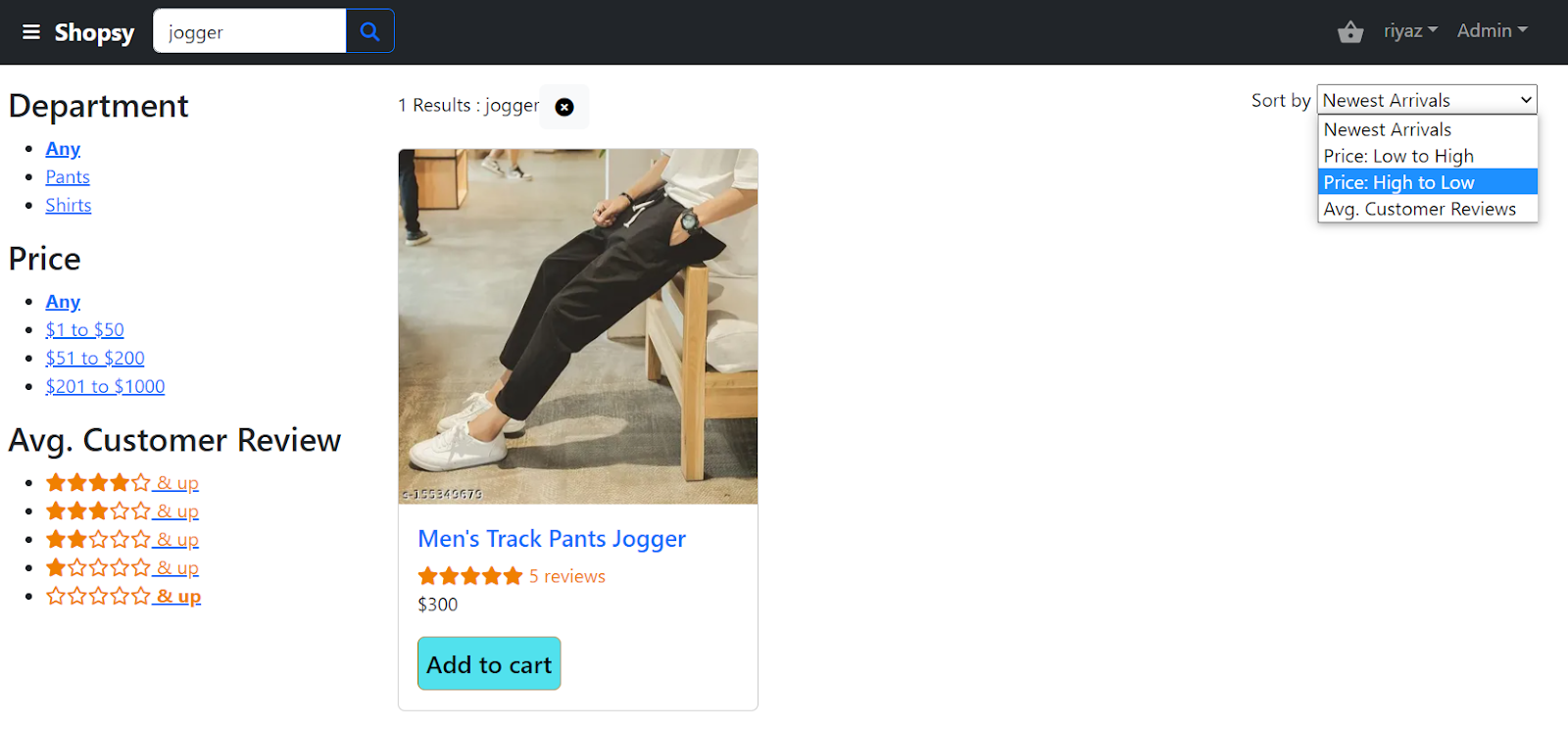
**ORDER SCREEN**



**ORDER HISTORY**



**SEARCH SCREEN**



**SYSTEM CODING**

**BACKEND**

**package.json**

{

"name": "backend",

"version": "1.0.0",

"description": "typroject backend",

"type": "module",

"main": "server.js",

"scripts": {

"start": "nodemon server.js --ignore client"

},

"author": "riyaz",

"license": "ISC",

"dependencies": {

"bcryptjs": "^2.4.3",

"cloudinary": "^1.28.1",

"dotenv": "^10.0.0",

"express": "^4.17.2",

"express-async-handler": "^1.2.0",

"jsonwebtoken": "^8.5.1",

"mailgun-js": "^0.22.0",

"mongoose": "^6.1.5",

"multer": "^1.4.4",

"streamifier": "^0.1.1"

},

"devDependencies": {

"nodemon": "^2.0.19"

}

}

**config.env**

PORT=4000

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

SECRET\_KEY=MERNSTACKDEVELOPMENT

PAYPAL\_CLIENT\_ID=AWRL7FNIrwylbEj03eZAblzwGPvKif5xnNo4OeTCQC4VFz60BoUN5Tm0h6jqUY5iJfLA5mOKDTRVglH6

**server.js**

**import** express from "express"**;**

**import** path from "path"**;**

**import** dotenv from "dotenv"**;**

**import** mongoose from "mongoose"**;**

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

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

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

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

dotenv**.**config**();**

**const** app **=** express**();**

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

mongoose

**.**connect**(**process**.**env**.**MONGODB\_URI**)**

**.**then**(()** **=>** **{**

console**.**log**(**"connected to db"**);**

**})**

**.catch((**err**)** **=>** **{**

console**.**log**(**err**.**message**);**

**});**

app**.**listen**(**port**,** **()** **=>** **{**

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

**});**

// Listen for body events

app**.**use**(**express**.**json**());**

app**.**use**(**express**.**json**());**

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

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

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

**});**

// routes

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

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

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

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

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

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

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

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

**});**

// error middleware

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

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

**});**

**Utils :**

**util.js**

**import** jwt from "jsonwebtoken"**;**

**export** **const** generateToken **=** **(**user**)** **=>** **{**

**return** jwt**.**sign**(**

**{**

\_id**:** user**.**\_id**,**

name**:** user**.**name**,**

email**:** user**.**email**,**

isAdmin**:** user**.**isAdmin**,**

**},**

process**.**env**.**SECRET\_KEY**,**

**{**

expiresIn**:** "30d"**,**

**}**

**);**

**};**

**export** **const** isAuth **=** **(**req**,** res**,** next**)** **=>** **{**

**const** authorization **=** req**.**headers**.**authorization**;**

**if** **(**authorization**)** **{**

**const** token **=** authorization**.**slice**(**7**,** authorization**.**length**);**

jwt**.**verify**(**token**,** process**.**env**.**SECRET\_KEY**,** **(**err**,** decode**)** **=>** **{**

**if** **(**err**)** **{**

res**.status(**401**).**send**({** message**:** "Invalid Token" **});**

**}** **else** **{**

// decrypting the token...

req**.**user **=** decode**;**

next**();**

**}**

**});**

**}** **else** **{**

res**.status(**401**).**send**({** message**:** "No Token" **});**

**}**

**};**

**export** **const** isAdmin **=** **(**req**,** res**,** next**)** **=>** **{**

**if** **(**req**.**user **&&** req**.**user**.**isAdmin**)** **{**

next**();**

**}** **else** **{**

res**.status(**401**).**send**({** message**:** "Invalid Admin Token" **});**

**}**

**};**

**Routes :**

**userRouter.js**

**import** express from "express"**;**

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

**import** expressAsyncHandler from "express-async-handler"**;**

**import** bcrypt from "bcryptjs"**;**

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

**const** userRouter **=** express**.**Router**();**

userRouter**.**post**(**

"/signin"**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

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

**if** **(**user**)** **{**

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

res**.**send**({**

\_id**:** user**.**\_id**,**

name**:** user**.**name**,**

email**:** user**.**email**,**

isAdmin**:** user**.**isAdmin**,**

token**:** generateToken**(**user**),**

**});**

**return;**

**}**

**}**

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

**})**

**);**

userRouter**.**post**(**

"/signup"**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

**const** newUser **=** **new** User**({**

name**:** req**.**body**.**name**,**

email**:** req**.**body**.**email**,**

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

**});**

**const** user **=** **await** newUser**.**save**();**

res**.**send**({**

\_id**:** user**.**\_id**,**

name**:** user**.**name**,**

email**:** user**.**email**,**

isAdmin**:** user**.**isAdmin**,**

token**:** generateToken**(**user**),**

**});**

**})**

**);**

userRouter**.**put**(**

"/profile"**,**

isAuth**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

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

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

**if** **(**user**)** **{**

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

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

**if** **(**req**.**body**.password)** **{**

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

**}**

**const** updatedUser **=** **await** user**.**save**();**

res**.**send**({**

\_id**:** updatedUser**.**\_id**,**

name**:** updatedUser**.**name**,**

email**:** updatedUser**.**email**,**

isAdmin**:** updatedUser**.**isAdmin**,**

token**:** generateToken**(**updatedUser**),**

**});**

**}** **else** **{**

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

**}**

**})**

**);**

**export** **default** userRouter**;**

**seedRouter.js**

**import** express from "express"**;**

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

**import** data from "../data.js"**;**

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

**const** seedRouter **=** express**.**Router**();**

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

**await** Product**.**deleteMany**({});**

**const** createdProducts **=** **await** Product**.**insertMany**(**data**.**products**);**

**await** User**.**remove**({});**

**const** createdUsers **=** **await** User**.**insertMany**(**data**.**users**);**

res**.**send**({** createdProducts**,** createdUsers **});**

**});**

**export** **default** seedRouter**;**

**productRouter.js**

**import** express from "express"**;**

**import** expressAsyncHandler from "express-async-handler"**;**

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

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

**const** productRouter **=** express**.**Router**();**

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

**const** products **=** **await** Product**.**find**();**

res**.**send**(**products**);**

**});**

productRouter**.**post**(**

"/"**,**

isAuth**,**

isAdmin**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

**const** newProduct **=** **new** Product**({**

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

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

**image:** "/images/p1.jpg"**,**

price**:** 0**,**

category**:** "sample category"**,**

brand**:** "sample brand"**,**

countInStock**:** 0**,**

rating**:** 0**,**

numReviews**:** 0**,**

description**:** "sample description"**,**

**});**

**const** product **=** **await** newProduct**.**save**();**

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

**})**

**);**

productRouter**.**put**(**

"/:id"**,**

isAuth**,**

isAdmin**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

**const** productId **=** req**.**params**.**id**;**

**const** product **=** **await** Product**.**findById**(**productId**);**

**if** **(**product**)** **{**

product**.**name **=** req**.**body**.**name**;**

product**.**slug **=** req**.**body**.**slug**;**

product**.**price **=** req**.**body**.**price**;**

product**.image** **=** req**.**body**.image;**

product**.images** **=** req**.**body**.images;**

product**.**category **=** req**.**body**.**category**;**

product**.**brand **=** req**.**body**.**brand**;**

product**.**countInStock **=** req**.**body**.**countInStock**;**

product**.**description **=** req**.**body**.**description**;**

**await** product**.**save**();**

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

**}** **else** **{**

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

**}**

**})**

**);**

productRouter**.delete(**

"/:id"**,**

isAuth**,**

isAdmin**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

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

**if** **(**product**)** **{**

**await** product**.**remove**();**

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

**}** **else** **{**

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

**}**

**})**

**);**

productRouter**.**post**(**

"/:id/reviews"**,**

isAuth**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

**const** productId **=** req**.**params**.**id**;**

**const** product **=** **await** Product**.**findById**(**productId**);**

**if** **(**product**)** **{**

**if** **(**product**.**reviews**.**find**((**x**)** **=>** x**.**name **===** req**.**user**.**name**))** **{**

**return** res

**.status(**400**)**

**.**send**({** message**:** "You already submitted a review" **});**

**}**

**const** review **=** **{**

name**:** req**.**user**.**name**,**

rating**:** Number**(**req**.**body**.**rating**),**

comment**:** req**.**body**.**comment**,**

**};**

product**.**reviews**.**push**(**review**);**

product**.**numReviews **=** product**.**reviews**.**length**;**

product**.**rating **=**

product**.**reviews**.**reduce**((**a**,** c**)** **=>** c**.**rating **+** a**,** 0**)** **/**

product**.**reviews**.**length**;**

**const** updatedProduct **=** **await** product**.**save**();**

res**.status(**201**).**send**({**

message**:** "Review Created"**,**

review**:** updatedProduct**.**reviews**[**updatedProduct**.**reviews**.**length **-** 1**],**

numReviews**:** product**.**numReviews**,**

rating**:** product**.**rating**,**

**});**

**}** **else** **{**

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

**}**

**})**

**);**

**const** PAGE\_SIZE **=** 3**;**

productRouter**.**get**(**

"/admin"**,**

isAuth**,**

isAdmin**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

**const** **{** query **}** **=** req**;**

**const** page **=** query**.**page **||** 1**;**

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

**const** products **=** **await** Product**.**find**()**

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

**.**limit**(**pageSize**);**

**const** countProducts **=** **await** Product**.**countDocuments**();**

res**.**send**({**

products**,**

countProducts**,**

page**,**

pages**:** Math**.**ceil**(**countProducts **/** pageSize**),**

**});**

**})**

**);**

productRouter**.**get**(**

"/search"**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

**const** **{** query **}** **=** req**;**

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

**const** page **=** query**.**page **||** 1**;**

**const** category **=** query**.**category **||** ""**;**

**const** price **=** query**.**price **||** ""**;**

**const** rating **=** query**.**rating **||** ""**;**

**const** order **=** query**.**order **||** ""**;**

**const** searchQuery **=** query**.**query **||** ""**;**

**const** queryFilter **=**

searchQuery **&&** searchQuery **!==** "all"

