Shopez: E-Commerce Application

Project on SmartInternz
Full Stack Developer

Submitted by:

Koushika.S.S	: 211521243093
Bolla Hemambika	: 211521243031
Nandhini.R	: 211521243108
JayaSakthi.J	: 211521243074
Praveena.S	: 211521243119

INTRODUCTION

Overview

ShopEZ is your one-stop destination for effortless online shopping. With a user-friendly interface and a comprehensive product catalog, finding the perfect items has never been easier. Seamlessly navigate through detailed product descriptions, customer reviews, and available discounts to make informed decisions. Enjoy a secure checkout process and receive instant order confirmation. For sellers, our robust dashboard provides efficient order management and insightful analytics to drive business growth. Experience the future of online shopping with ShopEZ today. Shopez is a user-friendly e-commerce application designed to make online shopping convenient for customers. The platform offers a wide range of products from various categories to cater to diverse customer needs. Shopez provides a secure and seamless shopping experience for users.

Purpose

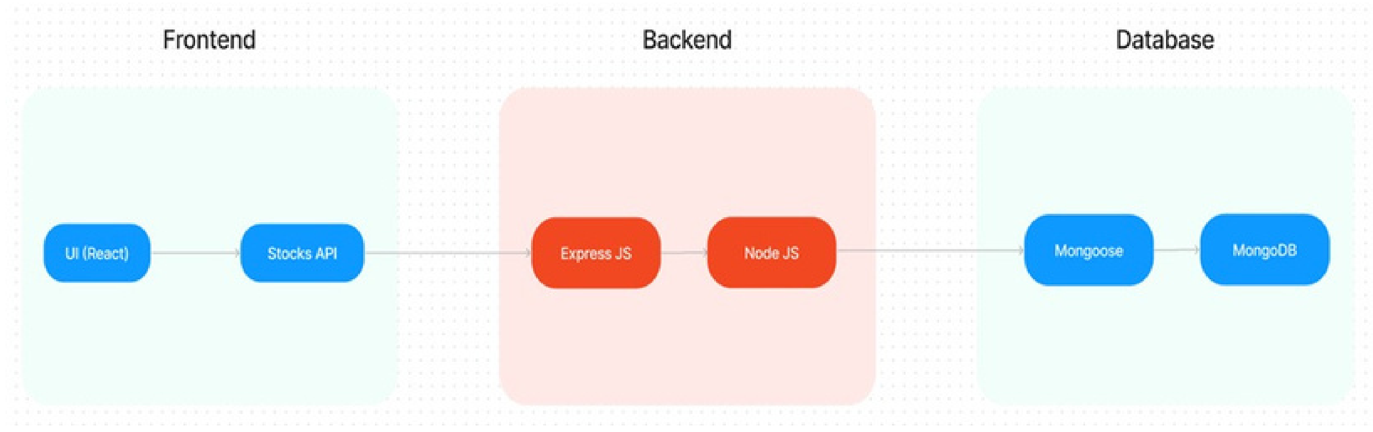
Shopez allows you to shop for a wide range of products, from clothing and accessories to electronics, home decor, and more.. It provides a convenient platform for browsing, searching, and purchasing items online, saving you time and effort.With Shopez, you can explore a vast catalog of products from various brands and sellers, giving you access to a diverse selection. The app offers detailed product descriptions, specifications, and customer reviews, helping you make informed purchasing decisions. The app allows you to track your orders and receive updates on their status, giving you peace of mind.Shopez often provides exclusive deals, discounts, and promotions, helping you save money on your purchases. It offers a user-friendly and intuitive interface, making it easy for anyone to navigate and use the app.Shopez provides customer support services to assist you with any queries, concerns, or issues you may encounter during your shopping experience. The app allows you to create wishlists, enabling you to save and organize products you're interested in for future reference.Shopez offers personalized recommendations based on your browsing and purchase history,

helping you discover new products that align with your interests. It provides a seamless shopping experience

across various devices. Including credit cards, digital wallets, and more.It ensures the privacy and security

of your personal information, safeguarding your data during transactions. Shopez offers reliable and efficient delivery services, ensuring that your purchases reach you in a timely manner.

Technical Architecture



In this architecture diagram:

• The frontend is represented by the "Frontend" section, including user interface components such as Us Authentication, Cart, Products, Profile, Admin dashboard, etc.,

• The backend is represented by the "Backend" section, consisting of API endpoints for Users, Orders, Products, etc., It also includes Admin Authentication and an Admin Dashboard.

• The Database section represents the database that stores collections for Users, cart, Orders and Produ

ER Diagram

The ShopEZ ER-diagram represents the entities and relationships involved in an e-commerce system. It illustrates how users, products, cart, and orders are interconnected. Here is a breakdown of the entities and their relationships: The Database section represents the database that stores collections for Users, Admin, Cart, Orders and products.

USER: Represents the individuals or entities who are registered in the platform. Admin: Represents a collection with important details such as Banner image and

Categories.

Products: Represents a collection of all the products available in the platform.

