



Inspiring Excellence

Project Title: Papier Factory

Table of Content

Sl. no	Content	Page no.
01	Introduction	03
02	Project Requirements	04
03	User Manual	05
04	Front End Development	27
05	Back End Development	50
06	Technology (Framework, Languages)	67
07	GitHub Repository Link	67
08	Individual Contributions	67

Introduction

The advent of digital platforms has revolutionized the way we engage with commerce, bridging distances and fostering accessibility. In this context, 'Papier Factory' emerges as a comprehensive e-commerce solution tailored to cater to the diverse needs of customers seeking quality stationery and reading materials. This report delves into the meticulous design and development of Papier Factory, an intuitive and feature-rich online platform aimed at enhancing the shopping experience for users across the spectrum.

Project Overview

Papier Factory embodies a multifaceted approach, seamlessly integrating modules to ensure a robust and user-friendly interface. Divided into distinct yet interconnected modules, this e-commerce website caters to various stakeholders, encompassing User Authentication and Authorization, Product Management, Seller Management, and Reviews and Ratings.

Module Highlights

Central to fostering trust and security, this module incorporates user registration, login/logout functionalities, comprehensive user profiles, authorization, access control, and essential features like password reset and recovery. Upholding user privacy and control is paramount within Papier Factory's framework.

Efficiently organized into categories, the product management module encapsulates the essence of Papier Factory, providing users with a seamless browsing experience. Detailed product pages, shopping cart integration, streamlined checkout processes, event showcases, wish lists, related product suggestions, and quick view options collectively enhance the purchasing journey.

Empowering sellers is at the core of Papier Factory's ethos. This module facilitates seamless onboarding through seller registration, product and event creation, discount code management, seller profile visibility, order confirmation, and shop updates, ensuring a symbiotic relationship between sellers and the platform.

Additional features such as sorting, filtering, messaging to sellers, discount code applications, and order tracking enhance user engagement and satisfaction.

Conclusion

Papier Factory's development underscores a commitment to combining functionality with user-centric design, culminating in an immersive and convenient online shopping experience. This report chronicles the intricacies of each module, underscoring the dedication and innovation invested in bringing the Papier Factory to fruition.

Project Requirements:

Module 1: User Authentication and Authorization

1. User Registration
2. Login/ Logout
3. User Profiles
4. Authorization and Access Control
5. Password Reset and Recovery
6. Update Profile

Module 2: Product Management

1. Product Category
2. Product Detail Page
3. Shopping Cart
4. Checkout Process
5. Events
6. Product wishlist
7. Related Product Suggestion
8. Product Quick View

Module 3: Seller Management

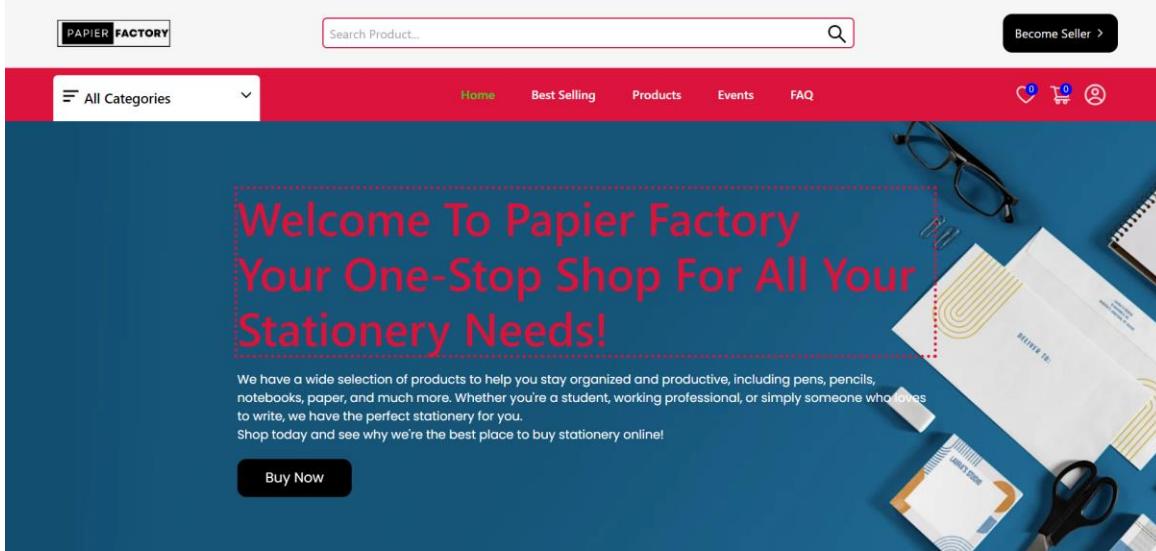
1. Seller Registration
2. Create & View Products
3. Create & View Events
4. Create Discount Code
5. View Seller Profile
6. Order Confirm
7. Update Shop

Module 4: Reviews and Ratings

1. Sort and Filter
2. Message to Seller
3. Apply Discount Code
4. Trace & Track Order

USER MANUAL

Homepage: This is the homepage view



Signup: If they're not registered yet, they can sign up by clicking the sign-up button.

Register as a new user

Full Name

Email address

Password

 (

Upload a file

Submit

Already have an account? [Sign In](#)

Sign-in: Registered Users can log in using their credentials here.

Login to your account

Email address

Password

Remember me [Forgot your password?](#)

Submit

Not have any account? [Sign Up](#)

Profile: Here one can see their profile.

- Profile
- Orders
- Inbox
- Track Order
- Change Password
- Address
- Log out

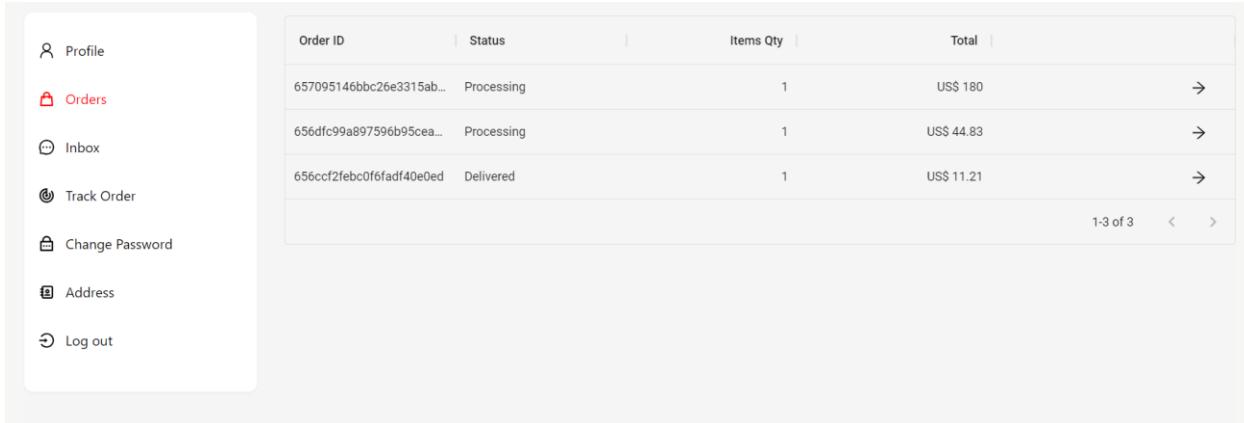
Full Name

Phone Number

Email Address

Enter your password

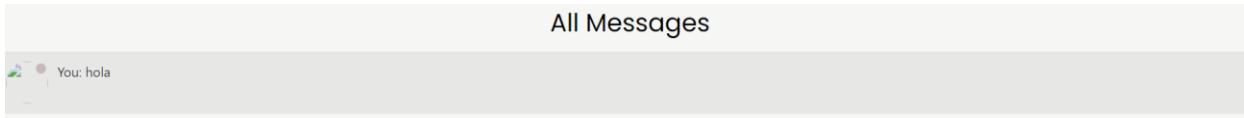
Orders: From here the customer can see the products he/she ordered.



The screenshot shows a customer dashboard with a sidebar on the left containing links: Profile, Orders (highlighted in red), Inbox, Track Order, Change Password, Address, and Log out. The main area displays a table of three orders. The columns are Order ID, Status, Items Qty, and Total. The first order is Processing, cost US\$ 180. The second is Processing, cost US\$ 44.83. The third is Delivered, cost US\$ 11.21. Navigation arrows at the bottom right indicate there are more pages.

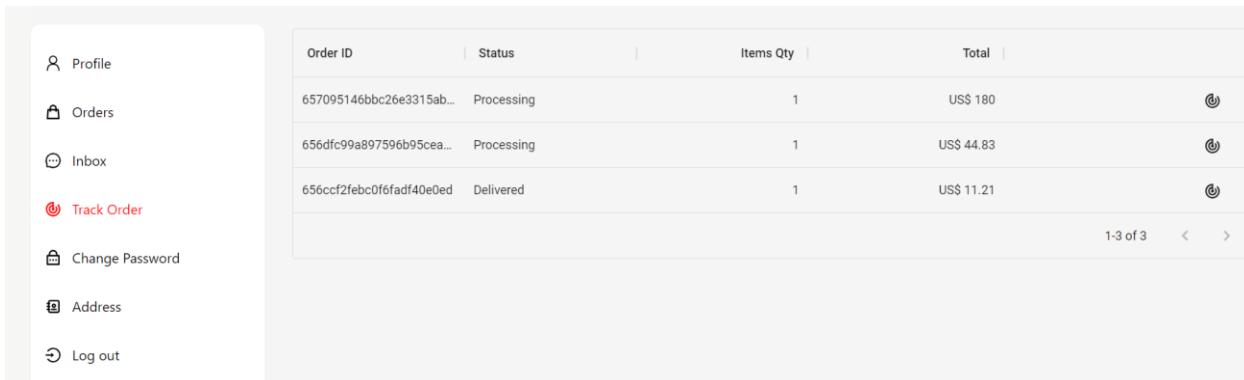
Order ID	Status	Items Qty	Total
657095146bbc26e3315ab...	Processing	1	US\$ 180
656dfc99a897596b95cea...	Processing	1	US\$ 44.83
656ccf2febcb0f6fadf40e0ed	Delivered	1	US\$ 11.21

Inbox: From here the customer can see his inbox for any messages.



The screenshot shows the customer's inbox with one message from "You" with the subject "holo". The message content is not visible.

Track Order: From here the customer can track his orders.



The screenshot shows a customer dashboard with a sidebar on the left containing links: Profile, Orders, Inbox, Track Order (highlighted in red), Change Password, Address, and Log out. The main area displays a table of three tracked orders. The columns are Order ID, Status, Items Qty, and Total. The first order is Processing, cost US\$ 180. The second is Processing, cost US\$ 44.83. The third is Delivered, cost US\$ 11.21. Each row has a circular icon with a gear symbol next to it. Navigation arrows at the bottom right indicate there are more pages.

Order ID	Status	Items Qty	Total
657095146bbc26e3315ab...	Processing	1	US\$ 180
656dfc99a897596b95cea...	Processing	1	US\$ 44.83
656ccf2febcb0f6fadf40e0ed	Delivered	1	US\$ 11.21

Change Password: The customer can change the password for his account from here.

- [Profile](#)
- [Orders](#)
- [Inbox](#)
- [Track Order](#)
- [Change Password](#)
- [Address](#)
- [Log out](#)

Change Password

Enter your old password

Enter your new password

Enter your confirm password

Update

Address: The customer can view his address

X

Add New Address

Country

choose your country ▼

Choose your City

choose your city ▼

Address 1

Address 2

Zip Code

Address Type

Choose your Address Type ▼

Best Selling: From here the customer can see the current best-selling products on this website.

The screenshot shows a grid of five product cards under the "Best Selling" section. Each card includes a thumbnail image, the brand name (Fardin), the product name, a star rating, the original price, the discounted price, and the number of sales (0 sold). The products are:

- Amazon Basics Woodcased #2 Pencils, Pre-Sharpened: 20\$ → 24.22\$
- Ticonderoga Wood-Cased Pencils, Pre-Sharpened: 16\$ → 20\$
- BIC Xtra-Smooth Mechanical Pencils With Eraser: 27.25\$ → 30\$
- Ticonderoga Wood-Cased Pencils, Unsharpened: 18.68\$ → 26.68\$
- Paper Mate Handwriting Triangular Mechanical Pencils: 10\$ → 15.5\$

Product: From here the customer can see the products which are available currently.

The screenshot shows a grid of five product cards under the "Products" section. Each card includes a thumbnail image, the brand name (Fardin), the product name, a star rating, the original price, the discounted price, and the number of sales (0 sold). The products are:

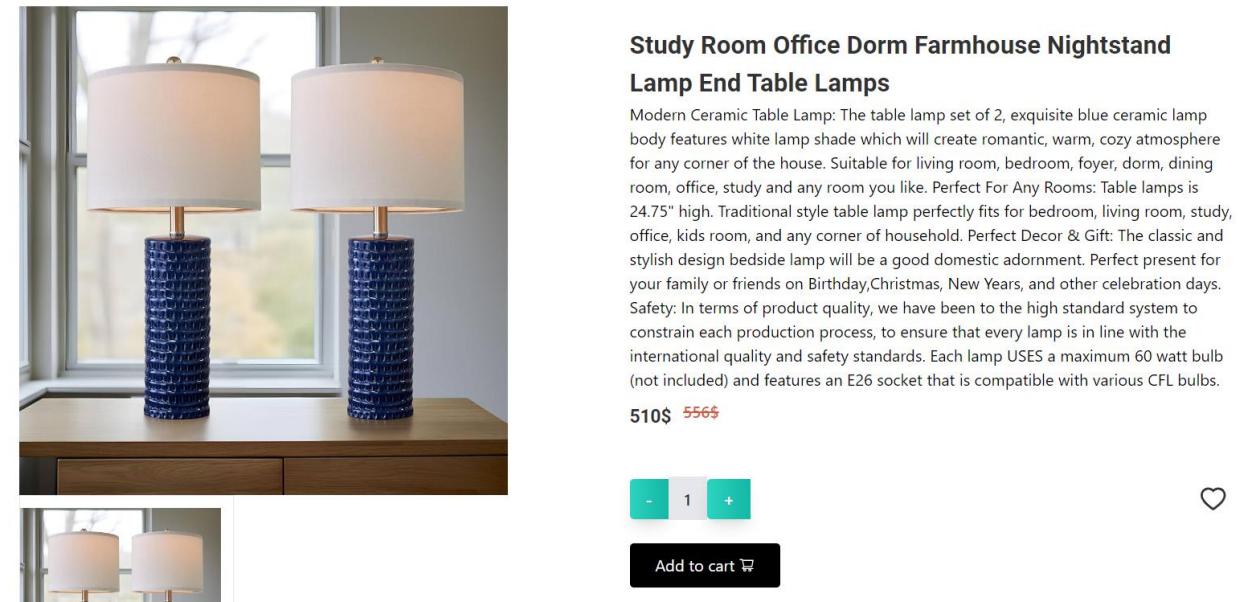
- Posca Oil and Wax Coloring Pencils Art Set: 28.89\$ → 32.6\$
- DEPESCHE TOPModel 12215 Colouring Pencils: 399\$ → 450\$
- Sharpie Permanent Marker, Fine Point, Black: 15.98\$ → 17.5\$
- Amazon Basics Fine Point Tip Permanent Markers: 26.99\$ → 31\$
- SHARPIE King Size Permanent Markers: 49\$ → 53\$

Events: Any current events that are happening right now.