**?** **{**

name**:** **{**

$regex**:** searchQuery**,**

$options**:** "i"**,**

**},**

**}**

**:** **{};**

**const** categoryFilter **=** category **&&** category **!==** "all" **?** **{** category **}** **:** **{};**

**const** ratingFilter **=**

rating **&&** rating **!==** "all"

**?** **{**

rating**:** **{**

$gte**:** Number**(**rating**),**

**},**

**}**

**:** **{};**

**const** priceFilter **=**

price **&&** price **!==** "all"

**?** **{**

// 1-50

price**:** **{**

$gte**:** Number**(**price**.**split**(**"-"**)[**0**]),**

$lte**:** Number**(**price**.**split**(**"-"**)[**1**]),**

**},**

**}**

**:** **{};**

**const** sortOrder **=**

order **===** "featured"

**?** **{** featured**:** **-**1 **}**

**:** order **===** "lowest"

**?** **{** price**:** 1 **}**

**:** order **===** "highest"

**?** **{** price**:** **-**1 **}**

**:** order **===** "toprated"

**?** **{** rating**:** **-**1 **}**

**:** order **===** "newest"

**?** **{** createdAt**:** **-**1 **}**

**:** **{** \_id**:** **-**1 **};**

**const** products **=** **await** Product**.**find**({**

**...**queryFilter**,**

**...**categoryFilter**,**

**...**priceFilter**,**

**...**ratingFilter**,**

**})**

**.**sort**(**sortOrder**)**

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

**.**limit**(**pageSize**);**

**const** countProducts **=** **await** Product**.**countDocuments**({**

**...**queryFilter**,**

**...**categoryFilter**,**

**...**priceFilter**,**

**...**ratingFilter**,**

**});**

res**.**send**({**

products**,**

countProducts**,**

page**,**

pages**:** Math**.**ceil**(**countProducts **/** pageSize**),**

**});**

**})**

**);**

productRouter**.**get**(**

"/categories"**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

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

res**.**send**(**categories**);**

**})**

**);**

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

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

**if** **(**product**)** **{**

res**.**send**(**product**);**

**}** **else** **{**

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

**}**

**});**

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

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

**if** **(**product**)** **{**

res**.**send**(**product**);**

**}** **else** **{**

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

**}**

**});**

**export** **default** productRouter**;**

**orderRouter.js**

**import** express from "express"**;**

**import** expressAsyncHandler from "express-async-handler"**;**

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

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

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

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

**const** orderRouter **=** express**.**Router**();**

orderRouter**.**get**(**

"/"**,**

isAuth**,**

isAdmin**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

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

res**.**send**(**orders**);**

**})**

**);**

orderRouter**.**post**(**

"/"**,**

isAuth**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

**const** newOrder **=** **new** Order**({**

// here we are considering x.\_id because we are referring to product object of orderModel

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

shippingAddress**:** req**.**body**.**shippingAddress**,**

paymentMethod**:** req**.**body**.**paymentMethod**,**

itemsPrice**:** req**.**body**.**itemsPrice**,**

shippingPrice**:** req**.**body**.**shippingPrice**,**

taxPrice**:** req**.**body**.**taxPrice**,**

totalPrice**:** req**.**body**.**totalPrice**,**

user**:** req**.**user**.**\_id**,**

**});**

//saving the orders in database..

**const** order **=** **await** newOrder**.**save**();**

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

**})**

**);**

orderRouter**.**get**(**

"/summary"**,**

isAuth**,**

isAdmin**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

**const** orders **=** **await** Order**.**aggregate**([**

**{**

$group**:** **{**

\_id**:** **null,**

numOrders**:** **{** $sum**:** 1 **},**

totalSales**:** **{** $sum**:** "$totalPrice" **},**

**},**

**},**

**]);**

**const** users **=** **await** User**.**aggregate**([**

**{**

$group**:** **{**

\_id**:** **null,**

numUsers**:** **{** $sum**:** 1 **},**

**},**

**},**

**]);**

**const** dailyOrders **=** **await** Order**.**aggregate**([**

**{**

$group**:** **{**

\_id**:** **{** $dateToString**:** **{** format**:** "%Y-%m-%d"**,** date**:** "$createdAt" **}** **},**

orders**:** **{** $sum**:** 1 **},**

sales**:** **{** $sum**:** "$totalPrice" **},**

**},**

**},**

**{** $sort**:** **{** \_id**:** 1 **}** **},**

**]);**

**const** productCategories **=** **await** Product**.**aggregate**([**

**{**

$group**:** **{**

\_id**:** "$category"**,**

count**:** **{** $sum**:** 1 **},**

**},**

**},**

**]);**

res**.**send**({** users**,** orders**,** dailyOrders**,** productCategories **});**

**})**

**);**

orderRouter**.**get**(**

"/mine"**,**

isAuth**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

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

res**.**send**(**orders**);**

**})**

**);**

orderRouter**.**get**(**

"/:id"**,**

isAuth**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

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

**if** **(**order**)** **{**

res**.**send**(**order**);**

**}** **else** **{**

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

**}**

**})**

**);**

orderRouter**.**put**(**

"/:id/deliver"**,**

isAuth**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

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

**if** **(**order**)** **{**

order**.**isDelivered **=** **true;**

order**.**deliveredAt **=** Date**.**now**();**

**await** order**.**save**();**

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

**}** **else** **{**

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

**}**

**})**

**);**

orderRouter**.**put**(**

"/:id/pay"**,**

isAuth**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

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

"user"**,**

"email name"

**);**

**if** **(**order**)** **{**

order**.**isPaid **=** **true;**

order**.**paidAt **=** Date**.**now**();**

order**.**paymentResult **=** **{**

id**:** req**.**body**.**id**,**

**status:** req**.**body**.status,**

update\_time**:** req**.**body**.**update\_time**,**

email\_address**:** req**.**body**.**email\_address**,**

**};**

**const** updatedOrder **=** **await** order**.**save**();**

mailgun**()**

**.**messages**()**

**.**send**(**

**{**

from**:** "Amazona <amazona@mg.yourdomain.com>"**,**

to**:** `${order.user.name} <${order.user.email}>`**,**

subject**:** `New order ${order.\_id}`**,**

html**:** payOrderEmailTemplate**(**order**),**

**},**

**(**error**,** body**)** **=>** **{**

**if** **(**error**)** **{**

console**.**log**(**error**);**

**}** **else** **{**

console**.**log**(**body**);**

**}**

**}**

**);**

res**.**send**({** message**:** "Order Paid"**,** order**:** updatedOrder **});**

**}** **else** **{**

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

**}**

**})**

**);**

orderRouter**.delete(**

"/:id"**,**

isAuth**,**

isAdmin**,**

expressAsyncHandler**(async** **(**req**,** res**)** **=>** **{**

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

**if** **(**order**)** **{**

**await** order**.**remove**();**

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

**}** **else** **{**

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

**}**

**})**

**);**

**export** **default** orderRouter**;**

**Models :**

**userModel.js**

**import** mongoose from "mongoose"**;**

**const** userSchema **=** **new** mongoose**.**Schema**(**

**{**

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

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

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

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

**},**

**{**

timestamps**:** **true,**

**}**

**);**

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

**export** **default** User**;**

**productModel.js**

**import** mongoose from "mongoose"**;**

**const** productSchema **=** **new** mongoose**.**Schema**(**

**{**

name**:** **{**

type**:** String**,**

required**:** **true,**

unique**:** **true,**

**},**

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

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

**images:** **[**String**],**

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

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

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

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

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

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

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

**},**

**{**

timestamps**:** **true,**

**}**

**);**

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

**export** **default** Product**;**

**orderModel.js**

**import** mongoose from 'mongoose'**;**

**const** orderSchema **=** **new** mongoose**.**Schema**(**

**{**

orderItems**:** **[**

**{**

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

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

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

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

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

product**:** **{**

type**:** mongoose**.**Schema**.**Types**.**ObjectId**,**

ref**:** 'Product'**,**

required**:** **true,**

**},**

**},**

**],**

shippingAddress**:** **{**

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

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

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

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

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

**location:** **{**

lat**:** Number**,**

lng**:** Number**,**

address**:** String**,**

name**:** String**,**

vicinity**:** String**,**

googleAddressId**:** String**,**

**},**

**},**

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

paymentResult**:** **{**

id**:** String**,**

**status:** String**,**

update\_time**:** String**,**

email\_address**:** String**,**

**},**

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

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

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

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

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

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

paidAt**:** **{** type**:** Date **},**

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

deliveredAt**:** **{** type**:** Date **},**

**},**

**{**

timestamps**:** **true,**

**}**

**);**

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

**export** **default** Order**;**

**FRONTEND**

**package.json**

{

"name": "frontend",

"proxy": "http://127.0.0.1:4000",

"version": "0.1.0",

"private": **true**,

"dependencies": {

"@emotion/react": "^11.10.4",

"@emotion/styled": "^11.10.4",

"@fortawesome/fontawesome": "^1.1.8",

"@fortawesome/react-fontawesome": "^0.2.0",

"@material-ui/core": "^4.12.4",

"@material-ui/icons": "^4.11.3",

"@mui/material": "^5.10.5",

"@paypal/react-paypal-js": "^7.6.0",

"@testing-library/jest-dom": "^5.16.5",

"@testing-library/react": "^13.3.0",

"@testing-library/user-event": "^13.5.0",

"axios": "^0.24.0",

"bootstrap": "^5.1.3",

"eslint-config-react-app": "^6.0.0",

"popper.js": "^1.16.1",

"react": "^18.2.0",

"react-bootstrap": "^2.1.0",

"react-dom": "^18.2.0",

"react-helmet-async": "^1.2.2",

"react-icons": "^4.4.0",

"react-router-bootstrap": "^0.26.0",

"react-router-dom": "^6.2.1",

"react-scripts": "5.0.1",

"react-toastify": "^8.1.0",

"use-reducer-logger": "^1.0.2",

"web-vitals": "^2.1.4"

},

"scripts": {

"start": "react-scripts start",

"build": "react-scripts build",

"test": "react-scripts test",

"eject": "react-scripts eject"

},

"eslintConfig": {

"extends": [

"react-app",

"react-app/jest"

]

},

"browserslist": {

"production": [

">0.2%",

"not dead",

"not op\_mini all"

],

"development": [

"last 1 chrome version",

"last 1 firefox version",

"last 1 safari version"

]

},

"devDependencies": {

"@babel/eslint-parser": "^7.15.4",

"@babel/preset-react": "^7.14.5",

"@types/node": "^16.9.0",

"@typescript-eslint/eslint-plugin": "^4.31.0",

"@typescript-eslint/parser": "^4.31.0",

"babel-eslint": "^10.1.0",

"eslint": "^7.32.0",

"prettier": "^2.4.0",

"typescript": "^4.4.2"

}

}

**Index.html**

<!DOCTYPE html>

<html lang=**"en"**>

<head>

<meta charset=**"utf-8"** />

<link rel=**"icon"** href=**"%PUBLIC\_URL%/favicon.ico"** />

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

<meta name=**"theme-color"** content=**"#000000"** />

<meta

name=**"description"**

content=**"Web site created using create-react-app"**

/>

<link rel=**"apple-touch-icon"** href=**"%PUBLIC\_URL%/logo192.png"** />

<link rel=**"manifest"** href=**"%PUBLIC\_URL%/manifest.json"** />

<link rel=**"stylesheet"** href=**"https://cdn.jsdelivr.net/npm/@fortawesome/fontawesome-free@5.15.4/css/all.min.css"** />

<link rel=**"stylesheet"** href=**"https://fonts.googleapis.com/icon?family=Material+Icons"** />

<link rel=**"stylesheet"** href=**"https://fonts.googleapis.com/css?family=Roboto:300,400,500,700&display=swap"** />

<link rel=**"stylesheet"** href=**"https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.2.0/css/all.min.css"**

integrity=**"sha512-xh6O/CkQoPOWDdYTDqeRdPCVd1SpvCA9XXcUnZS2FmJNp1coAFzvtCN9BmamE+4aHK8yyUHUSCcJHgXloTyT2A=="**

crossorigin=**"anonymous"** referrerpolicy=**"no-referrer"** />

<title>**React App**</title>

</head>

<body>

<noscript>**You need to enable JavaScript to run this app.**</noscript>

<div id=**"root"**></div>

</body>

</html>

**index.js**

**import** React from "react"**;**

**import** ReactDOM from "react-dom/client"**;**

**import** "./index.css"**;**

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

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

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

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

**import** App from "./App"**;**

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

root**.**render**(**

**<**React**.**StrictMode**>**

**<**StoreProvider**>**

**<**HelmetProvider**>**

**<**PayPalScriptProvider deferLoading**={true}>**

**<**App **/>**

**</**PayPalScriptProvider**>**

**</**HelmetProvider**>**

**</**StoreProvider**>**

**</**React**.**StrictMode**>**

**);**

**utils.js**

**export** **const** getError **=** **(**error**)** **=>** **{**

**return** error**.**response **&&** error**.**response**.**data**.**message

**?** error**.**response**.**data**.**message

**:** error**.**message**;**

**};**

**Store.js**

**import** **{** createContext**,** useReducer **}** from "react"**;**

**export** **const** Store **=** createContext**();**

**const** initialState **=** **{**

fullBox**:** **false,**

userInfo**:** localStorage**.**getItem**(**"userInfo"**)**

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

**:** **null,**

cart**:** **{**

shippingAddress**:** localStorage**.**getItem**(**"shippingAddress"**)**

**?** JSON**.**parse**(**localStorage**.**getItem**(**"shippingAddress"**))**

**:** **{** **location:** **{}** **},**

paymentMethod**:** localStorage**.**getItem**(**"paymentMethod"**)**

**?** localStorage**.**getItem**(**"paymentMethod"**)**

**:** ""**,**

cartItems**:** localStorage**.**getItem**(**"cartItems"**)**

**?** JSON**.**parse**(**localStorage**.**getItem**(**"cartItems"**))**

**:** **[],**

**},**

**};**

**function** reducer**(**state**,** action**)** **{**

**switch** **(**action**.**type**)** **{**

**case** "CART\_ADD\_ITEM"**:**

//add to cart...

// checking for existing item

**const** newItem **=** action**.**payload**;**

**const** existItem **=** state**.**cart**.**cartItems**.**find**(**

**(**item**)** **=>** item**.**\_id **===** newItem**.**\_id

**);**

**const** cartItems **=** existItem

**?** state**.**cart**.**cartItems**.**map**((**item**)** **=>**

item**.**\_id **===** existItem**.**\_id **?** newItem **:** item

**)**

**:** **[...**state**.**cart**.**cartItems**,** newItem**];**

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

**return** **{**

**...**state**,**

cart**:** **{**

**...**state**.**cart**,**

cartItems**,**

**},**

**};**

**case** "CART\_REMOVE\_ITEM"**:** **{**

**const** cartItems **=** state**.**cart**.**cartItems**.**filter**(**

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

**);**

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

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

**}**

**case** "CART\_CLEAR"**:**

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

**case** "USER\_SIGNIN"**:**

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

**case** "USER\_SIGNOUT"**:**

**return** **{**

**...**state**,**

userInfo**:** **null,**

cart**:** **{**

cartItems**:** **[],**

shippingAddress**:** **{},**

paymentMethod**:** ""**,**

**},**

**};**

**case** "SAVE\_SHIPPING\_ADDRESS"**:**

**return** **{**

**...**state**,**

cart**:** **{**

**...**state**.**cart**,**

shippingAddress**:** action**.**payload**,**

**},**

**};**

**case** "SAVE\_PAYMENT\_METHOD"**:**

**return** **{**

**...**state**,**

cart**:** **{** **...**state**.**cart**,** paymentMethod**:** action**.**payload **},**

**};**

**default:**

**return** state**;**

**}**

**}**

**export** **function** StoreProvider**(**props**)** **{**

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

**const** value **=** **{** state**,** dispatch **};**

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

**}**

**index.css**

body **{**

**margin: 0;**

**font-family: -apple-system, BlinkMacSystemFont,** 'Segoe UI'**,** 'Roboto'**,** 'Oxygen'**,**

'Ubuntu'**,** 'Cantarell'**,** 'Fira Sans'**,** 'Droid Sans'**,** 'Helvetica Neue'**,**

**sans-serif;**

**-webkit-font-smoothing: antialiased;**

-moz-osx-font-smoothing**: grayscale;**

**}**

code **{**

**font-family: source-code-pro, Menlo, Monaco, Consolas,** 'Courier New'**,**

**monospace;**

**}**

**.**site-container**{**

**min-height:100vh;**

**}**

main **{**

**flex: 1;**

**}**

**.**btn-primary**,**

**.**btn-outline-primary **{**

**background-color: #f0c040;**

**color: #000000;**

**}**

**.**products **{**

**display: flex;**

**flex-wrap: wrap;**

**justify-content: center;**

**}**

**.**product **{**

**border: 1px #404040 solid;**

**margin: 1rem;**

**}**

**.**product img **{**

**width: 100%;**

**max-width: 400px;**

**}**

**.**product-info **{**

**padding: 1rem;**

**}**

**.**rating span **{**

**color: #f08000;**

**}**

**.**img-large**{**

**max-width:100%;**

**}**

**.**img-thumbnail**{**

**height:80px;**

**}**

**.**small-container**{**

**max-width: 600px;**

**align-items: center;**

**justify-content: center;**

**left: 200px;**

**right: 400px;**

**}**

**.**checkout-steps**>**div **{**

**border-bottom: 0.2rem solid #a0a0a0;**

**color: #a0a0a0;**

**}**

**.**checkout-steps**>**div**.**active **{**

**border-bottom: 0.2rem solid #f08000;**

**color: #f08000;**

**}**

/\* for sidebar \*/

**.**navbar-brand **{**

**font-weight: bold;**

**}**

**.**side-navbar **{**

**width: 240px;**

**height: 100%;**

**position: absolute;**

**left: -300px;**

**background-color: #100901;**

**transition: 0.5s;**

**}**

**.**site-container **{**

**transition: 0.4s;**

**}**

**.**active-cont **{**

**margin-left: 240px;**

**}**

**.**active-nav **{**

**left: 0;**

**}**

**.**text-bold **{**

**font-weight: bold;**

**}**

/\* testing for new login screen... \*/

**.**login **{**

**margin-top: 0;**

**margin-top: auto;**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**box-sizing: border-box;**

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

**background-color: white;**

**height: 100vh;**

**}**

**.**login\_signIn **{**

**display: flex;**

**align-items: center;**

**justify-content: center;**

**background: #51E1ED;**

**font-weight: 500;**

**border: 1px solid;**

**border-color: #a88734;**

**color: black;**

**border-radius: 0.3rem;**

**transition: 1s ease all;**

**}**

**.**login\_container **{**

**width: 350px;**

**margin-top: 4rem;**

**display: flex;**

**flex-direction: column;**

**border-radius: 5px;**

**padding: 15px;**

**box-sizing: border-box;**

**box-shadow: 10px lightgray;**

**border: 2px solid black;**

**}**

**.**login\_container h1 **{**

**margin-bottom: 20px;**

**}**

**.**login\_container**>**form**>**h5 **{**

**margin-bottom: 5px;**

**font-weight: 550;**

**}**

**.**login\_container form input **{**

**width: 100%;**

**background-color: white;**

**transition: 2s all;**

**height: 30px;**

**margin-bottom: 5px;**

**border-color: #51E1ED;**

**}**

**.**login\_container p **{**

**font-weight: 400;**

**font-size: 12px;**

**margin-top: 14px;**

**}**

**.**login\_container form input**:hover** **{**

**width: 95%;**

**}**

**.**login\_register\_button **{**

**background: #51E1ED;**

**border: 1px solid;**

**padding: 0.4rem;**

**border-color: #a88734;**

**border-radius: 0.4rem;**

**color: #000000;**

**font-weight: 500;**

**transition: 1s ease all;**

**width: auto;**

**}**

**.**login\_signIn**:hover** **{**

**background-color: #000000;**

**color: #ffffff;**

**}**

**.**login\_register\_button**:hover** **{**

**background-color: #000000;**

**color: #ffffff;**

**text-decoration: underline;**

**}**

/\* home settings \*/

**.**home **{**

**display: flex;**

**justify-content: center;**

**margin-left: auto;**

**margin-right: auto;**

**max-width: 1500px;**

**}**

**.**home\_row **{**

**display: flex;**

**z-index: 1;**

**margin-left: 5px;**

**margin-right: 5px;**

**}**

**.**home\_image **{**

**width: 100%;**

**z-index: -1;**

**margin-bottom: -150px;**

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

**}**

**.**product **{**

**background-color: white;**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**justify-content: flex-end;**

**width: 100%;**

**min-width: 100px;**

**max-height: 800px;**

**padding: 15px;**

**margin: 10px;**

**z-index: 1;**

**box-sizing: border-box;**

**box-shadow: 10px lightgray;**

**transition: 0.6s all;**

**}**

**.**product**:hover** **{**

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

**transform: translateY(-20px);**

**}**

**.**product\_info **{**

**color: black;**

**height: 100px;**

**margin-bottom: 15px;**

**word-wrap: 1/2;**

**}**

**.**product\_price **{**

**margin-top: 2px;**

**}**

**.**product\_rating **{**

**display: flex;**

**justify-content: space-around;**

**align-items: center;**

**}**

**.**product img **{**

**max-height: 200px;**

**width: 100%;**

**object-fit: contain;**

**margin-bottom: 15px;**

**}**

**.**product button **{**

**display: flex;**

**align-items: center;**

**justify-content: center;**

**background: #f0c14b;**

**border: 1px solid;**

**margin-top: 10px;**

**border-color: #a88734;**

**color: black;**

**}**

**App.js**

**import** "./App.css"**;**

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

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

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

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

**import** HomeScreen from "./screens/HomeScreen"**;**

**import** ProductScreen from "./screens/ProductScreen"**;**

**import** Navbar from "react-bootstrap/Navbar"**;**

**import** Container from "react-bootstrap/Container"**;**

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

**import** Badge from "react-bootstrap/Badge"**;**

**import** Nav from "react-bootstrap/Nav"**;**

**import** NavDropdown from "react-bootstrap/NavDropdown"**;**

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

**import** **{** useContext **}** from "react"**;**

**import** SearchBox from "./components/SearchBox"**;**

**import** SigninScreen from "./screens/SigninScreen"**;**

**import** ShippingAddressScreen from "./screens/ShippingAddressScreen"**;**

**import** SignupScreen from "./screens/SignupScreen"**;**

**import** PaymentScreen from "./screens/PaymentScreen"**;**

**import** PlaceOrderScreen from "./screens/PlaceOrderScreen"**;**

**import** OrderScreen from "./screens/OrderScreen"**;**

**import** OrderHistoryScreen from "./screens/OrderHistoryScreen"**;**

**import** ProfileScreen from "./screens/ProfileScreen"**;**

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

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

**import** axios from "axios"**;**

**import** CartScreen from "./screens/CartScreen"**;**

**import** SearchScreen from "./screens/SearchScreen"**;**

**import** ProtectedRoute from "./components/ProtectedRoute"**;**

**import** AdminRoute from "./components/AdminRoute"**;**

**import** DashboardScreen from "./screens/DashboardScreen"**;**

**import** CashOnDelivery from "./screens/CashOnDelivery"**;**

**import** ShoppingBasket from "@material-ui/icons/ShoppingBasket"**;**

**import** **{** FaUserAlt **}** from "react-icons/fa"**;**

**import** **{** FaSignOutAlt **}** from "react-icons/fa"**;**

**function** App**()** **{**

**const** **{** state**,** dispatch**:** ctxDispatch **}** **=** useContext**(**Store**);**

**const** **{** cart**,** userInfo **}** **=** state**;**

**const** signoutHandler **=** **()** **=>** **{**

ctxDispatch**({** type**:** "USER\_SIGNOUT" **});**

localStorage**.**removeItem**(**"userInfo"**);**

localStorage**.**removeItem**(**"shippingAddress"**);**

localStorage**.**removeItem**(**"paymentMethod"**);**

**window.location.**href **=** "/signin"**;**

**};**

**const** **[**sidebarIsOpen**,** setSidebarIsOpen**]** **=** useState**(false);**

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

useEffect**(()** **=>** **{**

**const** fetchCategories **=** **async** **()** **=>** **{**

**try** **{**

**const** **{** data **}** **=** **await** axios**.**get**(**`/api/products/categories`**);**

setCategories**(**data**);**

**}** **catch** **(**err**)** **{**

toast**.**error**(**getError**(**err**));**

**}**

**};**

fetchCategories**();**

**},** **[]);**

**return** **(**

//the first one is for active sidebar

**<**BrowserRouter**>**

**<**div

className**={**

sidebarIsOpen

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

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

**}**

**>**

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

**<**header**>**

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

**<**Container**>**

**<**Button

variant**=**"dark"

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

**>**

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

**</**Button**>**

**<**LinkContainer to**=**"/"**>**

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

**</**LinkContainer**>**

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

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

**<**SearchBox **/>**

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

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

**<**ShoppingBasket **/>**

**{**cart**.**cartItems**.**length **>** 0 **&&** **(**

**<**Badge pill bg**=**"danger"**>**

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

**</**Badge**>**

**)}**

**</**Link**>**

**{**userInfo **?** **(**

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

**<**LinkContainer to**=**"/profile"**>**

**<**NavDropdown**.**Item**>**

**<**p**>**User Profile**</**p**>**

**</**NavDropdown**.**Item**>**

**</**LinkContainer**>**

**<**LinkContainer to**=**"/orderhistory"**>**

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

**</**LinkContainer**>**

**<**NavDropdown**.**Divider **/>**

**<**Link

className**=**"dropdown-item"

to**=**"#signout"

onClick**={**signoutHandler**}**

**>**

**<**p**>**

Signout

**<**FaSignOutAlt **/>**

**</**p**>**

**</**Link**>**

**</**NavDropdown**>**

**)** **:** **(**

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

**<**FaUserAlt **/>**

**</**Link**>**

**)}**

**{**userInfo **&&** userInfo**.**isAdmin **&&** **(**

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

**<**LinkContainer to**=**"/admin/dashboard"**>**

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

**</**LinkContainer**>**

**<**LinkContainer to**=**"/admin/products"**>**

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

**</**LinkContainer**>**

**<**LinkContainer to**=**"/admin/orders"**>**

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

**</**LinkContainer**>**

**<**LinkContainer to**=**"/admin/users"**>**

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

**</**LinkContainer**>**

**</**NavDropdown**>**

**)}**

**</**Nav**>**

**</**Navbar**.**Collapse**>**

**</**Container**>**

**</**Navbar**>**

**</**header**>**

**<**div

className**={**

sidebarIsOpen

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

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

**}**

**>**

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

**<**Nav**.**Item**>**

**<**strong**>**Categories**</**strong**>**

**</**Nav**.**Item**>**

**{**categories**.**map**((**category**)** **=>** **(**

**<**Nav**.**Item key**={**category**}>**

**<**LinkContainer

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

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

**>**

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

**</**LinkContainer**>**

**</**Nav**.**Item**>**

**))}**

**</**Nav**>**

**</**div**>**

**<**main className**=**"mt-3"**>**

**<**Container**>**

**<**Routes**>**

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

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

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

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

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

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

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

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

**<**Route

path**=**"/profile"