The screenshot shows a product page for a pair of blue ceramic lamps. At the top, there's a red navigation bar with a dropdown menu for "All Categories", and links for "Home", "Best Selling", "Products", "Events" (which is highlighted in green), and "FAQ". To the right of the navigation are icons for a heart, a shopping cart with a zero, and a user profile. The main title is "Study Room Office Dorm Farmhouse Nightstand Lamp End Table Lamps". Below the title is a detailed product description: "Modern Ceramic Table Lamp: The table lamp set of 2, exquisite blue ceramic lamp body features white lamp shade which will create romantic, warm, cozy atmosphere for any corner of the house. Suitable for living room, bedroom, foyer, dorm, dining room, office, study and any room you like. Perfect For Any Rooms: Table lamps is 24.75" high. Traditional style table lamp perfectly fits for bedroom, living room, study, office, kids room, and any corner of household. Perfect Decor & Gift: The classic and stylish design bedside lamp will be a good domestic adornment. Perfect present for your family or friends on Birthday,Christmas, New Years, and other celebration days. Safety: In terms of product quality, we have been to the high standard system to constrain each production process, to ensure that every lamp is in line with the international quality and safety standards. Each lamp USES a maximum 60 watt bulb (not included) and features an E26 socket that is compatible with various CFL bulbs." The price is listed as \$510.00. A "Time's Up" message is displayed in red. Below the description are two buttons: "See Details" and "Add to cart".

Event Product Details:



The screenshot shows a product page for a pair of blue ceramic lamps. On the left, there's a large main image of the lamps on a wooden end table next to a window. Below it is a smaller image showing the lamps from a different angle. To the right of the images is the product title "Study Room Office Dorm Farmhouse Nightstand Lamp End Table Lamps" and its detailed description. The description is identical to the one on the previous page. The price is \$510.00. Below the description are buttons for "See Details" and "Add to cart". Further down are quantity selection buttons (-, 1, +) and a "Add to cart" button with a shopping cart icon. A heart icon for favoriting is also present.

FAQ:

The screenshot shows a website's FAQ section. At the top, there is a red header bar with a search input field containing "All Categories" and a dropdown arrow, followed by navigation links for Home, Best Selling, Products, Events, and FAQ. To the right of these are icons for a heart (0), a shopping cart (0), and a user profile.

FAQ

- What is your return policy? >
- How do I track my order? >
- How do I contact customer support? >
- Can I change or cancel my order? >
- Do you offer international shipping? >
- What payment methods do you accept? >

Support © 2023 Papier Factory
FAQ All rights reserved

PAPIER FACTORY

All Categories: Users can click categories from the shop dropdown to get a view of the categories.

The image shows a mobile application's user interface. At the top, there is a red header bar. Below it, a white card has a dark blue header with the text "All Categories" and a three-line menu icon on the left, and a downward arrow icon on the right. The card contains a list of categories, each with a small icon to its left:

- Pencil
- Ballpoint Pen
- Colouring Pencil
- Marker
- Note Book
- Sketch Book
- Art Brush
- Rubber and Sharpner
- Table Lamp
- Accesories

Search Box: From here the customer can search his desired products.

Amazon



-  Amazon Basics Woodcased #2 Pencils, Pre-sharpened, HB Lead, Box of 30
-  Amazon Basics Retractable Ballpoint Pen - Black, 1.2mm, 12-Pack
-  Crayola Colored Pencils For Adults (50 Count), Colored Pencil Set, Pair With Adult Coloring Books, Art Supplies, Holiday Gifts [Amazon Exclusive]
-  Crayola Super Tips Marker Set (120ct), Kids Washable Markers, Scented Marker Set, Holiday Gift for Kids, Bulk Markers, Thick & Thin [Amazon Exclusive]
-  Amazon Basics Fine Point Tip Permanent Markers, Black, 24-Pack
-  Amazon Basics Classic Notebook, Line Ruled, 240 Pages, Black, Hardcover, 5 x 8.25-Inch
-  Amazon Basics Sketch Pad, 5.5"x8.5", 67 lb. / 100 gsm, 100 Sheets, White
-  Amazon Basics Paint Brush Set, PBT Paint Brushes for Acrylic, Oil, Watercolor, 10 Brush Sizes
-  Amazon Basics Multi-shaped Nylon Paint Brushes for Acrylic, Oil, Watercolor, Gouache, 24 Different Sizes, Wood Color
-  Amazon Basics Portable Electric Pencil Sharpener, Helical Blade, Auto Stop, Battery/USB Cord Operated, Black, 3.54 x 3.54 x 6.3 in

Quick View: Customers can tap on a product to see the full description of the product.

The screenshot shows a product detail page for 'Crayola Colored Pencils For Adults (50 Count)'. The page includes a large image of the product, its title, price (\$6.96), and a detailed description of its features like being nontoxic and suitable for adults.

Wishlist: The customer can add their desired products to the wishlist.

Best Deals

The screenshot shows a grid of four best deal products:

- Amazon Basics Woodcased #2 Pencils, Pre... - \$20\$ 24.22\$ 0 sold
- Ticonderoga Wood-Cased Pencils, Pre-Shar... - \$16\$ 20\$ 0 sold
- BIC Xtra-Smooth Mechanical Pencils With ... - \$27.25\$ 30\$ 0 sold
- Ticonderoga Wood-Cased Pencils, Unsharpe... - \$18.68\$ 26.60\$ 0 sold

Wishlist add: After adding their products to the wishlist they can see their products in this feature.

X

♡ 4 items

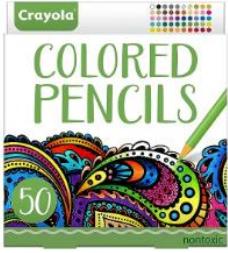
 Amazon Basics
Woodcased #2 Pencils,
Pre-sharpened, HB
Lead, Box of 30 
US\$20

 BIC Xtra-Smooth
Mechanical Pencils With
Erasers, Medium Point
(0.7mm), 10-Count Pack, 
Mechanical Pencils for
School or Office
Supplies (MPP101-BLK) 
US\$27.25

 Ticonderoga Wood-
Cased Pencils,
Unsharpened, #2 HB
Soft, Yellow, 96 Count 

US\$10.60

Product Cart: The customer can add their desired products in the cart to buy.



Fardin

Crayola Colored Pencils For Adults (50 C...)

5.0 ★★★★★

~~200₮~~ 450₮

0 - 14



Item added to cart successfully!

Crayola Crayon, 24

Item added to cart successfully!



Fardin

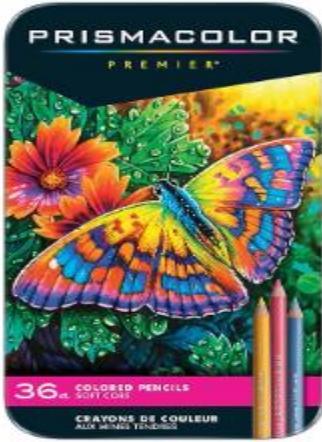
Crayola Super Tips Marker Set (120ct), K...

5.0 ★★★★★

~~26.00₮~~ 31₮

0 - 14

Product Cart Add: After adding their products to the cart, they can see their products in this feature.



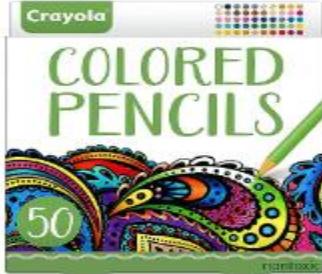
A box of Prismacolor Premier colored pencils featuring a vibrant butterfly and flowers. The box is labeled "PRISMACOLOR PREMIER" and "36 ct. COLORED PENCILS SOFT CORE".

+
1
-

Prismacolor Premier Colored Pencils, Soft Core, 36 Pack ×

\$28.89 * 1

US\$28.89



A box of Crayola colored pencils for adults, featuring a colorful abstract design. The box is labeled "Crayola COLORED PENCILS" and "50".

+
1
-

Crayola Colored Pencils For Adults (50 Count), Colored Pencil Set, Pair With Adult Coloring Books, Art Supplies, Holiday Gifts [Amazon Exclusive]

\$399 * 1

US\$399



A box of Crayola Super Tips washable markers, featuring a colorful design with cartoon characters. The box is labeled "Crayola Washable You Can Wear Super Tips" and "120 markers".

+
1
-

Crayola Super Tips Marker Set (120ct), Kids Washable Markers, Scented Marker Set, Holiday Gift for Kids, Bulk

Seller Signup: This process is for sellers who want to sell their products through this website.

Register as a seller

Shop Name

Phone Number

Email address

mr.if449@gmail.com

Address

Zip Code

Password

.....



Upload a file

Submit

Dashboard: This is the dashboard for the seller to use.

The screenshot shows the Seller Dashboard. On the left is a sidebar with icons for Dashboard, All Orders, All Products, Create Product, All Events, Create Event, Shop Inbox, Coupon Codes, and Settings. The main area has a header "Overview" with sections for "All Orders" (15 items) and "All Products" (68 items). Below this is a table titled "Latest Orders" with columns for Order ID, Status, Items Qty, Total, and a link icon. The table lists six recent orders.

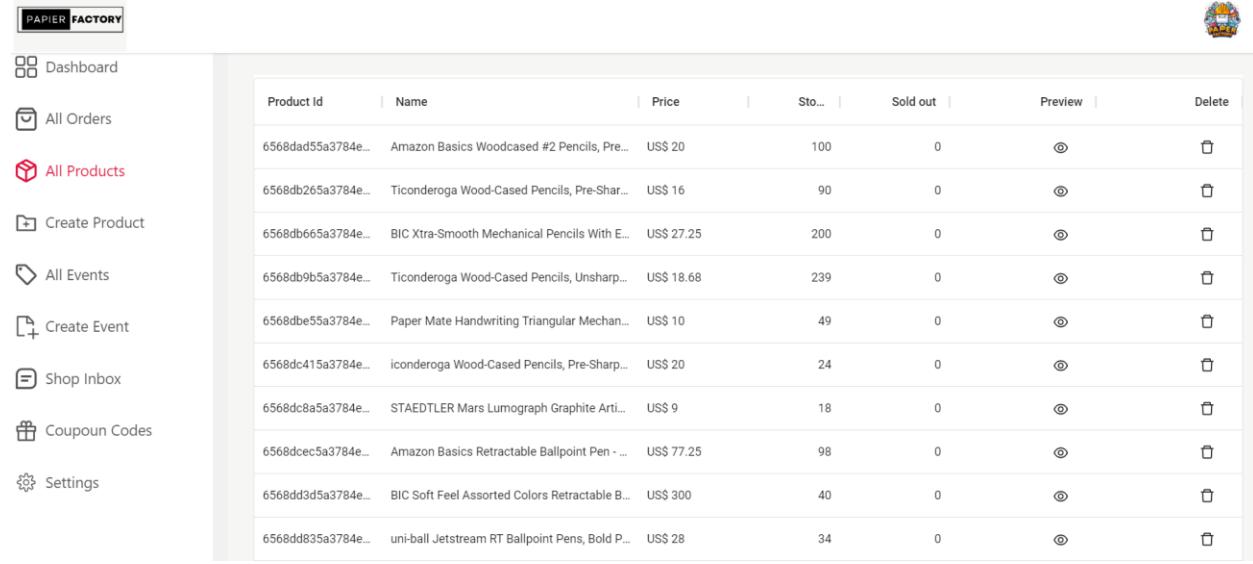
Order ID	Status	Items Qty	Total	Action
6570a96922e2c01418689df6	Processing	4	US\$ 302.33	→
657095146bbc26e3315ab0c7	Processing	1	US\$ 180	→
656dfc99a897596b95ceac8f	Processing	4	US\$ 44.83	→
656d619dd03813a802330496	Shipping	1	US\$ 16.35	→
656ccf2febc0f6fadf40e0ed	Delivered	1	US\$ 11.21	→
656b9df8ba92eaece6e98736	Delivered	1	US\$ 8.74	→

Seller All Orders: The seller can see the products that the customers ordered.

The screenshot shows the "All Orders" page. It has a sidebar with the same set of icons as the dashboard. The main area displays a table of all orders with columns for Order ID, Status, Items Qty, Total, and a link icon. The table lists ten distinct orders, each with a unique ID and status.

Order ID	Status	Items Qty	Total	Action
6570a96922e2c01418689df6	Processing	4	US\$ 302.33	→
657095146bbc26e3315ab0c7	Processing	1	US\$ 180	→
656dfc99a897596b95ceac8f	Processing	1	US\$ 44.83	→
656d619dd03813a802330496	Shipping	1	US\$ 16.35	→
656ccf2febc0f6fadf40e0ed	Delivered	1	US\$ 11.21	→
656b9df8ba92eaece6e98736	Delivered	1	US\$ 8.74	→
656b9396a6abc333301cd817		1	US\$ 22	→
656b7b835092c082ba9f894c	Delivered	4	US\$ 32.99	→
6568b88cd408d3dbfd13a856	Delivered	1	US\$ 612	→
6568bbeed408d3dbfd13a92c	Received	1	US\$ 561	→

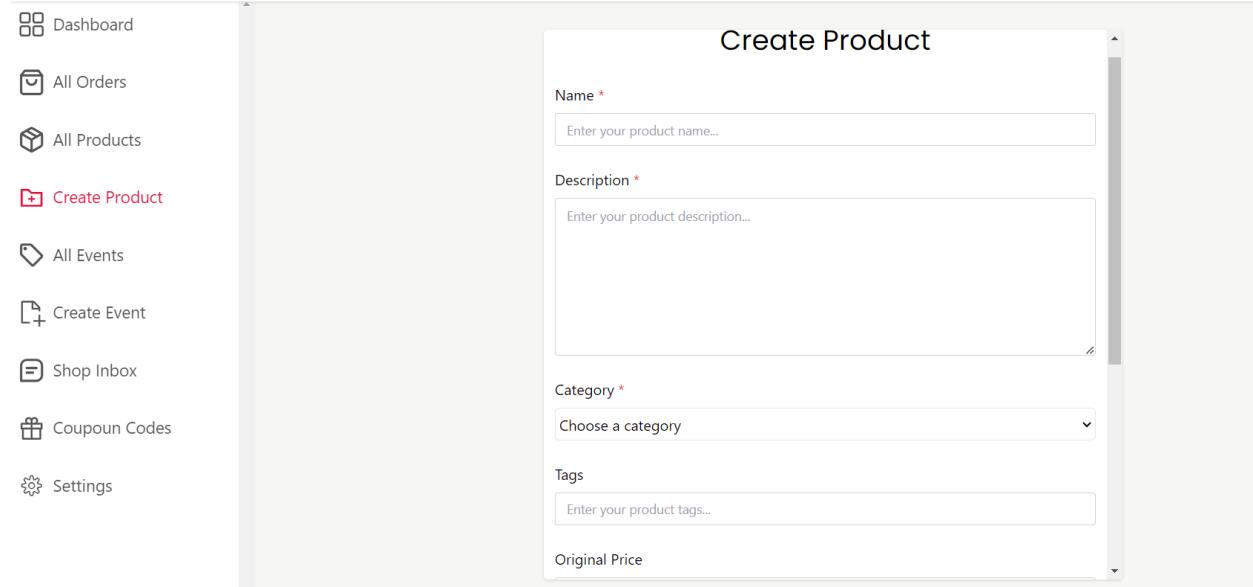
Seller All Products: The seller can see all the products he enlisted.



The screenshot shows a user interface for managing products. On the left is a sidebar with icons and labels: Dashboard, All Orders, All Products (highlighted in red), Create Product, All Events, Create Event, Shop Inbox, Coupon Codes, and Settings. The main area displays a table of products with columns: Product Id, Name, Price, Stock, Sold out, Preview, and Delete. The table contains 10 rows of product information, such as Amazon Basics Woodcased #2 Pencils and BIC Xtra-Smooth Mechanical Pencils.

Product Id	Name	Price	Stock	Sold out	Preview	Delete
6568dad55a3784e...	Amazon Basics Woodcased #2 Pencils, Pre...	US\$ 20	100	0		
6568db265a3784e...	Ticonderoga Wood-Cased Pencils, Pre-Shar...	US\$ 16	90	0		
6568db665a3784e...	BIC Xtra-Smooth Mechanical Pencils With E...	US\$ 27.25	200	0		
6568db9b5a3784e...	Ticonderoga Wood-Cased Pencils, Unsharpen...	US\$ 18.68	239	0		
6568dbe55a3784e...	Paper Mate Handwriting Triangular Mechan...	US\$ 10	49	0		
6568dc415a3784e...	Ticonderoga Wood-Cased Pencils, Pre-Sharpen...	US\$ 20	24	0		
6568dc8a5a3784e...	STAEDTLER Mars Lumograph Graphite Art...	US\$ 9	18	0		
6568dcec5a3784e...	Amazon Basics Retractable Ballpoint Pen - ...	US\$ 77.25	98	0		
6568dd3d5a3784e...	BIC Soft Feel Assorted Colors Retractable B...	US\$ 300	40	0		
6568dd835a3784e...	uni-ball Jetstream RT Ballpoint Pens, Bold P...	US\$ 28	34	0		

Seller Create Product: The Seller can create new products from this site.



The screenshot shows a 'Create Product' form. The sidebar on the left includes: Dashboard, All Orders, All Products, Create Product (highlighted in red), All Events, Create Event, Shop Inbox, Coupon Codes, and Settings. The main form has fields for Name*, Description*, Category*, Tags, and Original Price. Each field has a placeholder text indicating what should be entered.

Create Product

Name *

Enter your product name...

Description *

Enter your product description...

Category *

Choose a category

Tags

Enter your product tags...

Original Price

Seller All Events: The seller can see the events that other sellers are giving.

The screenshot shows a sidebar menu on the left with options: Dashboard, All Orders, All Products, Create Product, All Events (highlighted in red), and Create Event. The main area displays a table titled 'Product Id' with one row: 656831927979152... Study Room Office Dorm Farmhouse Night... US\$ 510. The table includes columns for Name, Price, Stock, Sold out, Preview, and Delete. A footer at the bottom right says '1-1 of 1'.

Seller Create Event: The seller can create events for his products.

The screenshot shows a sidebar menu on the left with options: Dashboard, All Orders, All Products, Create Product, All Events, Create Event (highlighted in red), Shop Inbox, and Coupon Codes. The main area is a 'Create Event' form with fields for Name*, Description*, and Category*. There is a placeholder text 'Enter your event product name...' in the Name field and 'Enter your event product description...' in the Description field. A 'Choose a category...' button is at the bottom.

Seller Shop Inbox: The seller can see the messages from customers that was sent to the seller to know a particular item.

The screenshot shows a sidebar menu on the left with options: Dashboard, All Orders, All Products, Create Product, All Events, Create Event, Shop Inbox (highlighted in red), Coupon Codes, and Settings. The main area is titled 'All Messages' and lists two messages from a customer named 'You'. The first message is 'You: hey paper factory!' and the second is 'You: sfdgsf'. Below these, there are two more messages: 'You: hi' and 'You: hello'.

Seller Coupon Codes: The seller can see the coupon codes from here.

The screenshot shows a sidebar with various shop management options: Dashboard, All Orders, All Products, Create Product, All Events, Create Event, Shop Inbox, and Coupoun Codes (highlighted in red). The main area displays a table for coupon codes. The table has columns for Id, Coupon Code, Value, and Delete. One row is shown: Id 65633ab7dba099d9daeea2b9, Coupon Code Free, Value 50 %, and a Delete button. Navigation buttons at the bottom right indicate 1-1 of 1.

Id	Coupon Code	Value	Delete
65633ab7dba099d9daeea2b9	Free	50 %	

Settings: From here the seller change his shop name or others.

The screenshot shows a sidebar with the same set of shop management options as the previous page. The main area features a logo for "PAPIER FACTORY" and a "Settings" button. To the right, there are several input fields for shop details: Shop Name (RGFT), Shop description (dfdsafadsfadsf), Shop Address (Dhaka, Bangladesh), Shop Phone Number (34324), and Shop Zip Code (1200). A large "Update Shop" button is at the bottom.

Seller Create Coupon: The seller can create new coupons for his products

X

Create Coupon code

Name *

Discount Percentenge *

Min Amount

Max Amount

Selected Product

Create

All Orders: The seller can see the orders for his products.

 Order Details

Order ID: #65709514 Placed on: 2023-12-06

 BIC Soft Feel Assorted Colors Retractable Ballpoint Pens, Medium Point (1.0mm), 36-Count Pack, Black and Blue Pens With Soft-Touch Comfort Grip, Perfect Color Pens For Note Taking
US\$300 x 1

Total Price: **US\$180**

Shipping Address:
234234 2342343
BD
14

Payment Info:
Status: Not Paid

[Send Message](#)

Order Status: From here, the seller can update the order status

Shipping Address:

Mohammadpur Tajmahal Road
BD
C

Payment Info:
Status: Not Paid

Order Status:

Processing ▾

[Update Status](#)

Checkout: This is the part where the customers can see and edit their information for the delivery.

1.Shipping 2.Payment 3.Success

Shipping Address

Full Name	Email Address
Rahadul Islam	mr.if449@gmail.com
Phone Number	Zip Code
Country	City
Address1	Address2

Choose From saved address

subtotal: \$503.88

shipping: \$50.39

Discount: -

\$554.27

Coupon code

[Apply code](#)

Proceed to Payment: Here the customers have to add their payment method to pay for their desired products.

1.Shipping 2.Payment 3.Success

Pay with Debit/credit card

Name On Card: Rahadul Islam Exp Date: MM / YY

Card Number: 1234 1234 1234 1234 CVV: CVC

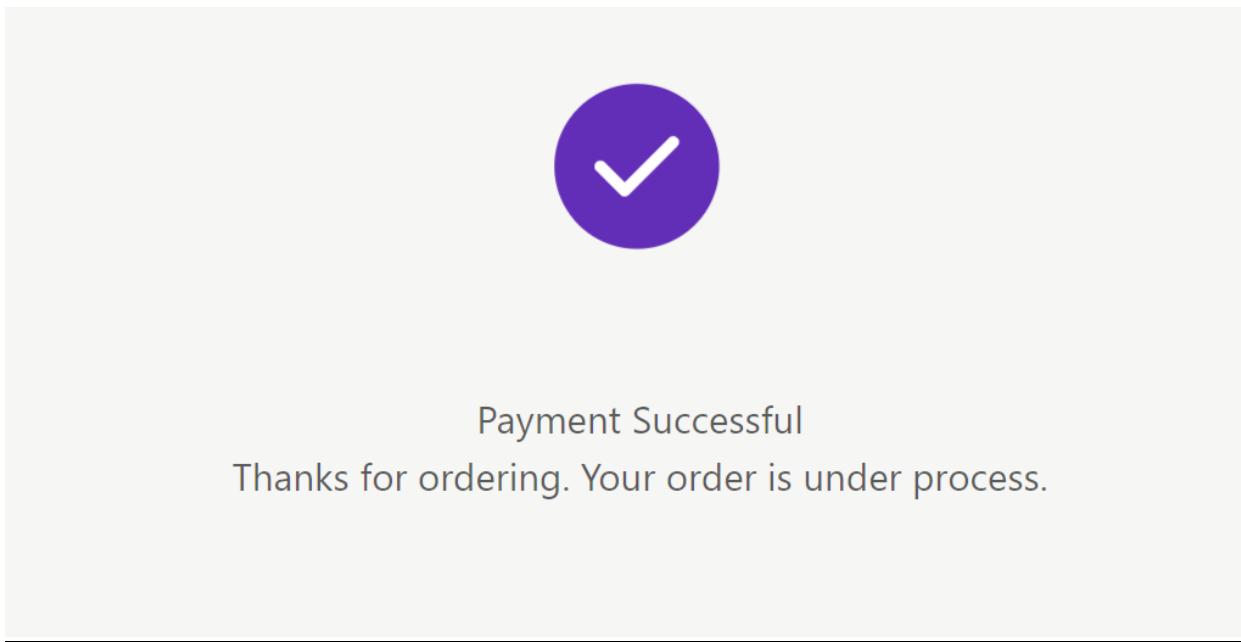
Submit

Pay with Bkash

Cash on Delivery

subtotal:	\$503.88
shipping:	\$50.39
Discount:	\$251.94
	\$302.33

Confirm: They will be shown this page if their payment is successful and their order will be confirmed.



CODE SNIPPETS:

Front end App.js

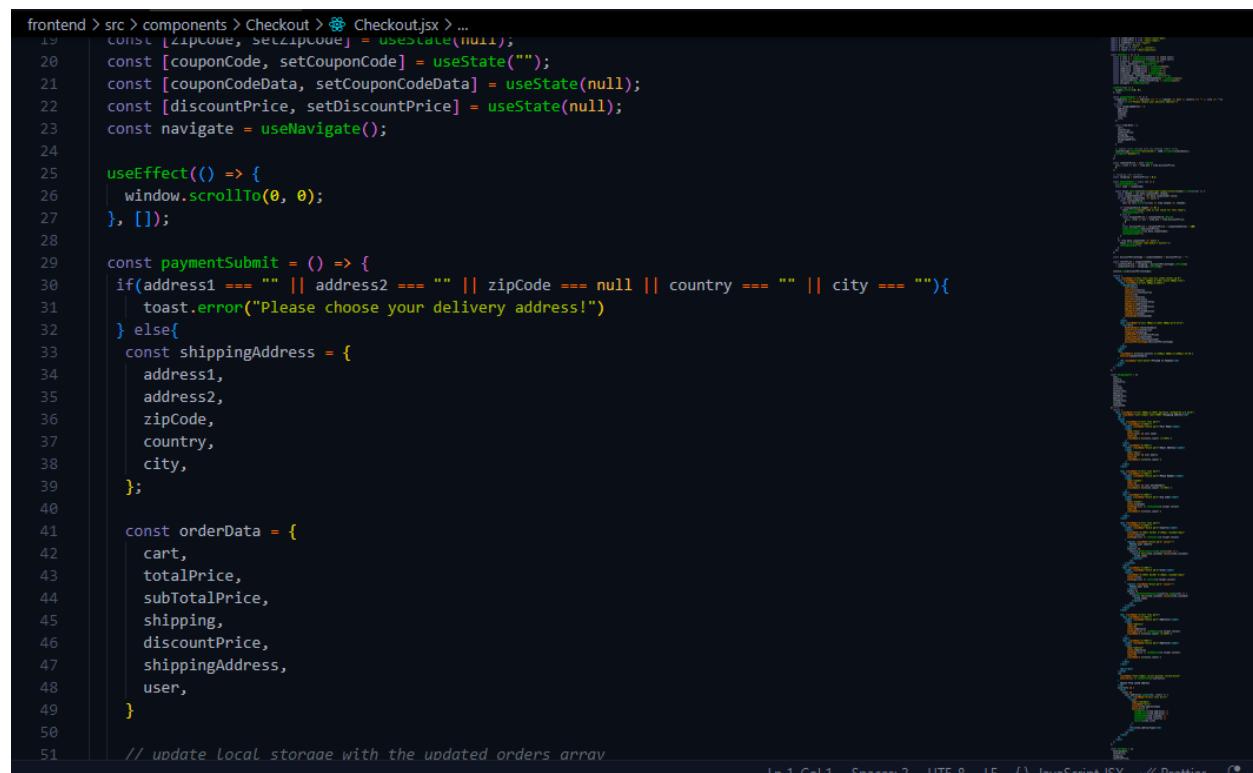
```
frontend > src > App.js > ...
38 |     ShopInboxPage
39 |   } from "./routes/ShopRoutes";
40 |   import { ToastContainer } from "react-toastify";
41 |   import "react-toastify/dist/ReactToastify.css";
42 |   import Store from "./redux/store";
43 |   import { loadSeller, loadUser } from "./redux/actions/user";
44 |   import ProtectedRoute from "./routes/ProtectedRoute";
45 |   import { ShopHomePage } from "./ShopRoutes.js";
46 |   import SellerProtectedRoute from "./routes/SellerProtectedRoute";
47 |   import { getAllProducts } from "./redux/actions/product";
48 |   import { getAllEvents } from "./redux/actions/event";
49 |   import axios from "axios";
50 |   import { server } from "./server";
51 |   import { Elements } from "@stripe/react-stripe-js";
52 |   import { loadStripe } from "@stripe/stripe-js";
53 |
54 | const App = () => {
55 |   const [stripeApiKey, setStripeApiKey] = useState("");
56 |
57 |   async function getStripeApiKey() {
58 |     const { data } = await axios.get(`${server}/payment/stripeapikey`);
59 |     setStripeApiKey(data.stripeApiKey);
60 |   }
61 |   useEffect(() => {
62 |     Store.dispatch(loadUser());
63 |     Store.dispatch(loadSeller());
64 |     Store.dispatch(getAllProducts());
65 |     Store.dispatch(getAllEvents());
66 |     getStripeApiKey();
67 |   }, []);
68 |
69 |   return (