**element={**

**<**ProtectedRoute**>**

**<**ProfileScreen **/>**

**</**ProtectedRoute**>**

**}**

**/>**

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

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

**<**Route

path**=**"/order/:id"

**element={**

**<**ProtectedRoute**>**

**<**OrderScreen **/>**

**</**ProtectedRoute**>**

**}**

**/>**

**<**Route

path**=**"/orderhistory"

**element={**

**<**ProtectedRoute**>**

**<**OrderHistoryScreen **/>**

**</**ProtectedRoute**>**

**}**

**/>**

**<**Route

path**=**"/admin/dashboard"

**element={**

**<**AdminRoute**>**

**<**DashboardScreen **/>**

**</**AdminRoute**>**

**}**

**/>**

**</**Routes**>**

**</**Container**>**

**</**main**>**

**<**footer**>**

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

All rights reserved @2022

**</**div**>**

**</**footer**>**

**</**div**>**

**</**BrowserRouter**>**

**);**

**}**

**export** **default** App**;**

**Screens**

**Carousel.js**

import React from "react";

import Carousel from "react-bootstrap/Carousel";

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

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

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

const Carousel1 = () => {

return (

<Carousel>

<Carousel.Item>

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

</Carousel.Item>

<Carousel.Item>

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

</Carousel.Item>

<Carousel.Item>

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

</Carousel.Item>

</Carousel>

);

};

export default Carousel1;

**CartScreen.js**

**import** **{** useContext **}** from "react"**;**

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

**import** Row from "react-bootstrap/Row"**;**

**import** Col from "react-bootstrap/Col"**;**

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

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

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

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

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

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

**import** axios from "axios"**;**

**export** **default** **function** CartScreen**()** **{**

**const** **navigate** **=** useNavigate**();**

**const** **{** state**,** dispatch**:** ctxDispatch **}** **=** useContext**(**Store**);**

**const** **{**

cart**:** **{** cartItems **},**

**}** **=** state**;**

**const** updateCartHandler **=** **async** **(**item**,** quantity**)** **=>** **{**

**const** **{** data **}** **=** **await** axios**.**get**(**`/api/products/${item.\_id}`**);**

**if** **(**data**.**countInStock **<** quantity**)** **{**

**window.alert(**"Sorry. Product is out of stock"**);**

**return;**

**}**

ctxDispatch**({**

type**:** "CART\_ADD\_ITEM"**,**

payload**:** **{** **...**item**,** quantity **},**

**});**

**};**

**const** removeItemHandler **=** **(**item**)** **=>** **{**

ctxDispatch**({** type**:** "CART\_REMOVE\_ITEM"**,** payload**:** item **});**

**};**

**const** checkoutHandler **=** **()** **=>** **{**

**navigate(**"/signin?redirect=/shipping"**);**

**};**

**return** **(**

**<**div**>**

**<**Helmet**>**

**<**title**>**Shopping Cart**</**title**>**

**</**Helmet**>**

**<**h1**>**Shopping Cart**</**h1**>**

**<**Row**>**

**<**Col md**={**8**}>**

**{**cartItems**.**length **===** 0 **?** **(**

**<**MessageBox**>**

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

**</**MessageBox**>**

**)** **:** **(**

**<**ListGroup**>**

**{**cartItems**.**map**((**item**)** **=>** **(**

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

**{**" "**}**

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

**{**" "**}**

**<**Col md**={**4**}>**

**<**img

src**={**item**.image}**

alt**={**item**.**name**}**

className**=**"img-fluid rounded img-thumbnail"

**></**img**>{**" "**}**

**<**Link to**={**`/product/${item.slug}`**}>{**item**.**name**}</**Link**>**

**</**Col**>**

**<**Col md**={**3**}>**

**<**Button

variant**=**"light"

onClick**={()** **=>** **{**

updateCartHandler**(**item**,** item**.**quantity **-** 1**);**

**}}**

disabled**={**item**.**quantity **===** 1**}**

**>**

**<**i className**=**"fas fa-minus-circle"**></**i**>**

**</**Button**>{**" "**}**

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

**<**Button

variant**=**"light"

onClick**={()** **=>** **{**

updateCartHandler**(**item**,** item**.**quantity **+** 1**);**

**}}**

disabled**={**item**.**quantity **===** item**.**countInStock**}**

**>**

**<**i className**=**"fas fa-plus-circle"**></**i**>**

**</**Button**>**

**</**Col**>**

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

**<**Col md**={**2**}>**

**<**Button

variant**=**"light"

onClick**={()** **=>** removeItemHandler**(**item**)}**

**>**

**<**i className**=**"fas fa-trash"**></**i**>**

**</**Button**>**

**</**Col**>**

**</**Row**>**

**</**ListGroup**.**Item**>**

**))}**

**</**ListGroup**>**

**)}{**" "**}**

**</**Col**>**

**<**Col md**={**4**}>**

**<**Card**>**

**<**Card**.**Body**>**

**<**ListGroup variant**=**"flush"**>**

**<**ListGroup**.**Item**>**

**<**h3**>**

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

items**)** **:** $

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

**</**h3**>**

**</**ListGroup**.**Item**>**

**<**ListGroup**.**Item**>**

**<**div className**=**"d-grid"**>**

**<button**

className**=**"login\_register\_button"

style**={{** textDecoration**:** "none" **}}**

type**=**"button"

variant**=**"info"

onClick**={**checkoutHandler**}**

disabled**={**cartItems**.**length **===** 0**}**

**>**

Proceed to Checkout

**</button>**

**</**div**>**

**</**ListGroup**.**Item**>**

**</**ListGroup**>**

**</**Card**.**Body**>**

**</**Card**>**

**</**Col**>**

**</**Row**>**

**</**div**>**

**);**

**}**

**CashOnDelivery.js**

**import** React from "react"**;**

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

**const** CashOnDelivery **=** **()** **=>** **{**

**return** **(**

**<**div**>**

**<**Helmet**>**

**<**title**>**Cash On Delivery**</**title**>**

**</**Helmet**>**

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

**</**div**>**

**);**

**};**

**export** **default** CashOnDelivery**;**

**DashboardScreen.js**

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

**import** axios from "axios"**;**

**import** logger from "use-reducer-logger"**;**

**import** Row from "react-bootstrap/Row"**;**

**import** Col from "react-bootstrap/Col"**;**

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

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

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

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

**const** reducer **=** **(**state**,** action**)** **=>** **{**

**switch** **(**action**.**type**)** **{**

**case** "FETCH\_REQUEST"**:**

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

**case** "FETCH\_SUCCESS"**:**

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

**case** "FETCH\_FAIL"**:**

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

**default:**

**return** state**;**

**}**

**};**

**const** DashboardScreen **=** **()** **=>** **{**

// here useReducer needs two parameter

**const** **[{** loading**,** error**,** products **},** dispatch**]** **=** useReducer**(**logger**(**reducer**),** **{**

products**:** **[],**

loading**:** **true,**

error**:** ""**,**

**});**

// const [products, setProducts] = useState([]);

useEffect**(()** **=>** **{**

**const** fetchData **=** **async** **()** **=>** **{**

dispatch**({** type**:** "FETCH\_REQUEST" **});**

**try** **{**

**const** result **=** **await** axios**.**get**(**"/api/products"**);**

dispatch**({** type**:** "FETCH\_SUCCESS"**,** payload**:** result**.**data **});**

**}** **catch** **(**err**)** **{**

dispatch**({** type**:** "FETCH\_FAIL"**,** payload**:** err**.**message **});**

**}**

// setProducts(result.data);

**};**

fetchData**();**

**},** **[]);**

**return** **(**

**<**div**>**

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

**<**Helmet**>**

**<**title**>**Amazon**</**title**>**

**</**Helmet**>**

**<**h1**>**Featured Products**</**h1**>**

**<**h2**>**

**<**div className**=**"products"**>**

**{**loading **?** **(**

**<**LoadingBox **/>**

**)** **:** error **?** **(**

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

**)** **:** **(**

**<**Row**>**

**{**products**.**map**((**product**)** **=>** **(**

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

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

**</**Col**>**

**))}**

**</**Row**>**

**)}**

**</**div**>**

**</**h2**>**

**</**div**>**

**);**

**};**

**export** **default** DashboardScreen**;**

**HomeScreen.js**

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

import axios from "axios";

import logger from "use-reducer-logger";

import Row from "react-bootstrap/Row";

import Col from "react-bootstrap/Col";

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

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

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

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

import Carousel from "./Carousel";

const reducer = (state, action) => {

switch (action.type) {

case "FETCH\_REQUEST":

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

case "FETCH\_SUCCESS":

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

case "FETCH\_FAIL":

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

default:

return state;

}

};

const HomeScreen = () => {

// here useReducer needs two parameter

const [{ loading, error, products }, dispatch] = useReducer(logger(reducer), {

products: [],

loading: true,

error: "",

});

// const [products, setProducts] = useState([]);

useEffect(() => {

const fetchData = async () => {

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

try {

const result = await axios.get("/api/products");

dispatch({ type: "FETCH\_SUCCESS", payload: result.data });

} catch (err) {

dispatch({ type: "FETCH\_FAIL", payload: err.message });

}

// setProducts(result.data);

};

fetchData();

}, []);

return (

<div className="container-fluid">

<Carousel />

<Helmet>

<title>Shop</title>

</Helmet>

<h1>Featured Products</h1>

<h2>

<div className="products">

{loading ? (

<LoadingBox />

) : error ? (

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

) : (

<Row>

{products.map((product) => (

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

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

</Col>

))}

</Row>

)}

</div>

</h2>

</div>

);

};

export default HomeScreen;

**OrderHistory.js**

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

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

**import** axios from "axios"**;**

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

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

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

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

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

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

**const** reducer **=** **(**state**,** action**)** **=>** **{**

**switch** **(**action**.**type**)** **{**

**case** "FETCH\_REQUEST"**:**

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

**case** "FETCH\_SUCCESS"**:**

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

**case** "FETCH\_FAIL"**:**

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

**default:**

**return** state**;**

**}**

**};**

**const** OrderHistoryScreen **=** **()** **=>** **{**

**const** **{** state **}** **=** useContext**(**Store**);**

**const** **{** userInfo **}** **=** state**;**

**const** **navigate** **=** useNavigate**();**

**const** **[{** loading**,** error**,** orders **},** dispatch**]** **=** useReducer**(**reducer**,** **{**

loading**:** **true,**

error**:** ""**,**

**});**

useEffect**(()** **=>** **{**

**const** fetchData **=** **async** **()** **=>** **{**

dispatch**({** type**:** "FETCH\_REQUEST" **});**

**try** **{**

**const** **{** data **}** **=** **await** axios**.**get**(**

`/api/orders/mine`**,**

**{** headers**:** **{** Authorization**:** `Bearer ${userInfo.token}` **}** **}**

**);**

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

**}** **catch** **(**error**)** **{**

dispatch**({**

type**:** "FETCH\_FAIL"**,**

payload**:** getError**(**error**),**

**});**

**}**

**};**

fetchData**();**

**},** **[**userInfo**]);**

**return** **(**

**<**div**>**

**<**Helmet**>**

**<**title**>**Order History**</**title**>**

**</**Helmet**>**

**<**h1**>**Order History**</**h1**>**

**{**loading **?** **(**

**<**LoadingBox**></**LoadingBox**>**

**)** **:** error **?** **(**

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

**)** **:** **(**

**<**table className**=**"table"**>**

**<**thead**>**

**<**tr**>**

**<**th**>**ID**</**th**>**

**<**th**>**DATE**</**th**>**

**<**th**>**TOTAL**</**th**>**

**<**th**>**PAID**</**th**>**

**<**th**>**DELIVERED**</**th**>**

**<**th**>**ACTIONS**</**th**>**

**</**tr**>**

**</**thead**>**

**<**tbody**>**

**{**orders**.**map**((**order**)** **=>** **(**

**<**tr key**={**order**.**\_id**}>**

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

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

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

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

**<**td**>**

**{**order**.**isDelivered

**?** order**.**deliveredAt**.**substring**(**0**,** 10**)**

**:** "No"**}**

**</**td**>**

**<**td**>**

**<**Button

type**=**"button"

variant**=**"info"

onClick**={()** **=>** **{**

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

**}}**

**>**

Details

**</**Button**>**

**</**td**>**

**</**tr**>**

**))}**

**</**tbody**>**

**</**table**>**

**)}**

**</**div**>**

**);**

**};**

**export** **default** OrderHistoryScreen**;**

**OrderScreen.js**

**import** axios from "axios"**;**

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

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

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

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

**import** Row from "react-bootstrap/Row"**;**

**import** Col from "react-bootstrap/Col"**;**

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

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

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

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

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

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

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

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

**import** **{** toast **}** from "react-toastify"**;**

// import { toast } from "react-toastify";

**const** reducer **=** **(**state**,** action**)** **=>** **{**

**switch** **(**action**.**type**)** **{**

**case** "FETCH\_REQUEST"**:**

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

**case** "FETCH\_SUCCESS"**:**

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

**case** "FETCH\_FAIL"**:**

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

**case** "PAY\_REQUEST"**:**

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

**case** "PAY\_SUCCESS"**:**

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

**case** "PAY\_FAIL"**:**

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

**case** "PAY\_RESET"**:**

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

**case** "DELIVER\_REQUEST"**:**

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

**case** "DELIVER\_SUCCESS"**:**

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

**case** "DELIVER\_FAIL"**:**

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

**case** "DELIVER\_RESET"**:**

**return** **{**

**...**state**,**

loadingDeliver**:** **false,**

successDeliver**:** **false,**

**};**

**default:**

**return** state**;**

**}**

**};**

**const** OrderScreen **=** **()** **=>** **{**

**const** **navigate** **=** useNavigate**();**

**const** **{** state **}** **=** useContext**(**Store**);**

**const** **{** userInfo **}** **=** state**;**

**const** params **=** useParams**();**

**const** **{** id**:** orderId **}** **=** params**;**

**const** **[**

**{**

loading**,**

error**,**

order**,**

successPay**,**

loadingPay**,**

loadingDeliver**,**

successDeliver**,**

**},**

dispatch**,**

**]** **=** useReducer**(**reducer**,** **{**

loading**:** **true,**

order**:** **{},**

error**:** ""**,**

successPay**:** **false,**

loadingPay**:** **false,**

**});**

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

**function** createOrder**(**data**,** actions**)** **{**

**return** actions**.**order

**.**create**({**

purchase\_units**:** **[**

**{**

amount**:** **{** value**:** order**.**totalPrice **},**

**},**

**],**

**})**

**.**then**((**orderID**)** **=>** **{**

**return** orderID**;**

**});**

**}**

**function** onApprove**(**data**,** actions**)** **{**

// if order is successfull then do this..