```

Frontend component codes

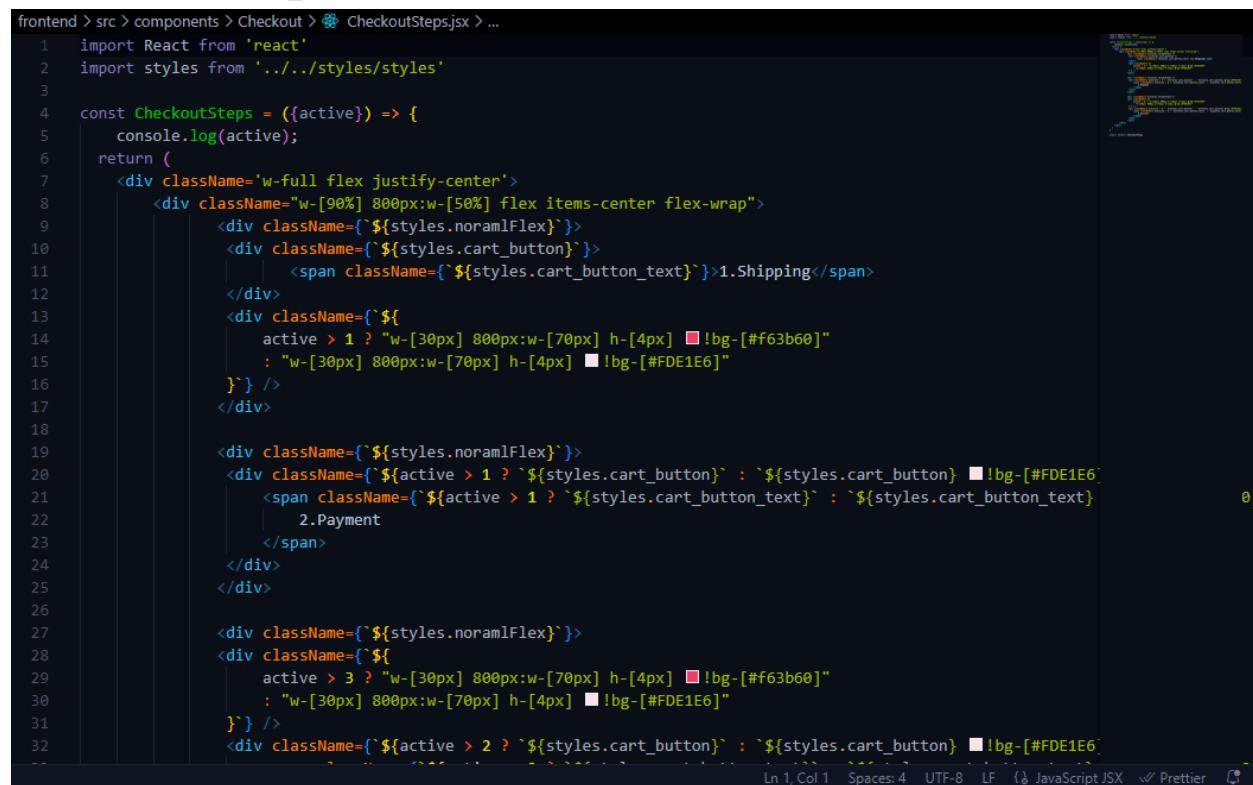
```
frontend > src > components > cart > Cart.jsx > ...
1 import React, { useState } from "react";
2 import { RxCross1 } from "react-icons/rx";
3 import { IoBagHandleOutline } from "react-icons/io5";
4 import { HiOutlineMinus, HiPlus } from "react-icons/hi";
5 import styles from "../../styles/styles";
6 import { Link } from "react-router-dom";
7 import { backend_url } from "../../server";
8 import { useDispatch, useSelector } from "react-redux";
9 import { addTocart, removeFromCart } from "../../redux/actions/cart";
10 import { toast } from "react-toastify";
11
12 const Cart = ({ setOpenCart }) => {
13   const { cart } = useSelector((state) => state.cart);
14   const dispatch = useDispatch();
15
16   const removeFromCartHandler = (data) => {
17     dispatch(removeFromCart(data));
18   };
19
20   const totalPrice = cart.reduce(
21     (acc, item) => acc + item.qty * item.discountPrice,
22     0
23   );
24
25   const quantityChangeHandler = (data) => {
26     dispatch(addTocart(data));
27   };
28
29   return (
30     <div className="fixed top-0 left-0 w-full bg-[#0000004b] h-screen z-10">
31       <div className="fixed top-0 right-0 h-full w-[25%] bg-white flex flex-col overflow-y-scroll justify-between">
32         {cart && cart.length === 0 ? (
33           ...
```

checkout.jsx



```
frontend > src > components > Checkout > Checkout.jsx > ...
19  const [zipCode, setZipCode] = useState(null);
20  const [couponCode, setCouponCode] = useState("");
21  const [couponCodeData, setCouponCodeData] = useState(null);
22  const [discountPrice, setDiscountPrice] = useState(null);
23  const navigate = useNavigate();
24
25  useEffect(() => {
26    window.scrollTo(0, 0);
27  }, []);
28
29  const paymentSubmit = () => {
30    if(address1 === "" || address2 === "" || zipCode === null || country === "" || city === ""){
31      toast.error("Please choose your delivery address!")
32    } else{
33      const shippingAddress = {
34        address1,
35        address2,
36        zipCode,
37        country,
38        city,
39      };
40
41      const orderData = {
42        cart,
43        totalPrice,
44        subTotalPrice,
45        shipping,
46        discountPrice,
47        shippingAddress,
48        user,
49      }
50
51      // update local storage with the updated orders array
52    }
53  }
54
```

checkoutSteps.jsx



```
frontend > src > components > Checkout > CheckoutSteps.jsx > ...
1  import React from 'react'
2  import styles from '../../../../../styles/styles'
3
4  const CheckoutSteps = ({active}) => {
5    console.log(active);
6    return (
7      <div className='w-full flex justify-center'>
8        <div className='w-[90%] 800px:w-[50%] flex items-center flex-wrap'>
9          <div className={`${styles.normalFlex}`}>
10            <div className={`${styles.cart_button}`}>
11              <span className={`${styles.cart_button_text}`}>1.Shipping</span>
12            </div>
13            <div className={`${` ${active > 1 ? "w-[30px] 800px:w-[70px] h-[4px] !bg-[#f63b60" : "w-[30px] 800px:w-[70px] h-[4px] !bg-[#FDE1E6]`}`}`}>
14            </div>
15          </div>
16        <div className={`${styles.normalFlex}`}>
17          <div className={`${` ${active > 1 ? `${styles.cart_button}` : `${styles.cart_button} !bg-[#FDE1E6]`}`}`}>
18            <span className={`${` ${active > 1 ? `${styles.cart_button_text}` : `${styles.cart_button_text}`}`}`}>2.Payment</span>
19          </div>
20        </div>
21
22        <div className={`${styles.normalFlex}`}>
23          <div className={`${` ${active > 3 ? "w-[30px] 800px:w-[70px] h-[4px] !bg-[#f63b60" : "w-[30px] 800px:w-[70px] h-[4px] !bg-[#FDE1E6]`}`}`}>
24            </div>
25        </div>
26      </div>
27    )
28  }
29
```

countdown.jsx

```
frontend > src > components > Events > CountDown.jsx ...
1 import React, { useEffect, useState } from "react";
2
3 const CountDown = ({data}) => {
4   const [timeLeft, setTimeLeft] = useState(calculateTimeLeft());
5
6   useEffect(() => {
7     const timer = setTimeout(() => {
8       setTimeLeft(calculateTimeLeft());
9     }, 1000);
10    return () => clearTimeout(timer);
11  });
12
13   function calculateTimeLeft() {
14     const difference = +new Date(data.Finish_Date) - +new Date();
15     let timeLeft = {};
16
17     if (difference > 0) {
18       timeLeft = {
19         days: Math.floor(difference / (1000 * 60 * 60 * 24)),
20         hours: Math.floor((difference / (1000 * 60 * 60)) % 24),
21         minutes: Math.floor((difference / 1000 / 60) % 60),
22         seconds: Math.floor((difference / 1000) % 60),
23       };
24     }
25
26     return timeLeft;
27   }
28
29   const timerComponents = Object.keys(timeLeft).map((interval) => {
30     if (!timeLeft[interval]) {
31       return null;
32     }
33   })
34 }
```

Ln 1 Col 1 | Spaces: 2 | UTF-8 | LF | $\frac{1}{3}$ JavaScript JSX // Prettier

eventcard.jsx

```
frontend > src > components > Events > EventCard.jsx ...
1 import React from "react";
2 import { backend_url } from "../../server";
3 import styles from "../../styles/styles";
4 import CountDown from "./CountDown";
5 import { Link } from "react-router-dom";
6 import { useDispatch, useSelector } from "react-redux";
7 import { addTocart } from "../../redux/actions/cart";
8 import { toast } from "react-toastify";
9
10 const EventCard = ({ active, data }) => {
11   const { cart } = useSelector((state) => state.cart);
12   const dispatch = useDispatch();
13
14   const addToCartHandler = (data) => {
15     const isItemExists = cart && cart.find((i) => i._id === data._id);
16     if (isItemExists) {
17       toast.error("Item already in cart!");
18     } else {
19       if (data.stock < 1) {
20         toast.error("Product stock limited!");
21       } else {
22         const cartData = { ...data, qty: 1 };
23         dispatch(addTocart(cartData));
24         toast.success("Item added to cart successfully!");
25       }
26     }
27   }
28
29   return (
30     <div
31       className={`${w-full block bg-white rounded-lg ${
32       active ? "unset" : "mb-12"
33     } lg:flex p-2`}
34   
```

Ln 1 Col 1 | Spaces: 2 | UTF-8 | LF | $\frac{1}{3}$ JavaScript JSX // Prettier

event.jsx

```
frontend > src > components > Events > Events.jsx > ...
1 import React, { useEffect } from 'react'
2 import { useSelector } from 'react-redux';
3 import styles from '../../../../../styles/styles'
4 import EventCard from "./EventCard";
5
6 const Events = () => {
7   const {allEvents, isLoading} = useSelector((state) => state.events);
8
9   return (
10     <div>
11       {
12         !isLoading && (
13           <div className={`${styles.section}`}>
14             <div className={`${styles.heading}`}>
15               <h1>Popular Events</h1>
16             </div>
17
18             <div className="w-full grid">
19               <EventCard data={allEvents && allEvents[0]} />
20             </div>
21
22           )
23         }
24       )
25     </div>
26   )
27 }
28
29 export default Events
```

dropdown.jsx

```
frontend > src > components > Layout > DropDown.jsx > ...
4
5 const DropDown = ({ categoriesData, setDropDown }) => {
6   const navigate = useNavigate();
7   const submitHandle = (i) => {
8     navigate(`/products?category=${i.title}`);
9     setDropDown(false);
10    window.location.reload();
11  };
12  return (
13    <div className="pb-4 w-[270px] bg-[#fff] absolute z-30 rounded-b-md shadow-sm">
14      {categoriesData &&
15        categoriesData.map((i, index) => (
16          <div
17            key={index}
18            className={`${styles.noMarginFlex}`}
19            onClick={() => submitHandle(i)}
20          >
21            <img
22              src={i.imageUrl}
23              style={{
24                width: "25px",
25                height: "25px",
26                objectFit: "contain",
27                marginLeft: "10px",
28                userSelect: "none",
29              }}
30              alt=""
31            />
32            <h3 className="m-3 cursor-pointer select-none">{i.title}</h3>
33          </div>
34        )));
35      </div>
```

In 1 Col 1 Spaces: 2 LITE:8 LF:13 JavaScript/JSX v2 Prettier

Footer.jsx

```
4
5  const Footer = () => {
6    return (
7      <footer className="bg-[#000] text-white">
8        <div className="container mx-auto grid grid-cols-1 sm:grid-cols-3 lg:grid-cols-3 gap-6 sm:px-8 px-5 py-16">
9          {/* Company Information */}
10         <div className="col-span-1 sm:col-span-1">
11           <img
12             src={require("../images/logo2.png")}
13             height={108}
14             width={192}
15             alt="Logo"
16             className="mb-6"
17           />
18           <p className="text-sm text-gray-400">
19             The home and elements needed to create beautiful products.
20           </p>
21         </div>
22
23         {/* Support Links */}
24         <div className="col-span-1 sm:col-span-1">
25           <h1 className="mb-6 font-semibold">Support</h1>
26           <ul>
27             {footerSupportLinks.map((link, index) => (
28               <li key={index} className="mb-3">
29                 <Link
30                   to={link.link}
31                   className="text-gray-400 hover:text-teal-400 duration-300 text-sm leading-6"
32                 >
33                   {link.name}
34                 </Link>
35               </li>
36             ))}
37           </ul>
38         </div>
39       </div>
40     </footer>
41   );
42 }
43
44 export default Footer;
```