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

**try** **{**

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

**const** **{** data **}** **=** **await** axios**.**put**(**

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

details**,**

**{**

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

**}**

**);**

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

toast**.**success**(**"Order is paid"**);**

**}** **catch** **(**err**)** **{**

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

toast**.**error**(**getError**(**err**));**

**}**

**});**

**}**

**function** onError**(**err**)** **{**

toast**.**error**(**getError**(**err**));**

**}**

useEffect**(()** **=>** **{**

**const** fetchOrder **=** **async** **()** **=>** **{**

**try** **{**

dispatch**({** type**:** "FETCH\_REQUEST" **});**

**const** **{** data **}** **=** **await** axios**.**get**(**`/api/orders/${orderId}`**,** **{**

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

**});**

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

**}** **catch** **(**err**)** **{**

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

**}**

**};**

**if** **(!**userInfo**)** **{**

**return** **navigate(**"/login"**);**

**}**

**if** **(**

**!**order**.**\_id **||**

successPay **||**

successDeliver **||**

**(**order**.**\_id **&&** order**.**\_id **!==** orderId**)**

**)** **{**

fetchOrder**();**

**if** **(**successPay**)** **{**

dispatch**({** type**:** "PAY\_RESET" **});**

**}**

**if** **(**successDeliver**)** **{**

dispatch**({** type**:** "DELIVER\_RESET" **});**

**}**

**}** **else** **{**

**const** loadPaypalScript **=** **async** **()** **=>** **{**

**const** **{** data**:** clientId **}** **=** **await** axios**.**get**(**"/api/keys/paypal"**,** **{**

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

**});**

paypalDispatch**({**

type**:** "resetOptions"**,**

value**:** **{**

"client-id"**:** clientId**,**

currency**:** "USD"**,**

**},**

**});**

paypalDispatch**({** type**:** "setLoadingStatus"**,** value**:** "pending" **});**

**};**

loadPaypalScript**();**

**}**

**},** **[**

order**,**

userInfo**,**

orderId**,**

**navigate,**

paypalDispatch**,**

successPay**,**

successDeliver**,**

**]);**

**const** deliverOrderHandler **=** **async** **()** **=>** **{**

**try** **{**

dispatch**({** type**:** "DELIVER\_REQUEST" **});**

**const** **{** data **}** **=** **await** axios**.**put**(**

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

**{},**

**{**

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

**}**

**);**

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

toast**.**success**(**"Order is delivered"**);**

**}** **catch** **(**err**)** **{**

toast**.**error**(**getError**(**err**));**

dispatch**({** type**:** "DELIVER\_FAIL" **});**

**}**

**};**

// async function deliverOrderHandler() {

// }

**return** loading **?** **(**

**<**LoadingBox**></**LoadingBox**>**

**)** **:** error **?** **(**

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

**)** **:** **(**

**<**div**>**

**<**Helmet**>**

**<**title**>**Order **{**orderId**}</**title**>**

**</**Helmet**>**

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

**<**Row**>**

**<**Col md**={**8**}>**

**<**Card className**=**"mb-3"**>**

**<**Card**.**Body**>**

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

**<**Card**.**Text**>**

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

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

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

**,{**order**.**shippingAddress**.**country**}**

**&**nbsp**;**

**{**order**.**shippingAddress**.location** **&&**

order**.**shippingAddress**.location.**lat **&&** **(**

**<**a

target**=**"\_new"

href**={**`https://maps.google.com?q=${order.shippingAddress.location.lat},${order.shippingAddress.location.lng}`**}**

**>**

Show On Map

**</**a**>**

**)}**

**</**Card**.**Text**>**

**{**/\* {order.isDelivered ? (

<MessageBox variant="success">

Delivered at {order.deliveredAt}

</MessageBox>

) : (

<MessageBox variant="danger">Not Delivered</MessageBox>

)} \*/**}**

**</**Card**.**Body**>**

**</**Card**>**

**<**Card className**=**"mb-3"**>**

**<**Card**.**Body**>**

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

**<**Card**.**Text**>**

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

**</**Card**.**Text**>**

**{**/\* {order.isPaid ? (

<MessageBox variant="success">

Paid at {order.paidAt}

</MessageBox>

) : (

<MessageBox variant="danger">Not Paid</MessageBox>

)} \*/**}**

**</**Card**.**Body**>**

**</**Card**>**

**<**Card className**=**"mb-3"**>**

**<**Card**.**Body**>**

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

**<**ListGroup variant**=**"flush"**>**

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

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

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

**<**Col md**={**6**}>**

**<**img

src**={**item**.image}**

alt**={**item**.**name**}**

className**=**"img-fluid rounded img-thumbnail"

**></**img**>{**" "**}**

**<**Link

style**={{** textDecoration**:** "none" **}}**

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

**>**

**{**item**.**name**}**

**</**Link**>**

**</**Col**>**

**<**Col md**={**3**}>**

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

**</**Col**>**

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

**</**Row**>**

**</**ListGroup**.**Item**>**

**))}**

**</**ListGroup**>**

**</**Card**.**Body**>**

**</**Card**>**

**</**Col**>**

**<**Col md**={**4**}>**

**<**Card className**=**"mb-3"**>**

**<**Card**.**Body**>**

**<**Card**.**Title**>**Order Summary**</**Card**.**Title**>**

**<**ListGroup variant**=**"flush"**>**

**<**ListGroup**.**Item**>**

**<**Row**>**

**<**Col**>**Items**</**Col**>**

**<**Col**>**$**{**order**.**itemsPrice**.**toFixed**(**2**)}</**Col**>**

**</**Row**>**

**</**ListGroup**.**Item**>**

**<**ListGroup**.**Item**>**

**<**Row**>**

**<**Col**>**Shipping**</**Col**>**

**<**Col**>**$**{**order**.**shippingPrice**.**toFixed**(**2**)}</**Col**>**

**</**Row**>**

**</**ListGroup**.**Item**>**

**<**ListGroup**.**Item**>**

**<**Row**>**

**<**Col**>**Tax**</**Col**>**

**<**Col**>**$**{**order**.**taxPrice**.**toFixed**(**2**)}</**Col**>**

**</**Row**>**

**</**ListGroup**.**Item**>**

**<**ListGroup**.**Item**>**

**<**Row**>**

**<**Col**>**

**<**strong**>** Order Total**</**strong**>**

**</**Col**>**

**<**Col**>**

**<**strong**>**$**{**order**.**totalPrice**.**toFixed**(**2**)}</**strong**>**

**</**Col**>**

**</**Row**>**

**</**ListGroup**.**Item**>**

**{**/\* if order is not paid then it is in pending state... \*/**}**

**{!**order**.**isPaid **&&** **(**

**<**ListGroup**.**Item**>**

**{**isPending **?** **(**

**<**LoadingBox **/>**

**)** **:** **(**

**<**div**>**

**<**PayPalButtons

createOrder**={**createOrder**}**

onApprove**={**onApprove**}**

onError**={**onError**}**

**></**PayPalButtons**>**

**</**div**>**

**)}**

**{**loadingPay **&&** **<**LoadingBox**></**LoadingBox**>}**

**</**ListGroup**.**Item**>**

**)}**

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

**<**Button variant**=**"info bold"**>**

**<**NavLink

style**={{**

textDecoration**:** "none"**,**

color**:** "black"**,**

fontWeight**:** "bold"**,**

**}}**

exact

to**=**"/cash"

**>**

Cash On Delivery

**</**NavLink**>**

**</**Button**>**

**{**userInfo**.**isAdmin **&&** order**.**isPaid **&&** **!**order**.**isDelivered **&&** **(**

**<**ListGroup**.**Item**>**

**{**loadingDeliver **&&** **<**LoadingBox**></**LoadingBox**>}**

**<**div className**=**"d-grid"**>**

**<**Button

variant**=**"info"

type**=**"button"

onClick**={**deliverOrderHandler**}**

**>**

Deliver Order

**</**Button**>**

**</**div**>**

**</**ListGroup**.**Item**>**

**)}**

**</**ListGroup**>**

**</**Card**.**Body**>**

**</**Card**>**

**</**Col**>**

**</**Row**>**

**</**div**>**

**);**

**};**

**export** **default** OrderScreen**;**

**PaymentScreen.js**

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

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

**import** Form from "react-bootstrap/Form"**;**

**import** Checkout from "../components/Checkout"**;**

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

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

**const** PaymentScreen **=** **()** **=>** **{**

**const** **navigate** **=** useNavigate**();**

**const** **{** state**,** dispatch**:** ctxDispatch **}** **=** useContext**(**Store**);**

**const** **{**

cart**:** **{** shippingAddress**,** paymentMethod **},**

**}** **=** state**;**

**const** **[**paymentMethodName**,** setPaymentMethod**]** **=** useState**(**

paymentMethod **||** "PayPal"

**);**

// checing if shipping address is present or not..

useEffect**(()** **=>** **{**

**if** **(!**shippingAddress**.**address**)** **{**

**navigate(**"/shipping"**);**

**}**

**},** **[**shippingAddress**,** **navigate]);**

**const** submitHandler **=** **(**e**)** **=>** **{**

e**.**preventDefault**();**

ctxDispatch**({**

type**:** "SAVE\_PAYMENT\_METHOD"**,**

payload**:** paymentMethodName**,**

**});**

localStorage**.**setItem**(**"paymentMethod"**,** paymentMethodName**);**

**navigate(**"/placeorder"**);**

**};**

**return** **(**

**<**div style**={{** justifyContent**:** "center"**,** alignItems**:** "center" **}}>**

**<**Checkout step1 step2 step3**></**Checkout**>**

**<**div className**=**"container small-container"**>**

**<**Helmet**>**Payment Gateway**</**Helmet**>**

**<**h1 className**=**"my-3"**>**Payment Method**</**h1**>**

**<**Form onSubmit**={**submitHandler**}>**

**<**div className**=**"mb-3"**>**

**<**Form**.**Check

type**=**"radio"

id**=**"PayPal"

label**=**"PayPal"

value**=**"PayPal"

checked**={**paymentMethodName **===** "PayPal"**}**

onChange**={(**e**)** **=>** setPaymentMethod**(**e**.**target**.**value**)}**

**/>**

**</**div**>**

**<**div className**=**"mb-3"**>**

**<**Form**.**Check

type**=**"radio"

id**=**"Stripe"

label**=**"Stripe"

value**=**"Stripe"

checked**={**paymentMethodName **===** "Stripe"**}**

onChange**={(**e**)** **=>** setPaymentMethod**(**e**.**target**.**value**)}**

**/>**

**</**div**>**

**<**div className**=**"mb-3"**>**

**<button**

className**=**"login\_register\_button"

style**={{** textDecoration**:** "none" **}}**

variant**=**"info"

type**=**"submit"

**>**

Continue

**</button>**

**</**div**>**

**</**Form**>**

**</**div**>**

**</**div**>**

**);**

**};**

**export** **default** PaymentScreen**;**

**PlaceOrderScreen.js**

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

**import** axios from "axios"**;**

**import** Container from "react-bootstrap/esm/Container"**;**

**import** Checkout from "../components/Checkout"**;**

**import** Row from "react-bootstrap/Row"**;**

**import** Col from "react-bootstrap/Col"**;**

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

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

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

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

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

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

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

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

**import** **{** toast **}** from "react-toastify"**;**

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

// creating a reducer function..

**const** reducer **=** **(**state**,** action**)** **=>** **{**

**switch** **(**action**.**type**)** **{**

**case** "CREATE\_REQUEST"**:**

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

**case** "CREATE\_SUCCESS"**:**

**return** **{** **...**state**,** loading**:** **false** **};**

**case** "CREATE\_FAIL"**:**

**return** **{** **...**state**,** loading**:** **false** **};**

**default:**

**return** state**;**

**}**

**};**

**const** PlaceOrderScreen **=** **()** **=>** **{**

**const** **navigate** **=** useNavigate**();**

**const** **[{** loading **},** dispatch**]** **=** useReducer**(**reducer**,** **{**

loading**:** **false,**

**});**

**const** **{** state**,** dispatch**:** ctxDispatch **}** **=** useContext**(**Store**);**

**const** **{** cart**,** userInfo **}** **=** state**;**

// we are doing this for roundoff the payment

**const** round2 **=** **(**num**)** **=>** Math**.**round**(**num **\*** 100 **+** Number**.**EPSILON**)** **/** 100**;** // 123.2345 => 123.23

cart**.**itemsPrice **=** round2**(**

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

**);**

cart**.**shippingPrice **=** cart**.**itemsPrice **>** 100 **?** round2**(**0**)** **:** round2**(**10**);**

cart**.**taxPrice **=** round2**(**0.15 **\*** cart**.**itemsPrice**);**

cart**.**totalPrice **=** cart**.**itemsPrice **+** cart**.**shippingPrice **+** cart**.**taxPrice**;**

**const** placeOrderHandler **=** **async** **()** **=>** **{**

**try** **{**

dispatch**({** type**:** "CREATE\_REQUEST" **});**

**const** **{** data **}** **=** **await** axios**.**post**(**

"/api/orders"**,**

**{**

orderItems**:** cart**.**cartItems**,**

shippingAddress**:** cart**.**shippingAddress**,**

paymentMethod**:** cart**.**paymentMethod**,**

itemsPrice**:** cart**.**itemsPrice**,**

shippingPrice**:** cart**.**shippingPrice**,**

taxPrice**:** cart**.**taxPrice**,**

totalPrice**:** cart**.**totalPrice**,**

**},**

**{**

headers**:** **{**

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

**},**

**}**

**);**

// it goes to store

ctxDispatch**({** type**:** "CART\_CLEAR" **});**

// it goes to reducer up

dispatch**({** type**:** "CREATE\_SUCCESS" **});**

localStorage**.**removeItem**(**"cartItems"**);**

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

**}** **catch** **(**err**)** **{**

dispatch**({** type**:** "CREATE\_FAIL" **});**

toast**.**error**(**getError**(**err**));**

**}**

**};**

useEffect**(()** **=>** **{**

**if** **(!**cart**.**paymentMethod**)** **{**

**navigate(**"/payment"**);**

**}**

**},** **[**cart**,** **navigate]);**

**return** **(**

**<**Container**>**

**<**Checkout step1 step2 step3 step4**></**Checkout**>**

**<**Helmet**>**Preview Order**</**Helmet**>**

**<**h1 className**=**"my-3"**>**Preview Order**</**h1**>**

**<**Row**>**

**<**Col md**={**8**}>**

**<**Card className**=**"mb-3"**>**

**<**Card**.**Body**>**

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

**<**Card**.**Text**>**

**{**/\* all the details are coming from store \*/**}**

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

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

**{**cart**.**shippingAddress**.**city**},** **{**cart**.**shippingAddress**.**postalCode**},**

**{**cart**.**shippingAddress**.**country**}**

**</**Card**.**Text**>**

**{**/\* here if user wants to edit something they can revert back.. \*/**}**

**<**Link style**={{** textDecoration**:** "none" **}}** to**=**"/shipping"**>**

Edit

**</**Link**>**

**</**Card**.**Body**>**

**</**Card**>**

**<**Card className**=**"mb-3"**>**

**<**Card**.**Body**>**

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

**<**Card**.**Text**>**

**<**strong**>**Method**:</**strong**>** **{**cart**.**paymentMethod**}**

**</**Card**.**Text**>**

**<**Link style**={{** textDecoration**:** "none" **}}** to**=**"/payment"**>**

Edit

**</**Link**>**

**</**Card**.**Body**>**

**</**Card**>**

**<**Card className**=**"mb-3"**>**

**<**Card**.**Body**>**

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

**<**ListGroup variant**=**"flush"**>**

**{**cart**.**cartItems**.**map**((**item**)** **=>** **(**

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

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

**<**Col md**={**6**}>**

**<**img

src**={**item**.image}**

alt**={**item**.**name**}**

className**=**"img-fluid rounded img-thumbnail"

**></**img**>{**" "**}**

**<**Link to**={**`/product/${item.slug}`**}>{**item**.**name**}</**Link**>**

**</**Col**>**

**<**Col md**={**3**}>**

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

**</**Col**>**

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

**</**Row**>**

**</**ListGroup**.**Item**>**

**))}**

**</**ListGroup**>**

**<**Link style**={{** textDecoration**:** "none" **}}** to**=**"/cart"**>**

Edit

**</**Link**>**

**</**Card**.**Body**>**

**</**Card**>**

**</**Col**>**

**<**Col md**={**4**}>**

**<**Card**>**

**<**Card**.**Body**>**

**<**Card**.**Title**>**Order Summary**</**Card**.**Title**>**

**<**ListGroup variant**=**"flush"**>**

**<**ListGroup**.**Item**>**

**<**Row**>**

**<**Col**>**Items**</**Col**>**

**<**Col**>**$**{**cart**.**itemsPrice**.**toFixed**(**2**)}</**Col**>**

**</**Row**>**

**</**ListGroup**.**Item**>**

**<**ListGroup**.**Item**>**

**<**Row**>**

**<**Col**>**Shipping**</**Col**>**

**<**Col**>**$**{**cart**.**shippingPrice**.**toFixed**(**2**)}</**Col**>**

**</**Row**>**

**</**ListGroup**.**Item**>**

**<**ListGroup**.**Item**>**

**<**Row**>**

**<**Col**>**Tax**</**Col**>**

**<**Col**>**$**{**cart**.**taxPrice**.**toFixed**(**2**)}</**Col**>**

**</**Row**>**

**</**ListGroup**.**Item**>**

**<**ListGroup**.**Item**>**

**<**Row**>**

**<**Col**>**

**<**strong**>** Order Total**</**strong**>**

**</**Col**>**

**<**Col**>**

**<**strong**>**$**{**cart**.**totalPrice**.**toFixed**(**2**)}</**strong**>**

**</**Col**>**

**</**Row**>**

**</**ListGroup**.**Item**>**

**<**ListGroup**.**Item**>**

**<**div className**=**"d-grid"**>**

**<**Button

variant**=**"info"

type**=**"button"

onClick**={**placeOrderHandler**}**

disabled**={**cart**.**cartItems**.**length **===** 0**}**

**>**

Place Order

**</**Button**>**

**</**div**>**

**{**loading **&&** **<**LoadingBox**></**LoadingBox**>}**

**</**ListGroup**.**Item**>**

**</**ListGroup**>**

**</**Card**.**Body**>**

**</**Card**>**

**</**Col**>**

**</**Row**>**

**</**Container**>**

**);};**

**export** **default** PlaceOrderScreen**;**

**ProductScreen.js**

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

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

**import** axios from "axios"**;**

**import** Row from "react-bootstrap/Row"**;**

**import** Col from "react-bootstrap/Col"**;**

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

**import** Badge from "react-bootstrap/Badge"**;**

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

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

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

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

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

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

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

**const** reducer **=** **(**state**,** action**)** **=>** **{**

**switch** **(**action**.**type**)** **{**

**case** "FETCH\_REQUEST"**:**

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

**case** "FETCH\_SUCCESS"**:**

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

**case** "FETCH\_FAIL"**:**

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

**default:**

**return** state**;**

**}**

**};**

**const** ProductScreen **=** **()** **=>** **{**

**const** **navigate** **=** useNavigate**();**

**const** params **=** useParams**();**

**const** **{** slug **}** **=** params**;**

// here useReducer needs two parameter

**const** **[{** loading**,** error**,** product **},** dispatch**]** **=** useReducer**(**reducer**,** **{**

product**:** **[],**

loading**:** **true,**

error**:** ""**,**

**});**

// const [products, setProducts] = useState([]);

useEffect**(()** **=>** **{**

**const** fetchData **=** **async** **()** **=>** **{**

dispatch**({** type**:** "FETCH\_REQUEST" **});**

**try** **{**

**const** result **=** **await** axios**.**get**(**`/api/products/slug/${slug}`**);**

dispatch**({** type**:** "FETCH\_SUCCESS"**,** payload**:** result**.**data **});**

**}** **catch** **(**err**)** **{**

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

**}**

// setProducts(result.data);

**};**

fetchData**();**

// here if user switches from one product to another so we have to pass slug

**},** **[**slug**]);**

**const** **{** state**,** dispatch**:** ctxDispatch **}** **=** useContext**(**Store**);**

// change

**const** **{** cart **}** **=** state**;**

// const { cart, userInfo } = state;

**const** addToCartHandler **=** **async** **()** **=>** **{**

// change

**const** existItem **=** cart**.**cartItems**.**find**((**x**)** **=>** x**.**\_id **===** product**.**\_id**);**

**const** quantity **=** existItem **?** existItem**.**quantity **+** 1 **:** 1**;**

**const** **{** data **}** **=** **await** axios**.**get**(**`/api/products/${product.\_id}`**);**

**if** **(**data**.**countInStock **<** quantity**)** **{**

**window.alert(**"Sorry. Product is out of stock"**);**

**return;**

**}**

// till here

ctxDispatch**({**

type**:** "CART\_ADD\_ITEM"**,**

payload**:** **{** **...**product**,** quantity**:** 1 **},**

**});**

**navigate(**"/cart"**);**

**};**

**return** loading **?** **(**

**<**LoadingBox **/>**

**)** **:** error **?** **(**

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

**)** **:** **(**

**<**div**>**

**<**Row**>**

**<**Col md**={**6**}>**

**{**" "**}**

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

**</**Col**>**

**<**Col md**={**3**}>**

**<**ListGroup variant**=**"flush"**>**

**<**ListGroup**.**Item**>**

**<**Helmet**>**

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

**</**Helmet**>**

**</**ListGroup**.**Item**>**

**<**ListGroup**.**Item**>**

**<**Rating

rating**={**product**.**rating**}**

numReviews**={**product**.**numReviews**}**

**></**Rating**>**

**</**ListGroup**.**Item**>**

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

**<**ListGroup**.**Item**>**

Description**:**

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

**</**ListGroup**.**Item**>**

**</**ListGroup**>**

**</**Col**>**

**<**Col md**={**3**}>**

**<**Card**>**

**<**Card**.**Body**>**

**<**ListGroup variant**=**"flush"**>**

**<**ListGroup**.**Item**>**

**<**Row**>**

**<**Col**>**Price**</**Col**>**

**<**Col**>**$**{**product**.**price**}</**Col**>**

**</**Row**>**

**</**ListGroup**.**Item**>**

**<**ListGroup**.**Item**>**

**<**Row**>**

**<**Col**>**Status**</**Col**>**

**<**Col**>**

**{**product**.**countInStock **>** 0 **?** **(**

**<**Badge bg**=**"success"**>**In Stock**</**Badge**>**

**)** **:** **(**

**<**Badge bg**=**"danger"**>**Unavailable**</**Badge**>**

**)}**

**</**Col**>**

**</**Row**>**

**</**ListGroup**.**Item**>**

**{**product**.**countInStock **>** 0 **&&** **(**

**<**ListGroup**.**Item**>**

**<**div className**=**"d-grid"**>**

**<button**

className**=**"login\_register\_button"

style**={{** textDecoration**:** "none" **}}**

onClick**={**addToCartHandler**}**

variant**=**"info"

**>**

Add to Cart

**</button>**

**</**div**>**

**</**ListGroup**.**Item**>**

**)}**

**</**ListGroup**>**

**</**Card**.**Body**>**

**</**Card**>**

**</**Col**>**

**</**Row**>**

**</**div**>**

**);**

**};**

**export** **default** ProductScreen**;**

**ProfileScreen.js**

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

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

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

import Form from "react-bootstrap/Form";

import { toast } from "react-toastify";

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

import axios from "axios";

const reducer = (state, action) => {

switch (action.type) {

case "UPDATE\_REQUEST":

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

case "UPDATE\_SUCCESS":

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

case "UPDATE\_FAIL":

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

default:

return state;

}

};

const ProfileScreen = () => {

const { state, dispatch: ctxDispatch } = useContext(Store);

const { userInfo } = state;

const [name, setName] = useState(userInfo.name);

const [email, setEmail] = useState(userInfo.email);

const [password, setPassword] = useState("");

const [confirmPassword, setConfirmPassword] = useState("");

const [dispatch] = useReducer(reducer, {

loadingUpdate: false,

});

const submitHandler = async (e) => {

e.preventDefault();

try {

const { data } = await axios.put(

"/api/users/profile",

{

name,

email,

password,

},

{

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

}

);

dispatch({

type: "UPDATE\_SUCCESS",

});

ctxDispatch({ type: "USER\_SIGNIN", payload: data });

localStorage.setItem("userInfo", JSON.stringify(data));

toast.success("User updated successfully");

} catch (err) {

dispatch({

type: "FETCH\_FAIL",

});

toast.error(getError(err));

}

};

return (

<div className="login">

<div className="login\_container">

<Helmet>

<title>User Profile</title>

</Helmet>

<h1 className="text-center">User Profile</h1>

<form onSubmit={submitHandler}>

{" "}

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

<h5>Name</h5>

<input

className="login\_input"

type="text"

value={name}

onChange={(e) => {

setName(e.target.value);