In 1 Col 1 Spaces: 2 LITE-R LF {λ JavaScript JSX ✓ Prettier ⌂

header.jsx

```
33
34  const [open, setOpen] = useState(false);
35
36  const handleSearchChange = (e) => {
37    const term = e.target.value;
38    setSearchTerm(term);
39
40    const filteredProducts = allProducts && allProducts.filter((product) =>
41      product.name.toLowerCase().includes(term.toLowerCase())
42    );
43    setData(filteredProducts);
44  };
45
46  window.addEventListener("scroll", () => {
47    if (window.scrollY > 70) {
48      setActive(true);
49    } else {
50      setActive(false);
51    }
52  });
53
54  return (
55    <>
56      <div className={`${styles.section}`}>
57        <div className="hidden 800px:h-[50px] 800px:my-[20px] 800px:flex items-center justify-between">
58          <div>
59            <Link to="/">
60              <img
61                src={require("../images/logo.png")}
62                alt=""
63                height={280}
64                width={157.5}
65              />
66            </Link>
67          </div>
68        </div>
69      </div>
70    </>
71  );
72}
```

In 1 Col 1 Spaces: 2 LITE-R LF {λ JavaScript JSX ✓ Prettier ⌂

loader.jsx

```
frontend > src > components > Layout > Loader.jsx > ...
1  import React from "react";
2  import Lottie from "react-lottie";
3
4  const Loader = () => {
5      const defaultOptions = {
6          loop: true,
7          autoplay: true,
8          rendererSettings: {
9              preserveAspectRatio: "xMidYMid slice",
10         },
11     };
12     return (
13         <div className="w-full h-screen flex items-center justify-center">
14             <Lottie options={defaultOptions} width={300} height={300} />
15         </div>
16     );
17 };
18
19 export default Loader;
20
```

navbar.jsx

```
frontend > src > components > Layout > Navbar.jsx > ...
● 1  import React from 'react'
2  import { Link } from 'react-router-dom'
3  import { navItems } from '../../../../../static/data'
4  import styles from '../../../../../styles/styles'
5
6  const Navbar = ({active}) => {
7      return (
8          <div className={`block 800px:${styles.normalFlex}`}>
9              {
10                  navItems && navItems.map((i,index) => (
11                      <div className="flex">
12                          <Link to={i.url}>
13                              className={`${active === index + 1 ? "text-[#17dd1f]" : "text-black"} 800px:text-[#fff]`}
14                          >
15                              {i.title}
16                          </Link>
17                      </div>
18                  ))
19              }
20          </div>
21      )
22  }
23
24 export default Navbar
```

login.jsx

```
9  const Login = () => {
10    const navigate = useNavigate();
11    const [email, setEmail] = useState("");
12    const [password, setPassword] = useState("");
13    const [visible, setVisible] = useState(false);
14
15    const handleSubmit = async (e) => {
16      e.preventDefault();
17
18      await axios
19        .post(
20          `${server}/user/login-user`,
21          {
22            email,
23            password,
24          },
25          { withCredentials: true }
26        )
27        .then((res) => {
28          toast.success("Login Success!");
29          navigate("/");
30          window.location.reload(true);
31        })
32        .catch((err) => {
33          toast.error(err.response.data.message);
34        });
35    };
36
37    return (
38      <div className="min-h-screen bg-gray-50 flex flex-col justify-center py-12 sm:px-6 lg:px-8">
39        <div className="sm:mx-auto sm:w-full sm:max-w-md">
40          <h2 className="mt-6 text-center text-3xl font-extrabold text-gray-900">
```

payment.jsx

```
83      const client_secret = data.client_secret;
84
85      if (!stripe || !elements) return;
86      const result = await stripe.confirmCardPayment(client_secret, {
87        payment_method: {
88          card: elements.getElement(CardNumberElement),
89        },
90      });
91
92      if (result.error) {
93        toast.error(result.error.message);
94      } else {
95        if (result.paymentIntent.status === "succeeded") {
96          order.paymentInfo = {
97            id: result.paymentIntent.id,
98            status: result.paymentIntent.status,
99            type: "Credit Card",
100          };
101
102          await axios
103            .post(`${server}/order/create-order`, order, config)
104            .then((res) => {
105              setOpen(false);
106              navigate("/order/success");
107              toast.success("Order successful!");
108              localStorage.setItem("cartItems", JSON.stringify([]));
109              localStorage.setItem("latestOrder", JSON.stringify([ ]));
110              window.location.reload();
111            });
112        }
113      }
114    } catch (error) {
```

productDetails.jsx

```
75
76 const totalReviewsLength =
77   products &&
78   products.reduce((acc, product) => acc + product.reviews.length, 0);
79
80 const totalRatings =
81   products &&
82   products.reduce(
83     (acc, product) =>
84       acc + product.reviews.reduce((sum, review) => sum + review.rating, 0),
85     0
86   );
87
88 const averageRating = totalRatings / totalReviewsLength || 0;
89
90 const handleMessageSubmit = async () => {
91   if (isAuthenticated) {
92     const groupTitle = data._id + user._id;
93     const userId = user._id;
94     const sellerId = data.shop._id;
95     await axios
96       .post(`${server}/conversation/create-new-conversation`, {
97         groupTitle,
98         userId,
99         sellerId,
100      })
101      .then((res) => {
102        navigate(`'/inbox?${res.data.conversation._id}`);
103      })
104      .catch((error) => {
105        toast.error(error.response.data.message);
106      });
107    }
108  }
109
```

Ratings.jsx

```
6 const stars = [];
7
8 for (let i = 1; i <= 5; i++) {
9   if (i <= rating) {
10     stars.push(
11       <AiFillStar
12         key={i}
13         size={20}
14         color="#f6b100"
15         className="mr-2 cursor-pointer"
16       />
17     );
18   } else if (i === Math.ceil(rating) && !Number.isInteger(rating)) {
19     stars.push(
20       <BsStarHalf
21         key={i}
22         size={17}
23         color="#f6ba00"
24         className="mr-2 cursor-pointer"
25       />
26     );
27   } else {
28     stars.push(
29       <AiOutlineStar
30         key={i}
31         size={20}
32         color="#f6ba00"
33         className="mr-2 cursor-pointer"
34       />
35     );
36   }
37 }
38 return (
39   <div style={{display: "flex", gap: "10px", align-items: "center", margin: "0 auto", width: "fit-content", padding: "0 10px", border: "1px solid #f6b100", border-radius: "10px", background: "#fff", position: "relative", height: "40px", overflow: "hidden"}>
40     {stars}
41   </div>
42 );
43 
```

SuggestedProduct.jsx

```
6  const SuggestedProduct = ({ data }) => {
7    const [allProducts] = useSelector((state) => state.products);
8    const [productData, setProductData] = useState();
9
10   useEffect(() => {
11     const d =
12       allProducts && allProducts.filter((i) => i.category === data.category);
13     setProductData(d);
14   }, []);
15
16   return (
17     <div>
18       {data ? (
19         <div className={`${p-4 ${styles.section}`}>
20           <h2 className={`${${styles.heading} text-[25px] font-[500] border-b mb-5`}>
21             Related Product
22           </h2>
23           <div className="grid grid-cols-1 gap-[20px] md:grid-cols-2 md:gap-[25px] lg:grid-cols-4 lg:gap-[25px]>
24             {
25               productData && productData.map((i, index) => (
26                 <ProductCard data={i} key={index} />
27               ))
28             }
29           </div>
30         </div>
31       ) : null}
32     </div>
33   );
34 );
35 );
36 );
37 );
38 );
```

profileContent.jsx

```
const ProfileContent = ({ active }) => {
  const [ user, error, successMessage ] = useSelector((state) => state.user);
  const [name, setName] = useState(user && user.name);
  const [email, setEmail] = useState(user && user.email);
  const [phoneNumber, setPhoneNumber] = useState(user && user.phoneNumber);
  const [password, setPassword] = useState("");
  const [avatar, setAvatar] = useState(null);
  const dispatch = useDispatch();

  useEffect(() => {
    if (error) {
      toast.error(error);
      dispatch({ type: "clearErrors" });
    }
    if (successMessage) {
      toast.success(successMessage);
      dispatch({ type: "clearMessages" });
    }
  }, [error, successMessage]);

  const handleSubmit = (e) => {
    e.preventDefault();
    dispatch(updateUserInformation(name, email, phoneNumber, password));
  };

  const handleImage = async (e) => {
    const file = e.target.files[0];
    setAvatar(file);
  };

  const formData = new FormData();
```

profileSidebar.jsx

```
const ProfileSidebar = ({ setActive, active }) => {
  const navigate = useNavigate();

  const logoutHandler = () => {
    axios
      .get(`/${server}/user/logout`, { withCredentials: true })
      .then((res) => {
        toast.success(res.data.message);
        window.location.reload(true);
        navigate("/login");
      })
      .catch((error) => {
        console.log(error.response.data.message);
      });
  };

  return (
    <div className="w-full bg-white shadow-sm rounded-[10px] p-4 pt-8">
      <div
        className="flex items-center cursor-pointer w-full mb-8"
        onClick={() => setActive(1)}
      >
        <RxPerson size={20} color={active === 1 ? "red" : ""} />
        <span className="pl-3 ${active === 1 ? "text-[red]" : ""} 800px:block hidden">
          Profile
        </span>
      </div>
      <div
        className="flex items-center cursor-pointer w-full mb-8"
        onClick={() => setActive(2)}
      >
        <RxLogout size={20} color="text-[red]" />
        <span>Logout</span>
      </div>
    </div>
  );
}
```



trackorder.jsx

```
frontend > src > components > Profile > TrackOrder.jsx > ...
17  const data = orders && orders.find((item) => item._id === id);
18
19  return (
20    <div className="w-full h-[80vh] flex justify-center items-center">
21      {" "}
22      <>
23        {data && data?.status === "Processing" ? (
24          <h1 className="text-[20px]">Your Order is processing in shop.</h1>
25        ) : data?.status === "Transferred to delivery partner" ? (
26          <h1 className="text-[20px]">
27            Your Order is on the way for delivery partner.
28          </h1>
29        ) : data?.status === "Shipping" ? (
30          <h1 className="text-[20px]">
31            Your Order is on the way with our delivery partner.
32          </h1>
33        ) : data?.status === "Received" ? (
34          <h1 className="text-[20px]">
35            Your Order is in your city. Our Delivery man will deliver it.
36          </h1>
37        ) : data?.status === "On the way" ? (
38          <h1 className="text-[20px]">
39            Our Delivery man is going to deliver your order.
40          </h1>
41        ) : data?.status === "Delivered" ? (
42          <h1 className="text-[20px]">Your order is delivered!</h1>
43        ) : data?.status === "Processing refund" ? (
44          <h1 className="text-[20px]">Your refund is processing!</h1>
45        ) : data?.status === "Refund Success" ? (
46          <h1 className="text-[20px]">Your Refund is success!</h1>
47        ) : null
48      </>
    </div>
  );
}
```



bestdeals.jsx

```
frontend > src > components > Route > BestDeals > BestDeals.jsx > ...
1  import React, { useEffect, useState } from "react";
2  import { useSelector } from "react-redux";
3  import styles from "../../../../../styles/styles";
4  import ProductCard from "../ProductCard/ProductCard";
5
6  const BestDeals = () => {
7    const [data, setData] = useState([]);
8    const { allProducts } = useSelector((state) => state.products);
9    useEffect(() => {
10      const allProductsData = allProducts ? [...allProducts] : [];
11      const sortedData = allProductsData?.sort((a,b) => b.sold_out - a.sold_out);
12      const firstFive = sortedData && sortedData.slice(0, 5);
13      setData(firstFive);
14    }, [allProducts]);
15
16
17  return (
18    <div>
19      <div className={`${styles.section}`}>
20        <div className={`${styles.heading}`}>
21          <h1>Best Deals</h1>
22        </div>
23        <div className="grid grid-cols-1 gap-[20px] md:grid-cols-2 md:gap-[25px] lg:grid-cols-4 lg:gap-[25px] xl:grid-cols-5 xl:gap-[25px]">
24          {data && data.map((i, index) => <ProductCard data={i} key={index} />)}
25        </div>
26      </div>
27    </div>
28  );
29};
30
31 export default BestDeals;
32
```

Route codes:

featuredProduct.jsx

```
frontend > src > components > Route > FeaturedProduct >  FeaturedProduct.jsx > ...
1  import React, { useEffect } from "react";
2  import { useSelector } from "react-redux";
3  import styles from "../../../../../styles/styles";
4  import ProductCard from "../ProductCard/ProductCard";
5
6  const FeaturedProduct = () => {
7    const {allProducts} = useSelector((state) => state.products);
8
9    return (
10      <div>
11        <div className={`${styles.section}`}>
12          <div className={`${styles.heading}`}>
13            <h1>Featured Products</h1>
14          </div>
15          <div className="grid grid-cols-1 gap-[20px] md:grid-cols-2 md:gap-[25px] lg:grid-cols-4 lg:gap-[25px] xl:grid-cols-5 xl:gap-[30px]">
16            {allProducts &&
17              allProducts.map((i, index) => <ProductCard data={i} key={index} />)
18            }
19          </div>
20        </div>
21      </div>
22    );
23  };
24
25  export default FeaturedProduct;
```

Productcard.jsx

```
1  import React, { useState, useEffect } from "react";
2  import Ratings from "../../../../../Products/Ratings";
3
4  const ProductCard = ({ data, isEvent }) => {
5    const { wishlist } = useSelector((state) => state.wishlist);
6    const { cart } = useSelector((state) => state.cart);
7    const [click, setClick] = useState(false);
8    const [open, setOpen] = useState(false);
9    const dispatch = useDispatch();
10
11    useEffect(() => {
12      if (wishlist && wishlist.find((i) => i._id === data._id)) {
13        setClick(true);
14      } else {
15        setClick(false);
16      }
17    }, [wishlist]);
18
19    const removeFromWishlistHandler = (data) => {
20      setClick(!click);
21      dispatch(removeFromWishlist(data));
22    };
23
24    const addToWishlistHandler = (data) => {
25      setClick(!click);
26      dispatch(addToWishlist(data));
27    };
28
29    const addToCartHandler = (id) => {
30      const isItemExists = cart && cart.find((i) => i._id === id);
31      if (!isItemExists) {
```

Sponsoder.jsx

```
frontend > src > components > Route > 🚧 Sponsoredjsx > ...
4  const Sponsored = () => {
5    return (
6      <div
7        |   className={`${styles.section} hidden sm:block bg-white py-10 px-5 mb-12 cursor-pointer rounded-xl`}
8      >
9        <div className="flex justify-between w-full">
10       <div className="flex items-start">
11         
16       </div>
17       <div className="flex items-start">
18         
23       </div>
24       <div className="flex items-start">
25         
30       </div>
31       <div className="flex items-start">
32         
37     </div>
38   </div>
39 
```

allcoupons.jsx

```
const dispatch = useDispatch();

useEffect(() => {
  setIsLoading(true);
  axios
    .get(`${server}/coupon/get-coupon/${seller._id}`, {
      withCredentials: true,
    })
    .then((res) => {
      setIsLoading(false);
      setCoupons(res.data.couponCodes);
    })
    .catch((error) => {
      setIsLoading(false);
    });
}, [dispatch]);

const handleDelete = async (id) => {
  axios.delete(`${server}/coupon/delete-coupon/${id}`, {withCredentials: true}).then((res) => {
    toast.success("Coupon code deleted successfully!")
  })
  window.location.reload();
};

const handleSubmit = async (e) => {
  e.preventDefault();

  await axios
    .post(`
```

allevent.jsx

```
const AllEvents = () => {
  const { events, isLoading } = useSelector((state) => state.events);
  const { seller } = useSelector((state) => state.seller);

  const dispatch = useDispatch();

  useEffect(() => {
    dispatch(getAllEventsShop(seller._id));
  }, [dispatch]);

  const handleDelete = (id) => {
    dispatch(deleteEvent(id));
    window.location.reload();
  }

  const columns = [
    { field: "id", headerName: "Product Id", minWidth: 150, flex: 0.7 },
    {
      field: "name",
      headerName: "Name",
      minWidth: 180,
      flex: 1.4,
    },
    {
      field: "price",
      headerName: "Price",
      minWidth: 100,
      flex: 0.6,
    },
  ];
}
```



allorders.jsx

```
frontend > src > components > Shop > AllOrdersjsx > ...
11  const { orders, isLoading } = useSelector((state) => state.order);
12  const { seller } = useSelector((state) => state.seller);

13  const dispatch = useDispatch();

14  useEffect(() => {
15    dispatch(getAllOrdersOfShop(seller._id));
16  }, [dispatch]);

17  const columns = [
18    { field: "id", headerName: "Order ID", minWidth: 150, flex: 0.7 },
19    {
20      field: "status",
21      headerName: "Status",
22      minWidth: 130,
23      flex: 0.7,
24      cellClassName: (params) => {
25        return params.getValue(params.id, "status") === "Delivered"
26        ? "greenColor"
27        : "redColor";
28      },
29    },
30    {
31      field: "itemsQty",
32      headerName: "Items Qty",
33      type: "number",
34      minWidth: 130,
35      flex: 0.7,
36    },
37    {
38      field: "total",
39    },
40  ];
41
42
```



Allproducts.jsx

```
10
11 const AllProducts = () => {
12   const { products, isLoading } = useSelector((state) => state.products);
13   const { seller } = useSelector((state) => state.seller);
14
15   const dispatch = useDispatch();
16
17   useEffect(() => {
18     dispatch(getAllProductsShop(seller._id));
19   }, [dispatch]);
20
21   const handleDelete = (id) => {
22     dispatch(deleteProduct(id));
23     window.location.reload();
24   };
25
26   const columns = [
27     { field: "id", headerName: "Product Id", minWidth: 150, flex: 0.7 },
28     {
29       field: "name",
30       headerName: "Name",
31       minWidth: 180,
32       flex: 1.4,
33     },
34     {
35       field: "price",
36       headerName: "Price",
37       minWidth: 100,
38       flex: 0.6,
39     },
40   ];
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
698
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
788
789
789
790
791
792
793
794
795
796
797
798
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
888
889
889
890
891
892
893
894
895
896
897
897
898
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
987
988
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1087
1088
1088
1089
1089
1090
1091
1092
1093
1094
1095
1095
1096
1096
1097
1097
1098
1099
1099
1100
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1137
1138
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1167
1168
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1177
1178
1178
1179
1179
1180
1181
1182
1183
1184
1185
1186
1187
1187
1188
1188
1189
1189
1190
1191
1192
1193
1194
1195
1196
1197
1197
1198
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1217
1218
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1227
1228
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1237
1238
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1247
1248
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1257
1258
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1267
1268
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1277
1278
1278
1279
1279
1280
1281
1282
1283
1284
1285
1286
1287
1287
1288
1288
1289
1289
1290
1291
1292
1293
1294
1295
1296
1297
1297
1298
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1317
1318
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1327
1328
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1337
1338
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1347
1348
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1357
1358
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1367
1368
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1377
1378
1378
1379
1379
1380
1381
1382
1383
1384
1385
1386
1387
1387
1388
1388
1389
1389
1390
1391
1392
1393
1394
1395
1396
1397
1397
1398
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1407
1408
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1417
1418
1418
1419
1419
1420
1421
1422
1423
1424
1425
1426
1427
1427
1428
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1437
1438
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1447
1447
1448
1448
1449
1449
1450
1451
1452
1453
1454
1455
1456
1457
1457
1458
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1467
1468
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1477
1478
1478
1479
1479
1480
1481
1482
1483
1484
1485
1486
1487
1487
1488
1488
1489
1489
1490
1491
1492
1493
1494
1495
1496
1497
1497
1498
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1507
1508
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1517
1518
1518
1519
1519
1520
1521
1522
1523
1524
1525
1526
1527
1527
1528
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1537
1538
1538
1539
1539
1540
1541
1542
1543
1544
1545
1546
1547
1547
1548
1548
1549
1549
1550
1551
1552
1553
1554
1555
1556
1557
1557
1558
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1567
1568
1568
1569
1569
1570
1571
1572
1573
1574
1575
1576
1577
1577
1578
1578
1579
1579
1580
1581
1582
1583
1584
1585
1586
1587
1587
1588
1588
1589
1589
1590
1591
1592
1593
1594
1595
1596
1597
1597
1598
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1607
1608
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1617
1618
1618
1619
1619
1620
1621
1622
1623
1624
1625
1626
1627
1627
1628
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1637
1638
1638
1639
1639
1640
1641
1642
1643
1644
1645
1646
1647
1647
1648
1648
1649
1649
1650
1651
1652
1653
1654
1655
1656
1657
1657
1658
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1667
1668
1668
1669
1669
1670
1671
1672
1673
1674
1675
1676
1677
1677
1678
1678
1679
1679
1680
1681
1682
1683
1684
1685
1686
1687
1687
1688
1688
1689
1689
1690
1691
1692
1693
1694
1695
1696
1697
1697
1698
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1707
1708
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1717
1718
1718
1719
1719
1720
1721
1722
1723
1724
1725
1726
1727
1727
1728
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1737
1738
1738
1739
1739
1740
1741
1742
1743
1744
1745
1746
1747
1747
1748
1748
1749
1749
1750
1751
1752
1753
1754
1755
1756
1757
1757
1758
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1767
1768
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1777
1778
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1787
1788
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1797
1798
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1807
1808
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1817
1818
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1827
1828
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1837
1838
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1847
1848
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1857
1858
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1867
1868
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1877
1878
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1887
1888
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1897
1898
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1907
1908
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1917
1918
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1927
1928
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1937
1938
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1947
1948
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1957
1958
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1967
1968
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1977
1978
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1987
1988
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1997
1998
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2007
2008
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2017
2018
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2027
2028
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2037
2038
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2047
2048
2048
2049
2049
2050
2051
2052
2053
2054
2055
2056
2057
2057
2058
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2067
2068
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2077
2078
2078
2079
2079
2080
2081
2082
2083
2084
2085
2086
208
```

```

const AllRefundOrders = () => {
  const { orders, isLoading } = useSelector((state) => state.order);
  const { seller } = useSelector((state) => state.seller);

  const dispatch = useDispatch();

  useEffect(() => {
    dispatch(getAllOrdersOfShop(seller._id));
  }, [dispatch]);

  const refundOrders = orders && orders.filter((item) => item.status === "Processing refund" || item.status ===
}

const columns = [
  { field: "id", headerName: "Order ID", minWidth: 150, flex: 0.7 },
  {
    field: "status",
    headerName: "Status",
    minWidth: 130,
    flex: 0.7,
    cellClassName: (params) => {
      return params.getValue(params.id, "status") === "Delivered"
        ? "greenColor"
        : "redColor";
    },
  },
  {
    field: "itemsQty",
    headerName: "Items Qty",
    type: "number",
    minWidth: 130,
  }
];

const CreateEvent = () => {
  const { seller } = useSelector((state) => state.seller);
  const { success, error } = useSelector((state) => state.events);
  const navigate = useNavigate();
  const dispatch = useDispatch();

  const [images, setImages] = useState([]);
  const [name, setName] = useState("");
  const [description, setDescription] = useState("");
  const [category, setCategory] = useState("");
  const [tags, setTags] = useState("");
  const [originalPrice, setOriginalPrice] = useState();
  const [discountPrice, setDiscountPrice] = useState();
  const [stock, setStock] = useState();
  const [startDate, setStartDate] = useState(null);
  const [endDate, setEndDate] = useState(null);

  const handleStartDateChange = (e) => {
    const startDate = new Date(e.target.value);
    const minEndDate = new Date(startDate.getTime() + 3 * 24 * 60 * 60 * 1000);
    setStartDate(startDate);
    setEndDate(null);
    document.getElementById("end-date").min = minEndDate.toISOString().slice(0,10);
  }

  const handleEndDateChange = (e) => {
    const endDate = new Date(e.target.value);
    setEndDate(endDate);
  };

  const today = new Date().toISOString().slice(0,10);
}

```

```

const CreateProduct = () => {
  const { seller } = useSelector((state) => state.seller);
  const { success, error } = useSelector((state) => state.products);
  const navigate = useNavigate();
  const dispatch = useDispatch();

  const [images, setImages] = useState([]);
  const [name, setName] = useState("");
  const [description, setDescription] = useState("");
  const [category, setCategory] = useState("");
  const [tags, setTags] = useState("");
  const [originalPrice, setOriginalPrice] = useState();
  const [discountPrice, setDiscountPrice] = useState();
  const [stock, setStock] = useState();

  useEffect(() => {
    if (error) {
      toast.error(error);
    }
    if (success) {
      toast.success("Product created successfully!");
      navigate("/dashboard");
      window.location.reload();
    }
  }, [dispatch, error, success]);

  const handleImageChange = (e) => {
    e.preventDefault();

    let files = Array.from(e.target.files);
    setImages([...prevImages, ...files]);
  };
}

const DashboardMessages = () => {
  const { seller } = useSelector((state) => state.seller);
  const [conversations, setConversations] = useState([]);
  const [arrivalMessage, setArrivalMessage] = useState(null);
  const [currentChat, setCurrentChat] = useState();
  const [messages, setMessages] = useState([]);
  const [userData, setUserData] = useState(null);
  const [newMessage, setNewMessage] = useState("");
  const [onlineUsers, setOnlineUsers] = useState([]);
  const [activeStatus, setActiveStatus] = useState(false);
  const [open, setOpen] = useState(false);
  const scrollRef = useRef(null);

  useEffect(() => {
    socketId.on("getMessage", (data) => {
      setArrivalMessage({
        sender: data.senderId,
        text: data.text,
        createdAt: Date.now(),
      });
    });
  }, []);

  useEffect(() => {
    arrivalMessage &&
      currentChat?.members.includes(arrivalMessage.sender) &&
      setMessages((prev) => [...prev, arrivalMessage]);
  }, [arrivalMessage, currentChat]);

  useEffect(() => {
    const getConversation = async () => {
      ...
    };
  });
}

```

```

const OrderDetails = () => {
  const { orders, isLoading } = useSelector((state) => state.order);
  const { seller } = useSelector((state) => state.seller);
  const dispatch = useDispatch();
  const [status, setStatus] = useState("");
  const navigate = useNavigate();

  const { id } = useParams();

  useEffect(() => {
    dispatch(getAllOrdersOfShop(seller._id));
  }, [dispatch]);

  const data = orders && orders.find((item) => item._id === id);

  const orderUpdateHandler = async (e) => {
    await axios
      .put(` ${server}/order/update-order-status/${id}`,
      {
        status,
      },
      { withCredentials: true }
    )
      .then((res) => {
        toast.success("Order updated!");
        navigate("/dashboard-orders");
      })
      .catch((error) => {
        toast.error(error.response.data.message);
      });
  };
}

```

shopcreate.jsx

```

const ShopCreate = () => {
  const navigate = useNavigate();
  const [email, setEmail] = useState("");
  const [name, setName] = useState("");
  const [phoneNumber, setPhoneNumber] = useState();
  const [address, setAddress] = useState("");
  const [zipCode, setZipCode] = useState();
  const [avatar, setAvatar] = useState();
  const [password, setPassword] = useState("");
  const [visible, setVisible] = useState(false);

  const handleSubmit = async (e) => {
    e.preventDefault();
    const config = { headers: { "Content-Type": "multipart/form-data" } };

    const newForm = new FormData();

    newForm.append("file", avatar);
    newForm.append("name", name);
    newForm.append("email", email);
    newForm.append("password", password);
    newForm.append("zipCode", zipCode);
    newForm.append("address", address);
    newForm.append("phoneNumber", phoneNumber);
    axios
      .post(` ${server}/shop/create-shop`, newForm, config)
      .then((res) => {
        toast.success(res.data.message);
        constName("/");
      });
  };
}