}}

/>

</Form.Group>

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

<h5>Email</h5>

<input

className="login\_input"

type="email"

value={email}

onChange={(e) => {

setEmail(e.target.value);

}}

/>

</Form.Group>

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

<h5>Password</h5>

<input

className="login\_input"

type="password"

value={password}

onChange={(e) => {

setPassword(e.target.value);

}}

/>

</Form.Group>

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

<h5>Confirm Password</h5>

<input

className="login\_input"

type="password"

value={confirmPassword}

onChange={(e) => {

setConfirmPassword(e.target.value);

}}

/>

</Form.Group>

<div className="mb-3">

<button

style={{ textDecoration: "none" }}

className="login\_register\_button"

variant="info"

type="submit"

>

Update

</button>

</div>

</form>

</div>

</div>

);

};

export default ProfileScreen;

**SearchScreen.js**

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

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

**import** axios from "axios"**;**

**import** **{** toast **}** from "react-toastify"**;**

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

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

**import** Row from "react-bootstrap/Row"**;**

**import** Col from "react-bootstrap/Col"**;**

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

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

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

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

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

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

**const** reducer **=** **(**state**,** action**)** **=>** **{**

**switch** **(**action**.**type**)** **{**

**case** "FETCH\_REQUEST"**:**

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

**case** "FETCH\_SUCCESS"**:**

**return** **{**

**...**state**,**

products**:** action**.**payload**.**products**,**

page**:** action**.**payload**.**page**,**

pages**:** action**.**payload**.**pages**,**

countProducts**:** action**.**payload**.**countProducts**,**

loading**:** **false,**

**};**

**case** "FETCH\_FAIL"**:**

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

**default:**

**return** state**;**

**}**

**};**

**const** prices **=** **[**

**{**

name**:** "$1 to $50"**,**

value**:** "1-50"**,**

**},**

**{**

name**:** "$51 to $200"**,**

value**:** "51-200"**,**

**},**

**{**

name**:** "$201 to $1000"**,**

value**:** "201-1000"**,**

**},**

**];**

**export** **const** ratings **=** **[**

**{**

name**:** "4stars & up"**,**

rating**:** 4**,**

**},**

**{**

name**:** "3stars & up"**,**

rating**:** 3**,**

**},**

**{**

name**:** "2stars & up"**,**

rating**:** 2**,**

**},**

**{**

name**:** "1stars & up"**,**

rating**:** 1**,**

**},**

**];**

**export** **default** **function** SearchScreen**()** **{**

**const** **navigate** **=** useNavigate**();**

**const** **{** search **}** **=** useLocation**();**

**const** sp **=** **new** URLSearchParams**(**search**);** // /search?category=Shirts

**const** category **=** sp**.**get**(**"category"**)** **||** "all"**;**

**const** query **=** sp**.**get**(**"query"**)** **||** "all"**;**

**const** price **=** sp**.**get**(**"price"**)** **||** "all"**;**

**const** rating **=** sp**.**get**(**"rating"**)** **||** "all"**;**

**const** order **=** sp**.**get**(**"order"**)** **||** "newest"**;**

**const** page **=** sp**.**get**(**"page"**)** **||** 1**;**

**const** **[{** loading**,** error**,** products**,** pages**,** countProducts **},** dispatch**]** **=**

useReducer**(**reducer**,** **{**

loading**:** **true,**

error**:** ""**,**

**});**

useEffect**(()** **=>** **{**

**const** fetchData **=** **async** **()** **=>** **{**

**try** **{**

**const** **{** data **}** **=** **await** axios**.**get**(**

`/api/products/search?page=${page}&query=${query}&category=${category}&price=${price}&rating=${rating}&order=${order}`

**);**

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

**}** **catch** **(**err**)** **{**

dispatch**({**

type**:** "FETCH\_FAIL"**,**

payload**:** getError**(**error**),**

**});**

**}**

**};**

fetchData**();**

**},** **[**category**,** error**,** order**,** page**,** price**,** query**,** rating**]);**

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

useEffect**(()** **=>** **{**

**const** fetchCategories **=** **async** **()** **=>** **{**

**try** **{**

**const** **{** data **}** **=** **await** axios**.**get**(**`/api/products/categories`**);**

setCategories**(**data**);**

**}** **catch** **(**err**)** **{**

toast**.**error**(**getError**(**err**));**

**}**

**};**

fetchCategories**();**

**},** **[**dispatch**]);**

**const** getFilterUrl **=** **(**filter**)** **=>** **{**

**const** filterPage **=** filter**.**page **||** page**;**

**const** filterCategory **=** filter**.**category **||** category**;**

**const** filterQuery **=** filter**.**query **||** query**;**

**const** filterRating **=** filter**.**rating **||** rating**;**

**const** filterPrice **=** filter**.**price **||** price**;**

**const** sortOrder **=** filter**.**order **||** order**;**

**return** `/search?category=${filterCategory}&query=${filterQuery}&price=${filterPrice}&rating=${filterRating}&order=${sortOrder}&page=${filterPage}`**;**

**};**

**return** **(**

**<**div**>**

**<**Helmet**>**

**<**title**>**Search Products**</**title**>**

**</**Helmet**>**

**<**Row**>**

**<**Col md**={**3**}>**

**<**h3**>**Department**</**h3**>**

**<**div**>**

**<**ul**>**

**<**li**>**

**<**Link

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

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

**>**

Any

**</**Link**>**

**</**li**>**

**{**categories**.**map**((**c**)** **=>** **(**

**<**li key**={**c**}>**

**<**Link

className**={**c **===** category **?** "text-bold" **:** ""**}**

to**={**getFilterUrl**({** category**:** c **})}**

**>**

**{**c**}**

**</**Link**>**

**</**li**>**

**))}**

**</**ul**>**

**</**div**>**

**<**div**>**

**<**h3**>**Price**</**h3**>**

**<**ul**>**

**<**li**>**

**<**Link

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

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

**>**

Any

**</**Link**>**

**</**li**>**

**{**prices**.**map**((**p**)** **=>** **(**

**<**li key**={**p**.**value**}>**

**<**Link

to**={**getFilterUrl**({** price**:** p**.**value **})}**

className**={**p**.**value **===** price **?** "text-bold" **:** ""**}**

**>**

**{**p**.**name**}**

**</**Link**>**

**</**li**>**

**))}**

**</**ul**>**

**</**div**>**

**<**div**>**

**<**h3**>**Avg**.** Customer Review**</**h3**>**

**<**ul**>**

**{**ratings**.**map**((**r**)** **=>** **(**

**<**li key**={**r**.**name**}>**

**<**Link

to**={**getFilterUrl**({** rating**:** r**.**rating **})}**

className**={**`${r.rating}` **===** `${rating}` **?** "text-bold" **:** ""**}**

**>**

**<**Rating caption**={**" & up"**}** rating**={**r**.**rating**}></**Rating**>**

**</**Link**>**

**</**li**>**

**))}**

**<**li**>**

**<**Link

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

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

**>**

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

**</**Link**>**

**</**li**>**

**</**ul**>**

**</**div**>**

**</**Col**>**

**<**Col md**={**9**}>**

**{**loading **?** **(**

**<**LoadingBox**></**LoadingBox**>**

**)** **:** error **?** **(**

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

**)** **:** **(**

**<>**

**<**Row className**=**"justify-content-between mb-3"**>**

**<**Col md**={**6**}>**

**<**div**>**

**{**countProducts **===** 0 **?** "No" **:** countProducts**}** Results

**{**query **!==** "all" **&&** " : " **+** query**}**

**{**category **!==** "all" **&&** " : " **+** category**}**

**{**price **!==** "all" **&&** " : Price " **+** price**}**

**{**rating **!==** "all" **&&** " : Rating " **+** rating **+** " & up"**}**

**{**query **!==** "all" **||**

category **!==** "all" **||**

rating **!==** "all" **||**

price **!==** "all" **?** **(**

**<**Button

variant**=**"light"

onClick**={()** **=>** **navigate(**"/search"**)}**

**>**

**<**i className**=**"fas fa-times-circle"**></**i**>**

**</**Button**>**

**)** **:** **null}**

**</**div**>**

**</**Col**>**

**<**Col className**=**"text-end"**>**

Sort by**{**" "**}**

**<select**

value**={**order**}**

onChange**={(**e**)** **=>** **{**

**navigate(**getFilterUrl**({** order**:** e**.**target**.**value **}));**

**}}**

**>**

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

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

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

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

**</select>**

**</**Col**>**

**</**Row**>**

**{**products**.**length **===** 0 **&&** **(**

**<**MessageBox**>**No Product Found**</**MessageBox**>**

**)}**

**<**Row**>**

**{**products**.**map**((**product**)** **=>** **(**

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

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

**</**Col**>**

**))}**

**</**Row**>**

**<**div**>**

**{[...**Array**(**pages**).**keys**()].**map**((**x**)** **=>** **(**

**<**LinkContainer

key**={**x **+** 1**}**

className**=**"mx-1"

to**={**getFilterUrl**({** page**:** x **+** 1 **})}**

**>**

**<**Button

className**={**Number**(**page**)** **===** x **+** 1 **?** "text-bold" **:** ""**}**

variant**=**"light"

**>**

**{**x **+** 1**}**

**</**Button**>**

**</**LinkContainer**>**

**))}**

**</**div**>**

**</>**

**)}**

**</**Col**>**

**</**Row**>**

**</**div**>**

**);**

**}**

**ShippingAddressScreen.js**

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

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

**import** Form from "react-bootstrap/Form"**;**

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

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

**import** Checkout from "../components/Checkout"**;**

**const** ShippingAddressScreen **=** **()** **=>** **{**

**const** **navigate** **=** useNavigate**();**

**const** **{** state**,** dispatch**:** ctxDispatch **}** **=** useContext**(**Store**);**

**const** **{**

userInfo**,**

cart**:** **{** shippingAddress **},**

**}** **=** state**;**

**const** **[**fullName**,** setFullName**]** **=** useState**(**shippingAddress**.**fullName **||** ""**);**

**const** **[**address**,** setAddress**]** **=** useState**(**shippingAddress**.**address **||** ""**);**

**const** **[**city**,** setCity**]** **=** useState**(**shippingAddress**.**city **||** ""**);**

**const** **[**postalCode**,** setPostalCode**]** **=** useState**(**

shippingAddress**.**postalCode **||** ""

**);**

**const** **[**country**,** setCountry**]** **=** useState**(**shippingAddress**.**country **||** ""**);**

useEffect**(()** **=>** **{**

**if** **(!**userInfo**)** **{**

**navigate(**"/signin?redirect=/shipping"**);**

**}**

**},** **[**userInfo**,** **navigate]);**

**const** makeShipping **=** **(**e**)** **=>** **{**

e**.**preventDefault**();**

ctxDispatch**({**

type**:** "SAVE\_SHIPPING\_ADDRESS"**,**

payload**:** **{**

fullName**,**

address**,**

city**,**

postalCode**,**

country**,**

**location:** shippingAddress**.location,**

**},**

**});**

localStorage**.**setItem**(**

"shippingAddress"**,**

JSON**.**stringify**({**

fullName**,**

address**,**

city**,**

postalCode**,**

country**,**

**location:** shippingAddress**.location,**

**})**

**);**

**navigate(**"/payment"**);**

**};**

**return** **(**

**<**div**>**

**{**" "**}**

**<**Helmet**>**

**<**title**>**Shipping Address**</**title**>**

**</**Helmet**>**

**<**Checkout step1 step2**></**Checkout**>**

**<**div className**=**"login"**>**

**<**div className**=**"login\_container"**>**

**<**h3 className**=**"text-center"**>**Shipping Address**</**h3**>**

**<**Form onSubmit**={**makeShipping**}>**

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

**<**h6**>**Full Name**</**h6**>**

**<**input

type**=**"text"

value**={**fullName**}**

className**=**"login\_input"

onChange**={(**e**)** **=>** setFullName**(**e**.**target**.**value**)}**

required

**/>**

**</**Form**.**Group**>**

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

**<**h6**>**Address**</**h6**>**

**<**input

type**=**"text"

value**={**address**}**

className**=**"login\_input"

onChange**={(**e**)** **=>** setAddress**(**e**.**target**.**value**)}**

required

**/>**

**</**Form**.**Group**>**

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

**<**h6**>**City**</**h6**>**

**<**input

value**={**city**}**

className**=**"login\_input"

onChange**={(**e**)** **=>** setCity**(**e**.**target**.**value**)}**

required

**/>**

**</**Form**.**Group**>**

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

**<**h6**>**Postal Code**</**h6**>**

**<**input

value**={**postalCode**}**

className**=**"login\_input"

onChange**={(**e**)** **=>** setPostalCode**(**e**.**target**.**value**)}**

required

**/>**

**</**Form**.**Group**>**

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

**<**h6**>**Country**</**h6**>**

**<**input

value**={**country**}**

className**=**"login\_input"

onChange**={(**e**)** **=>** setCountry**(**e**.**target**.**value**)}**

required

**/>**

**</**Form**.**Group**>**

**<**div className**=**"mb-3"**>**

**<button** className**=**"login\_register\_button" type**=**"submit"**>**

Continue

**</button>**

**</**div**>**

**</**Form**>**

**</**div**>**

**</**div**>**

**</**div**>**

**);**

**};**

**export** **default** ShippingAddressScreen**;**

**SigninSceen.js**

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

import axios from "axios";

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

import Form from "react-bootstrap/Form";

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

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

import { toast } from "react-toastify";