```

ShopInfo.jsx

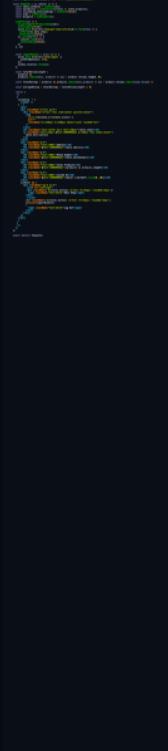
```
const ShopInfo = ({ isOwner }) => {
  const [data, setData] = useState({});

  const {products} = useSelector((state) => state.products);
  const [isLoading, setIsLoading] = useState(false);
  const {id} = useParams();
  const dispatch = useDispatch();

  useEffect(() => {
    dispatch(getAllProductsShop(id));
    setIsLoading(true);
    axios.get(`${server}/shop/get-shop-info/${id}`).then((res) => {
      setData(res.data.shop);
      setIsLoading(false);
    }).catch((error) => {
      console.log(error);
      setIsLoading(false);
    })
  }, [id])

  const logoutHandler = async () => {
    axios.get(`${server}/shop/logout`, {
      withCredentials: true,
    });
    window.location.reload();
  };

  const totalReviewsLength =
    products &&
    products.reduce((acc, product) => acc + product.reviews.length, 0);
}
```



ShopLogin.jsx

```
const ShopLogin = () => {
  const navigate = useNavigate();
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [visible, setVisible] = useState(false);

  const handleSubmit = async (e) => {
    e.preventDefault();

    await axios
      .post(` ${server}/shop/login-shop`,
        {
          email,
          password,
        },
        { withCredentials: true }
      )
      .then((res) => {
        toast.success("Login Success!");
        navigate("/dashboard");
        window.location.reload(true);
      })
      .catch((err) => {
        toast.error(err.response.data.message);
      });
  };

  return (
    <div className="min-h-screen bg-gray-50 flex flex-col justify-center py-12 sm:px-6 lg:px-8">
      <div className="sm:mx-auto sm:w-full sm:max-w-md">
        <h2 className="mt-6 text-center text-3xl font-extrabold text-gray-900">
```



ShopProfileData.jsx

```
const ShopProfileData = ({ isOwner }) => {
  const { products } = useSelector((state) => state.products);
  const { events } = useSelector((state) => state.events);
  const { seller } = useSelector((state) => state.seller);
  const { id } = useParams();
  const dispatch = useDispatch();

  useEffect(() => {
    dispatch(getAllProductsShop(id));
    dispatch(getAllEventsShop(seller._id));
  }, [dispatch]);

  const [active, setActive] = useState(1);

  const allReviews =
    products && products.map((product) => product.reviews).flat();

  return (
    <div className="w-full">
      <div className="flex w-full items-center justify-between">
        <div className="w-full flex">
          <div className="flex items-center" onClick={() => setActive(1)}>
            <h5
              className={`font-[600] text-[20px] ${active === 1 ? "text-red-500" : "text-[#333]`}
              cursor-pointer pr-[20px]`}>
              >
              Shop Products
            </h5>
          </div>
        </div>
      </div>
    </div>
  );
}
```



Shopsettings.jsx

```
const ShopSettings = () => {
  const { seller } = useSelector((state) => state.seller);
  const [avatar, setAvatar] = useState();
  const [name, setName] = useState(seller && seller.name);
  const [description, setDescription] = useState(seller && seller.description ? seller.description : "");
  const [address, setAddress] = useState(seller && seller.address);
  const [phoneNumber, setPhoneNumber] = useState(seller && seller.phoneNumber);
  const [zipCode, setZipcode] = useState(seller && seller.zipCode);

  const dispatch = useDispatch();

  const handleImage = async (e) => {
    e.preventDefault();
    const file = e.target.files[0];
    setAvatar(file);

    const formData = new FormData();

    formData.append("image", e.target.files[0]);

    await axios.put(`${server}/shop/update-shop-avatar`, formData, {
      headers: {
        "Content-Type": "multipart/form-data",
      },
      withCredentials: true,
    }).then((res) => {
      dispatch(loadSeller());
      toast.success("Avatar updated successfully!")
    }).catch((error) => {
      toast.error(error.response.data.message);
    });
  };
}
```



Signup.jsx

```
const Singup = () => {
  const [email, setEmail] = useState("");
  const [name, setName] = useState("");
  const [password, setPassword] = useState("");
  const [visible, setVisible] = useState(false);
  const [avatar, setAvatar] = useState(null);

  const handleFileInputChange = (e) => {
    const file = e.target.files[0];
    setAvatar(file);
  };

  const handleSubmit = async (e) => {
    e.preventDefault();
    const config = { headers: { "Content-Type": "multipart/form-data" } };

    const newForm = new FormData();

    newForm.append("file", avatar);
    newForm.append("name", name);
    newForm.append("email", email);
    newForm.append("password", password);

    axios
      .post(`${server}/user/create-user`, newForm, config)
      .then((res) => {
        toast.success(res.data.message);
        setName("");
        setEmail("");
        setPassword("");
        setAvatar();
      });
  };
}
```



Wishlist.jsx

```
const Wishlist = ({ setOpenWishlist }) => {
  const { wishlist } = useSelector((state) => state.wishlist);
  const dispatch = useDispatch();

  const removeFromWishlistHandler = (data) => {
    dispatch(removeFromWishlist(data));
  };

  const addToCartHandler = (data) => {
    const newData = { ...data, qty:1 };
    dispatch(addToCart(newData));
    setOpenWishlist(false);
  };

  return (
    <div className="fixed top-0 left-0 w-full bg-[#0000004b] h-screen z-10">
      <div className="fixed top-0 right-0 min-h-full w-[25%] bg-white flex flex-col justify-between shadow-sm">
        {wishlist && wishlist.length === 0 ? (
          <div className="w-full h-screen flex items-center justify-center">
            <div className="flex w-full justify-end pt-5 pr-5 fixed top-3 right-3">
              <RxCross1
                size={25}
                className="cursor-pointer"
                onClick={() => setOpenWishlist(false)}
              />
            </div>
            <h5>Wishlist Items is empty!</h5>
          </div>
        ) : (
          <>
            <div>
```



UserOrderDetails.jsx



```
const UserOrderDetails = () => {
  const { orders } = useSelector((state) => state.order);
  const { user } = useSelector((state) => state.user);
  const dispatch = useDispatch();
  const [open, setOpen] = useState(false);
  const [comment, setComment] = useState("");
  const [selectedItem, setSelectedItem] = useState(null);
  const [rating, setRating] = useState(1);

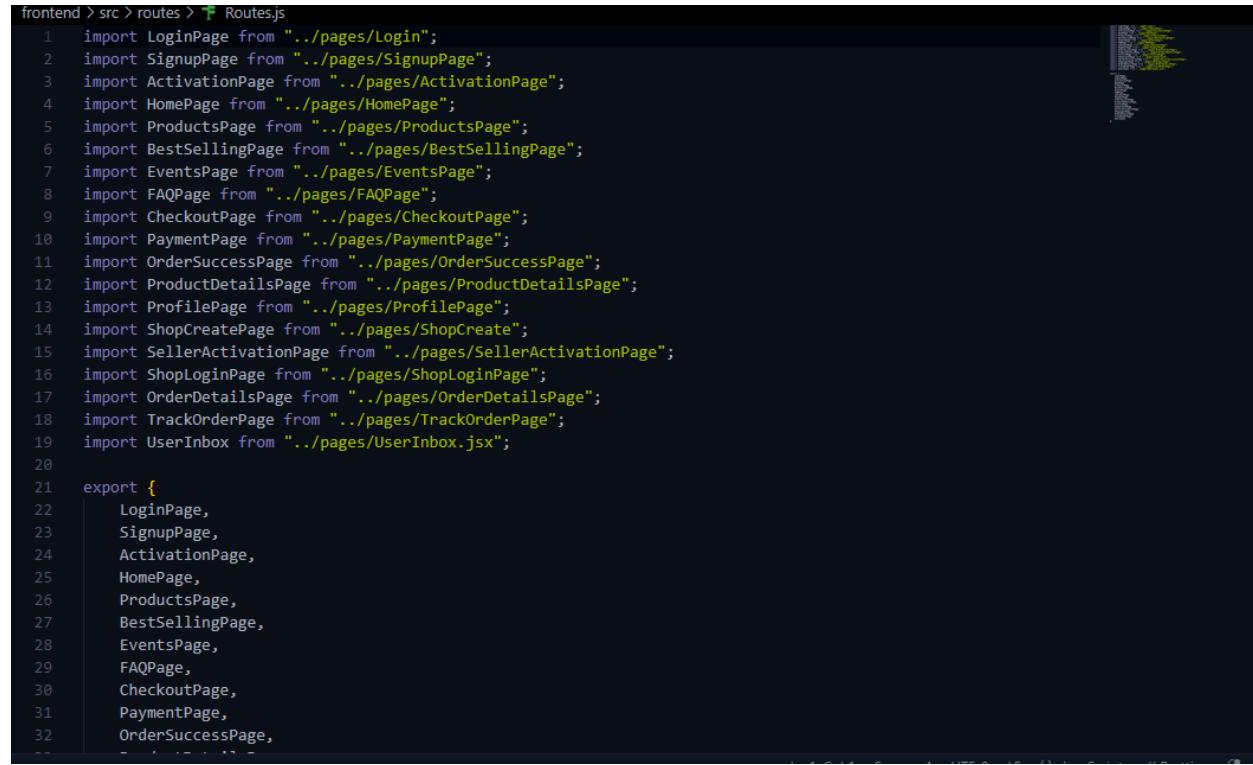
  const { id } = useParams();

  useEffect(() => {
    dispatch(getAllOrdersOfUser(user._id));
  }, [dispatch]);

  const data = orders && orders.find((item) => item._id === id);

  const reviewHandler = async (e) => {
    await axios
      .put(
        `${server}/product/create-new-review`,
        {
          user,
          rating,
          comment,
          productId: selectedItem?._id,
          orderId: id,
        },
        { withCredentials: true }
      )
      .then((res) => {
        // Handle success
      })
      .catch((err) => {
        // Handle error
      });
  };
}
```

Routes.js



```
frontend > src > routes > Routes.js
1 import LoginPage from "../pages/Login";
2 import SignupPage from "../pages/SignupPage";
3 import ActivationPage from "../pages/ActivationPage";
4 import HomePage from "../pages/HomePage";
5 import ProductsPage from "../pages/ProductsPage";
6 import BestSellingPage from "../pages/BestSellingPage";
7 import EventsPage from "../pages/EventsPage";
8 import FAQPage from "../pages/FAQPage";
9 import CheckoutPage from "../pages/CheckoutPage";
10 import PaymentPage from "../pages/PaymentPage";
11 import OrderSuccessPage from "../pages/OrderSuccessPage";
12 import ProductDetailsPage from "../pages/ProductDetailsPage";
13 import ProfilePage from "../pages/ProfilePage";
14 import ShopCreatePage from "../pages/ShopCreate";
15 import SellerActivationPage from "../pages/SellerActivationPage";
16 import ShopLoginPage from "../pages/ShopLoginPage";
17 import OrderDetailsPage from "../pages/OrderDetailsPage";
18 import TrackOrderPage from "../pages/TrackOrderPage";
19 import UserInbox from "../pages/UserInbox.jsx";
20
21 export {
22   LoginPage,
23   SignupPage,
24   ActivationPage,
25   HomePage,
26   ProductsPage,
27   BestSellingPage,
28   EventsPage,
29   FAQPage,
30   CheckoutPage,
31   PaymentPage,
32   OrderSuccessPage,
```

Conversation.js

```
backend > model > js conversation.js > ...
1  const mongoose = require("mongoose");
2
3  const conversationSchema = new mongoose.Schema(
4    {
5      groupTitle: {
6        type: String,
7      },
8      members: {
9        type: Array,
10     },
11      lastMessage: {
12        type: String,
13      },
14      lastMessageId: {
15        type: String,
16      },
17    },
18    { timestamps: true }
19  );
20
21 module.exports = mongoose.model("Conversation", conversationSchema);
22
```

CouponCode.js

```
backend > model > js coupounCodejs > ...
1  const mongoose = require("mongoose");
2
3  const coupounCodeSchema = new mongoose.Schema({
4      name:{
5          type: String,
6          required:[true,"Enter Coupoun Code Here"],
7          unique: true,
8      },
9      value:{
10         type: Number,
11         required: true,
12     },
13     minAmount:{
14         type: Number,
15     },
16     maxAmount:{
17         type: Number,
18     },
19     shopId:{
20         type: String,
21         required: true,
22     },
23     selectedProduct:{
24         type: String,
25     },
26     createdAt:{
27         type: Date,
28         default: Date.now(),
29     }
30 });
31
32 module.exports = mongoose.model("CoupounCode", coupounCodeSchema);
```

Event.js

```
backend > model > js event.js > ...
1  const mongoose = require("mongoose");
2
3  const eventSchema = new mongoose.Schema({
4      name:{  
5          type: String,  
6          required:[true,"Please enter your event product name!"],  
7      },  
8      description:{  
9          type: String,  
10         required:[true,"Please enter your event product description!"],  
11     },  
12     category:{  
13         type: String,  
14         required:[true,"Please enter your event product category!"],  
15     },  
16     start_Date: {  
17         type: Date,  
18         required: true,  
19     },  
20     Finish_Date: {  
21         type: Date,  
22         required: true,  
23     },  
24     status: {  
25         type: String,  
26         default: "Running",  
27     },  
28     tags:{  
29         type: String,  
30     },  
31     originalPrice:{  
32         type: Number,
```

Messages.js

```
backend > model > messages.js > ...
1  const mongoose = require("mongoose");
2
3  const messagesSchema = new mongoose.Schema(
4    {
5      conversationId: {
6        type: String,
7      },
8      text: {
9        type: String,
10     },
11      sender: {
12        type: String,
13     },
14      images: [
15        {
16          type: String,
17        }
18      ],
19    },
20    { timestamps: true }
21  );
22
23 module.exports = mongoose.model("Messages", messagesSchema);
24
```

Order.js

```
backend > model > js order.js > ...
1  const mongoose = require("mongoose");
2
3  const orderSchema = new mongoose.Schema({
4    cart: {
5      type: Array,
6      required: true,
7    },
8    shippingAddress: {
9      type: Object,
10     required: true,
11   },
12   user: {
13     type: Object,
14     required: true,
15   },
16   totalPrice: {
17     type: Number,
18     required: true,
19   },
20   status: {
21     type: String,
22     default: "Processing",
23   },
24   paymentInfo: {
25     id: {
26       type: String,
27     },
28     status: {
29       type: String,
30     },
31     type: {
32       type: String,
33     }
34   }
35 });
36
37 module.exports = mongoose.model("Order", orderSchema);
```

Product.js

```
backend > model > product.js > ...
1  const mongoose = require("mongoose");
2
3  const productSchema = new mongoose.Schema({
4    name: {
5      type: String,
6      required: [true, "Please enter your product name!"],
7    },
8    description: {
9      type: String,
10     required: [true, "Please enter your product description!"],
11   },
12   category: {
13     type: String,
14     required: [true, "Please enter your product category!"],
15   },
16   tags: {
17     type: String,
18   },
19   originalPrice: {
20     type: Number,
21   },
22   discountPrice: {
23     type: Number,
24     required: [true, "Please enter your product price!"],
25   },
26   stock: {
27     type: Number,
28     required: [true, "Please enter your product stock!"],
29   },
30   images: [
31     {
32       type: String,
33     }
34   ]
35 })
36
37 module.exports = mongoose.model("Product", productSchema);
```

Shop.js

```
backend > model > shop.js > ...
● 1 const mongoose = require("mongoose");
  2 const bcrypt = require("bcryptjs");
  3 const jwt = require("jsonwebtoken");
  4
  5 const shopSchema = new mongoose.Schema({
  6   name: {
  7     type: String,
  8     required: [true, "Please enter your shop name!"],
  9   },
 10   email: {
 11     type: String,
 12     required: [true, "Please enter your shop email address"],
 13   },
 14   password: {
 15     type: String,
 16     required: [true, "Please enter your password"],
 17     minLength: [6, "Password should be greater than 6 characters"],
 18     select: false,
 19   },
 20   description: {
 21     type: String,
 22   },
 23   address: {
 24     type: String,
 25     required: true,
 26   },
 27   phoneNumber: {
 28     type: Number,
 29     required: true,
 30   },
 31   role: {
 32     type: String,
```

Ln 1, Col 1 Spaces:2 UTF-8 LF

User.js

```
backend > model > js user.js > ...
1  const mongoose = require("mongoose");
2  const bcrypt = require("bcryptjs");
3  const jwt = require("jsonwebtoken");
4
5  const userSchema = new mongoose.Schema({
6    name: {
7      type: String,
8      required: [true, "Please enter your name!"],
9    },
10   email: {
11     type: String,
12     required: [true, "Please enter your email address"],
13   },
14   password: {
15     type: String,
16     required: [true, "Please enter your password"],
17     minLength: [6, "Password should be greater than 6 characters"],
18     select: false,
19   },
20   phoneNumber: {
21     type: Number,
22   },
23   addresses: [
24     {
25       country: {
26         type: String,
27       },
28       city: {
29         type: String,
30       },
31       address1: {
32         type: String,
33       }
34     }
35   ]
36 });
37
38 module.exports = mongoose.model("User", userSchema);
```

event.js

```
const AllEvents = () => {
  const { events, isLoading } = useSelector((state) => state.events);
  const { seller } = useSelector((state) => state.seller);

  const dispatch = useDispatch();

  useEffect(() => {
    dispatch(getAllEventsShop(seller._id));
  }, [dispatch]);

  const handleDelete = (id) => {
    dispatch(deleteEvent(id));
    window.location.reload();
  }

  const columns = [
    { field: "id", headerName: "Product Id", minWidth: 150, flex: 0.7 },
    {
      field: "name",
      headerName: "Name",
      minWidth: 180,
      flex: 1.4,
    },
    {
      field: "price",
      headerName: "Price",
      minWidth: 100,
      flex: 0.6,
    },
  ];
}
```

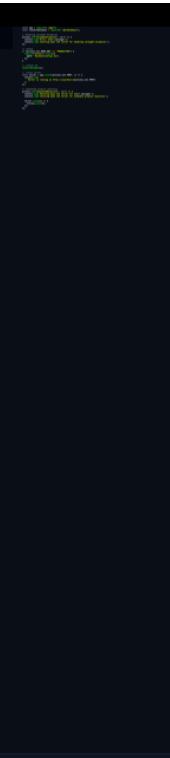


server.js

```
backend > js server.js > ...
1  const app = ...require("./app");
2  const connectDatabase = require("./db/Database");
3
4 // Handling uncaught Exception
5 process.on("uncaughtException", (err) => {
6   console.log(`Error: ${err.message}`);
7   console.log(`shutting down the server for handling uncaught exception`);
8 });
9
10 // config
11 if (process.env.NODE_ENV !== "PRODUCTION") {
12   require("dotenv").config({
13     path: "backend/config/.env",
14   });
15 }
16
17 // connect db
18 connectDatabase();
19
20 // create server
21 const server = app.listen(process.env.PORT, () => {
22   console.log(`Server is running on http://localhost:${process.env.PORT}`);
23   });
24 });
25 });
26
27 // unhandled promise rejection
28 process.on("unhandledRejection", (err) => {
29   console.log(`Shutting down the server for ${err.message}`);
30   console.log(`shutting down the server for unhandled promise rejection`);
31
32   server.close(() => {
33     });

34 });

35 
```



ErrorHandler.js

```
backend > utils > js ErrorHandler.js > ErrorHandler
1  class ErrorHandler extends Error{
2    constructor(message,statusCode){
3      super(message);
4      this.statusCode = statusCode
5
6      Error.captureStackTrace(this,this.constructor);
7
8    }
9
10 }
11 module.exports = ErrorHandler
```

Jwttoken.js

```
backend > utils > js jwtToken.js > ...
● 1 // create token and saving that in cookies
2 const sendToken = (user, statusCode, res) => {
3   const token = user.getJwtToken();
4
5   // Options for cookies
6   const options = {
7     expires: new Date(Date.now() + 90 * 24 * 60 * 60 * 1000),
8     httpOnly: true,
9     Secure: true,
10   };
11
12   res.status(statusCode).cookie("token", token, options).json({
13     success: true,
14     user,
15     token,
16   });
17 }
18
19 module.exports = sendToken;
20
```

SendMail.js

```
backend > utils > js sendMail.js > ...
1  const nodemailer = require("nodemailer");
2
3  const sendMail = async (options) => {
4      const transporter = nodemailer.createTransport({
5          host: process.env.SMPT_HOST,
6          port: process.env.SMPT_PORT,
7          service: process.env.SMPT_SERVICE,
8          auth:{
9              user: process.env.SMPT_MAIL,
10             pass: process.env.SMPT_PASSWORD,
11         },
12     });
13
14     const mailOptions = {
15         from: process.env.SMPT_MAIL,
16         to: options.email,
17         subject: options.subject,
18         text: options.message,
19     };
20
21     await transporter.sendMail(mailOptions);
22 };
23
24 module.exports = sendMail;
```

ShopToken.js

```
backend > utils > shopToken.js > ...
1 // create token and saving that in cookies
2 const sendShopToken = (user, statusCode, res) => {
3   const token = user.getJwtToken();
4
5   // Options for cookies
6   const options = {
7     expires: new Date(Date.now() + 90 * 24 * 60 * 60 * 1000),
8     httpOnly: true,
9     Secure: true,
10    };
11
12   res.status(statusCode).cookie("seller_token", token, options).json({
13     success: true,
14     user,
15     token,
16   });
17 }
18
19 module.exports = sendShopToken;
20
```

Database.js

```
backend > db > Database.js > ...
1 const mongoose = require("mongoose");
2
3 const connectDatabase = () => {
4   mongoose
5     .connect(process.env.DB_URL, {
6       useNewUrlParser: true,
7       useUnifiedTopology: true,
8     })
9     .then((data) => {
10       console.log(`mongod connected with server: ${data.connection.host}`);
11     });
12 }
13
14 module.exports = connectDatabase;
15
```

conversation.js

```
6  const router = express.Router();
7
8 // create a new conversation
9 router.post(
10   "/create-new-conversation",
11   catchAsyncErrors(async (req, res, next) => {
12     try {
13       const { groupTitle, userId, sellerId } = req.body;
14
15       const isConversationExist = await Conversation.findOne({ groupTitle });
16
17       if (isConversationExist) {
18         const conversation = isConversationExist;
19         res.status(201).json({
20           success: true,
21           conversation,
22         });
23       } else {
24         const conversation = await Conversation.create({
25           members: [userId, sellerId],
26           groupTitle: groupTitle,
27         });
28
29         res.status(201).json({
30           success: true,
31           conversation,
32         });
33       }
34     } catch (error) {
35       return next(new ErrorHandler(error.response.message), 500);
36     }
37   })
38 );
```



couponCode.js

```
8
9 // create coupon code
10 router.post(
11   "/create-coupon-code",
12   isSeller,
13   catchAsyncErrors(async (req, res, next) => {
14     try {
15       const isCoupounCodeExists = await CoupounCode.find({
16         name: req.body.name,
17       });
18
19       if (isCoupounCodeExists.length !== 0) {
20         return next(new ErrorHandler("Coupoun code already exists!", 400));
21       }
22
23       const coupounCode = await CoupounCode.create(req.body);
24
25       res.status(201).json({
26         success: true,
27         coupounCode,
28       });
29     } catch (error) {
30       return next(new ErrorHandler(error, 400));
31     }
32   })
33 );
```



CreateEvent.js

```
1 // create event
2 router.post(
3   "/create-event",
4   upload.array("images"),
5   catchAsyncErrors(async (req, res, next) => {
6     try {
7       const shopId = req.body.shopId;
8       const shop = await Shop.findById(shopId);
9       if (!shop) {
10         return next(new ErrorHandler("Shop Id is invalid!", 400));
11     } else {
12       const files = req.files;
13       const imageUrl = files.map((file) => `${file.filename}`);
14
15       const eventData = req.body;
16       eventData.images = imageUrl;
17       eventData.shop = shop;
18
19       const product = await Event.create(eventData);
20
21       res.status(201).json({
22         success: true,
23       });
24     }
25   })
26 );
```

Message.js

```
7
8 // create new message
9 router.post(
10   "/create-new-message",
11   upload.array("images"),
12   catchAsyncErrors(async (req, res, next) => {
13     try {
14       const messageData = req.body;
15
16       if (req.files) {
17         const files = req.files;
18         const imageUrl = files.map((file) => `${file.fileName}`);
19         messageData.images = imageUrl;
20       }
21
22       messageData.conversationId = req.body.conversationId;
23       messageData.sender = req.body.sender;
24       messageData.text = req.body.text;
25
26       const message = new Messages({
27         conversationId: messageData.conversationId,
28         text: messageData.text,
29         sender: messageData.sender,
30         images: messageData.images ? messageData.images : undefined,
31       });
32     }
33   })
34 );
```

order.js

```
9 // create new order
10 router.post(
11   "/create-order",
12   catchAsyncErrors(async (req, res, next) => {
13     try {
14       const { cart, shippingAddress, user, totalPrice, paymentInfo } = req.body;
15
16       // group cart items by shopId
17       const shopItemsMap = new Map();
18
19       for (const item of cart) {
20         const shopId = item.shopId;
21         if (!shopItemsMap.has(shopId)) {
22           shopItemsMap.set(shopId, []);
23         }
24         shopItemsMap.get(shopId).push(item);
25       }
26
27       // create an order for each shop
28       const orders = [];
29
30       for (const [shopId, items] of shopItemsMap) {
31         const order = await Order.create({
32           cart: items,
33           shippingAddress,
34           user,
35           totalPrice,
```



payment.js

```
6
7   router.post(
8     "/process",
9     catchAsyncErrors(async (req, res, next) => {
10       const myPayment = await stripe.paymentIntents.create({
11         amount: req.body.amount,
12         currency: "inr",
13         metadata: {
14           company: "Becodemy",
15         },
16       });
17       res.status(200).json({
18         success: true,
19         client_secret: myPayment.client_secret,
20       });
21     })
22   );
23
24   router.get(
25     "/stripeapikey",
26     catchAsyncErrors(async (req, res, next) => {
27       res.status(200).json({ stripeApiKey: process.env.STRIPE_API_KEY });
28     })
29   );
30
31
32   module.exports = router;
```

product.js

```
11 // create product
12 router.post(
13   "/create-product",
14   upload.array("images"),
15   catchAsyncErrors(async (req, res, next) => {
16     try {
17       const shopId = req.body.shopId;
18       const shop = await Shop.findById(shopId);
19       if (!shop) {
20         return next(new ErrorHandler("Shop Id is invalid!", 400));
21       } else {
22         const files = req.files;
23         const imageUrls = files.map((file) => `${file.filename}`);
24
25         const productData = req.body;
26         productData.images = imageUrls;
27         productData.shop = shop;
28
29         const product = await Product.create(productData);
30
31         res.status(201).json({
32           success: true,
33           product,
34         });
35       }
36     } catch (error) {
37       return next(new ErrorHandler(error, 400));
38     }
39   })
40 )
```

Shop.js

```
15 // create shop
16 router.post("/create-shop", upload.single("file"), async (req, res, next) => {
17   try {
18     const { email } = req.body;
19     const sellerEmail = await Shop.findOne({ email });
20     if (sellerEmail) {
21       const filename = req.file.filename;
22       const filePath = `uploads/${filename}`;
23       fs.unlink(filePath, (err) => {
24         if (err) {
25           console.log(err);
26           res.status(500).json({ message: "Error deleting file" });
27         }
28       });
29       return next(new ErrorHandler("User already exists", 400));
30     }
31
32     const filename = req.file.filename;
33     const fileUrl = path.join(filename);
34
35     const seller = {
36       name: req.body.name,
37       email: email,
38       password: req.body.password,
39       avatar: fileUrl,
40       address: req.body.address,
41       phoneNumber: req.body.phoneNumber,
42       zipCode: req.body.zipCode,
43     };
44
45     const activationToken = createActivationToken(seller);
```

user.js

```
14 router.post("/create-user", upload.single("file"), async (req, res, next) => {
15   try {
16     const { name, email, password } = req.body;
17     const userEmail = await User.findOne({ email });
18
19     if (userEmail) {
20       const filename = req.file.filename;
21       const filePath = `uploads/${filename}`;
22       fs.unlink(filePath, (err) => {
23         if (err) {
24           console.log(err);
25           res.status(500).json({ message: "Error deleting file" });
26         }
27       });
28       return next(new ErrorHandler("User already exists", 400));
29     }
30
31     const filename = req.file.filename;
32     const fileUrl = path.join(filename);
33
34     const user = {
35       name,
36       email,
37       password,
38       avatar: fileUrl,
39     };
40
41     const activationToken = createActivationToken(user);
42
43   }
```

Technology (Framework, Languages)

Framework: MERN Stack

GitHub Repository

Link: <https://github.com/mrif449/Papier-Factory>