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

const SigninScreen = () => {

const navigate = useNavigate();

const { search } = useLocation();

const redirectInUrl = new URLSearchParams(search).get("redirect");

const redirect = redirectInUrl ? redirectInUrl : "/";

const [email, setEmail] = useState("");

const [password, setPassword] = useState("");

const { state, dispatch: ctxDispatch } = useContext(Store);

const { userInfo } = state;

const submitHandler = async (e) => {

e.preventDefault();

try {

const { data } = await axios.post("/api/users/signin", {

email,

password,

});

ctxDispatch({ type: "USER\_SIGNIN", payload: data });

localStorage.setItem("userInfo", JSON.stringify(data));

navigate(redirect || "/");

console.log(data);

} catch (err) {

toast.error(getError(err));

}

};

// checking if user is signin already then redirct it

useEffect(() => {

if (userInfo) {

navigate(redirect);

}

}, [navigate, redirect, userInfo]);

return (

<div>

<div className="login">

<div className="login\_container">

<Helmet>

<title>Sign In</title>

</Helmet>

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

<form onSubmit={submitHandler}>

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

<h5>Email</h5>

<input

className="login\_input"

type="email"

value={email}

onChange={(e) => {

setEmail(e.target.value);

}}

/>

</Form.Group>

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

<h5>Password</h5>

<input

className="login\_input"

type="password"

value={password}

onChange={(e) => {

setPassword(e.target.value);

}}

/>

</Form.Group>

<div className="mb-3">

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

Sign in

</button>

</div>

<p>

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

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

</p>

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

New customer?{" "}

<Link

style={{ textDecoration: "none" }}

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

>

Create your account

</Link>

</div>

</form>

</div>

</div>

</div>

);

};

export default SigninScreen;

**SignupScreen.js**

**import** React from "react"**;**

**import** axios from "axios"**;**

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

**import** Form from "react-bootstrap/Form"**;**

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

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

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

**import** **{** toast **}** from "react-toastify"**;**

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

**const** SignupScreen **=** **()** **=>** **{**

**const** **navigate** **=** useNavigate**();**

**const** **{** search **}** **=** useLocation**();**

**const** redirectInUrl **=** **new** URLSearchParams**(**search**).**get**(**"redirect"**);**

**const** redirect **=** redirectInUrl **?** redirectInUrl **:** "/"**;**

**const** **[**name**,** setName**]** **=** useState**(**""**);**

**const** **[**email**,** setEmail**]** **=** useState**(**""**);**

**const** **[password,** setPassword**]** **=** useState**(**""**);**

**const** **[**confirmPassword**,** setConfirmPassword**]** **=** useState**(**""**);**

**const** **{** state**,** dispatch**:** ctxDispatch **}** **=** useContext**(**Store**);**

**const** **{** userInfo **}** **=** state**;**

**const** submitHandler **=** **async** **(**e**)** **=>** **{**

e**.**preventDefault**();**

**if** **(password** **!==** confirmPassword**)** **{**

toast**.**error**(**"Passwords do not match"**);**

**return;**

**}**

**if** **(!**name **||** **!**email **||** **!password** **||** **!**confirmPassword**)** **{**

toast**.**error**(**"Please Enter all the details properly..."**);**

**return;**

**}**

**try** **{**

**const** **{** data **}** **=** **await** axios**.**post**(**"/api/users/signup"**,** **{**

name**,**

email**,**

**password,**

**});**

ctxDispatch**({** type**:** "USER\_SIGNIN"**,** payload**:** data **});**

localStorage**.**setItem**(**"userInfo"**,** JSON**.**stringify**(**data**));**

**navigate(**redirect **||** "/"**);**

**}** **catch** **(**err**)** **{**

toast**.**error**(**getError**(**err**));**

**}**

**};**

useEffect**(()** **=>** **{**

**if** **(**userInfo**)** **{**

**navigate(**redirect**);**

**}**

**},** **[navigate,** redirect**,** userInfo**]);**

**return** **(**

**<**div className**=**"login"**>**

**<**div className**=**"login\_container"**>**

**<**Helmet**>**

**<**title**>**Sign Up**</**title**>**

**</**Helmet**>**

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

**<**Form onSubmit**={**submitHandler**}>**

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

**<**h5**>**Name**</**h5**>**

**<**input

className**=**"login\_input"

type**=**"text"

value**={**name**}**

onChange**={(**e**)** **=>** **{**

setName**(**e**.**target**.**value**);**

**}}**

**/>**

**</**Form**.**Group**>**

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

**<**h5**>**Email**</**h5**>**

**<**input

className**=**"login\_input"

type**=**"email"

value**={**email**}**

onChange**={(**e**)** **=>** **{**

setEmail**(**e**.**target**.**value**);**

**}}**

**/>**

**</**Form**.**Group**>**

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

**<**h5**>**Password**</**h5**>**

**<**input

className**=**"login\_input"

type**=**"password"

value**={password}**

onChange**={(**e**)** **=>** **{**

setPassword**(**e**.**target**.**value**);**

**}}**

**/>**

**</**Form**.**Group**>**

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

**<**h5**>**Confirm Password**</**h5**>**

**<**input

className**=**"login\_input"

type**=**"password"

value**={**confirmPassword**}**

onChange**={(**e**)** **=>** **{**

setConfirmPassword**(**e**.**target**.**value**);**

**}}**

**/>**

**</**Form**.**Group**>**

**<**div className**=**"mb-3"**>**

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

Sign Up

**</button>**

**</**div**>**

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

Already have an account**?{**" "**}**

**<**Link to**={**`/signin?redirect=${redirect}`**}>**Sign**-**In**</**Link**>**

**</**div**>**

**</**Form**>**

**</**div**>**

**</**div**>**

**);**

**};**

**export** **default** SignupScreen**;**

**Components**

**AdminRoute.js**

import React, { useContext } from "react";

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

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

const AdminRoute = ({ children }) => {

const { state } = useContext(Store);

const { userInfo } = state;

return userInfo && userInfo.isAdmin ? children : <Navigate to="/signin" />;

};

export default AdminRoute;

**Checkout.js**

**import** React from "react"**;**

**import** Row from "react-bootstrap/Row"**;**

**import** Col from "react-bootstrap/Col"**;**

**const** Checkout **=** **(**props**)** **=>** **{**

**return** **(**

**<**Row className**=**"checkout-steps"**>**

**<**Col className**={**props**.**step1 **?** "active" **:** ""**}>**Sign**-**In**</**Col**>**

**<**Col className**={**props**.**step2 **?** "active" **:** ""**}>**Shipping**</**Col**>**

**<**Col className**={**props**.**step3 **?** "active" **:** ""**}>**Payment**</**Col**>**

**<**Col className**={**props**.**step4 **?** "active" **:** ""**}>**Place Order**</**Col**>**

**</**Row**>**

**);**

**};**

**export** **default** Checkout**;**

**LoadingBox.js**

**import** Spinner from "react-bootstrap/Spinner"**;**

**export** **default** **function** LoadingBox**()** **{**

**return** **(**

**<**Spinner animation**=**"border" role**=**"status"**>**

**<**span className**=**"visually-hidden"**>**Loading**...</**span**>**

**</**Spinner**>**

**);**

**}**

**MessageBox.js**

**import** Alert from "react-bootstrap/Alert"**;**

**export** **default** **function** MessageBox**(**props**)** **{**

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

**}**

**Product.js**

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

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

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

**import** Rating from "./Rating"**;**

**import** axios from "axios"**;**

**import** **{** useContext **}** from "react"**;**

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

**function** Product**(**props**)** **{**

**const** **{** product **}** **=** props**;**

**const** **{** state**,** dispatch**:** ctxDispatch **}** **=** useContext**(**Store**);**

**const** **{**

cart**:** **{** cartItems **},**

**}** **=** state**;**

**const** addToCartHandler **=** **async** **(**item**)** **=>** **{**

**const** existItem **=** cartItems**.**find**((**x**)** **=>** x**.**\_id **===** product**.**\_id**);**

**const** quantity **=** existItem **?** existItem**.**quantity **+** 1 **:** 1**;**

**const** **{** data **}** **=** **await** axios**.**get**(**`/api/products/${item.\_id}`**);**

**if** **(**data**.**countInStock **<** quantity**)** **{**

**window.alert(**"Sorry. Product is out of stock"**);**

**return;**

**}**

ctxDispatch**({**

type**:** "CART\_ADD\_ITEM"**,**

payload**:** **{** **...**item**,** quantity **},**

**});**

**};**

**return** **(**

**<**Card**>**

**<**Link style**={{** textDecoration**:** "none" **}}** to**={**`/product/${product.slug}`**}>**

**<**img

style**={{** textDecoration**:** "none" **}}**

src**={**product**.image}**

className**=**"card-img-top images-home"

alt**={**product**.**name**}**

**/>**

**</**Link**>**

**<**Card**.**Body**>**

**<**Link

style**={{** textDecoration**:** "none" **}}**

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

**>**

**<**Card**.**Title style**={{** textDecoration**:** "none" **}}>**

**{**product**.**name**}**

**</**Card**.**Title**>**

**</**Link**>**

**<**Rating

style**={{** textDecoration**:** "none" **}}**

rating**={**product**.**rating**}**

numReviews**={**product**.**numReviews**}**

**/>**

**<**Card**.**Text**>**$**{**product**.**price**}</**Card**.**Text**>**

**{**product**.**countInStock **===** 0 **?** **(**

**<**Button variant**=**"light" disabled**>**

Out **of** stock

**</**Button**>**

**)** **:** **(**

**<button**

className**=**"login\_register\_button"

style**={{**

textDecoration**:** "none"**,**

width**:** "auto"**,**

fontSize**:** "1.3rem"**,**

**}}**

variant**=**"info"

onClick**={()** **=>** addToCartHandler**(**product**)}**

**>**

Add to cart

**</button>**

**)}**

**</**Card**.**Body**>**

**</**Card**>**

**);**

**}**

**export** **default** Product**;**

**ProtectedRoute.js**

**import** React**,** **{** useContext **}** from "react"**;**

**import** **{** Navigate **}** from "react-router-dom"**;**

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

**const** ProtectedRoute **=** **({** children **})** **=>** **{**

**const** **{** state **}** **=** useContext**(**Store**);**

**const** **{** userInfo **}** **=** state**;**

**return** userInfo **?** children **:** **<**Navigate to**=**"/signin" **/>;**

**};**

**export** **default** ProtectedRoute**;**

**Rating.js**

**function** Rating**(**props**)** **{**

**const** **{** rating**,** numReviews**,** caption **}** **=** props**;**

**return** **(**

**<**div className**=**"rating"**>**

**<**span**>**

**<**i

className**={**

rating **>=** 1

**?** "fas fa-star"

**:** rating **>=** 0.5

**?** "fas fa-star-half-alt"

**:** "far fa-star"

**}**

**/>**

**</**span**>**

**<**span**>**

**<**i

className**={**

rating **>=** 2

**?** "fas fa-star"

**:** rating **>=** 1.5

**?** "fas fa-star-half-alt"

**:** "far fa-star"

**}**

**/>**

**</**span**>**

**<**span**>**

**<**i

className**={**

rating **>=** 3

**?** "fas fa-star"

**:** rating **>=** 2.5

**?** "fas fa-star-half-alt"

**:** "far fa-star"

**}**

**/>**

**</**span**>**

**<**span**>**

**<**i

className**={**

rating **>=** 4

**?** "fas fa-star"

**:** rating **>=** 3.5

**?** "fas fa-star-half-alt"

**:** "far fa-star"

**}**

**/>**

**</**span**>**

**<**span**>**

**<**i

className**={**

rating **>=** 5

**?** "fas fa-star"

**:** rating **>=** 4.5

**?** "fas fa-star-half-alt"

**:** "far fa-star"

**}**

**/>**

**</**span**>**

**{**caption **?** **(**

**<**span**>{**caption**}</**span**>**

**)** **:** **(**

**<**span**>{**" " **+** numReviews **+** " reviews"**}</**span**>**

**)}**

**</**div**>**

**);**

**}**

**export** **default** Rating**;**

**SearchBox.js**

**import** React**,** **{** useState **}** from "react"**;**

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

**import** Form from "react-bootstrap/Form"**;**

**import** InputGroup from "react-bootstrap/InputGroup"**;**

**import** FormControl from "react-bootstrap/FormControl"**;**

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

**const** SearchBox **=** **()** **=>** **{**

**const** **navigate** **=** useNavigate**();**

**const** **[**query**,** setQuery**]** **=** useState**(**""**);**

**const** submitHandler **=** **(**e**)** **=>** **{**

e**.**preventDefault**();**

**navigate(**query **?** `/search/?query=${query}` **:** "/search"**);**

**};**

**return** **(**

**<**Form className**=**"d-flex me-auto" onSubmit**={**submitHandler**}>**

**<**InputGroup**>**

**<**FormControl

type**=**"text"

name**=**"q"

id**=**"q"

onChange**={(**e**)** **=>** setQuery**(**e**.**target**.**value**)}**

placeholder**=**"search products..."

aria**-**label**=**"Search Products"

aria**-**describedby**=**"button-search"

**></**FormControl**>**

**<**Button variant**=**"outline-primary" type**=**"submit" id**=**"button-search"**>**

**<**i className**=**"fas fa-search"**></**i**>**

**</**Button**>**

**</**InputGroup**>**

**</**Form**>**

**);**

**};**

**export** **default** SearchBox**;**

**FUTURE ENHANCEMENT**

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

**REFERENCE AND BIBLIOGRAPHY**

* <https://www.youtube.com>
* <https://stackoverflow.com>
* <https://web.learncodeonline.in>