Cart: This collection stores all the products that are added to the cart by users. Here, the elements in the

cart are differentiated by the user Id. Orders: This collection stores all the orders that are made by the users in the platform. ShopEZ is designed to elevate your online shopping experience by

providing a seamless and user-friendly

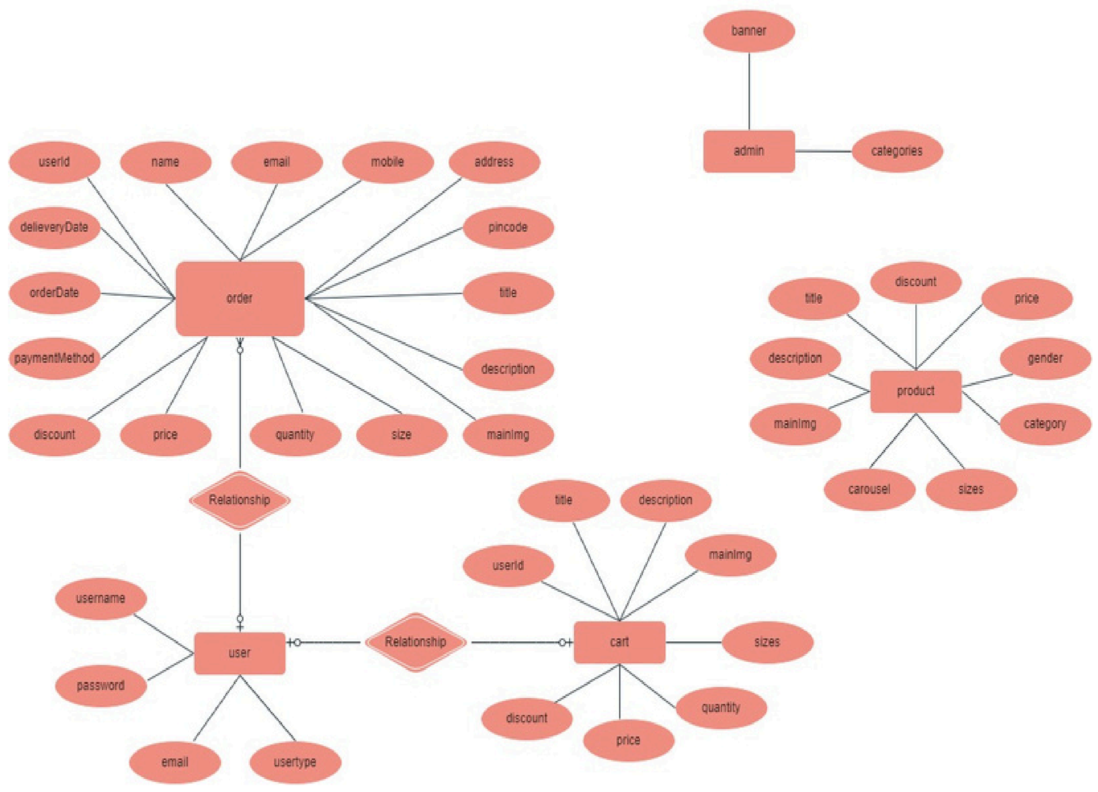
way to discover and purchase products. With our efficient checkout process, comprehensive product

catalog, and robust seller dashboard, we ensure a convenient and enjoyable online shopping

experience for

both shoppers and sellers alike.

ER-MODEL



Features

- Comprehensive Product Catalog: ShopEZ boasts an extensive catalog of products, offering a diverse range of items and options for shoppers. You can effortlessly explore and discover various products, complete with detailed descriptions, customer reviews, pricing, and available discounts, to find the perfect items for your needs.

- Shop Now Button: Each product listing features a convenient "Shop Now" button. When you find a product that aligns with your preferences, simply click on the button to initiate the purchasing process.

- Order Details Page: Upon clicking the "Shop Now" button, you will be directed to an order details page. Here, you can provide relevant information such as your shipping address, preferred payment method, and any specific product requirements.

- Order Confirmation and Details: After successfully placing an order, you will receive a confirmation notification. Subsequently, you will be directed to an order details page, where you can review all pertinent information about your order, including shipping details, payment method, and any specific product requests you specified.

Prerequisites

To develop a full-stack e-commerce app using React JS, Node.js, and MongoDB, there are several prerequisites you should consider. Here are the key prerequisites for developing such an application:

Node.js and npm:

Install Node.js, which includes npm (Node Package Manager), on your development machine. Node.js is required to run JavaScript on the server side.

MongoDB: Set up a MongoDB database to store hotel and booking information. Install MongoDB locally or use a cloud-based MongoDB service.

Express.js: Express.js is a web application framework for Node.js. Install Express.js to handle server-side routing,middleware, and API development. Installation: Open your command prompt or terminal and run the following command: npm install express

React.js: React.js is a popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components, making it easier to build dynamic and responsive web applications. To install React.js, a JavaScript library for building user interfaces, follow the installation guide:

HTML, CSS, and JavaScript: Basic knowledge of HTML for creating the structure of your app, CSS for styling,and JavaScript for client-side interactivity is essential.

Database Connectivity: Use a MongoDB driver or an Object-Document Mapping (ODM) library like Mongoose to connect your Node.js server with the MongoDB database and perform CRUD (Create, Read, Update, Delete) operations.

Front-end Framework: Utilize Angular to build the user-facing part of the application, including product listings, booking forms, and user interfaces for the admin dashboard.

Version Control: Use Git for version control, enabling collaboration and tracking changes throughout the development process. Platforms like GitHub or Bitbucket can host your repository.

Development Environment: Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.

User & Admin Flow

User Flow:

- Users start by registering for an account.
- After registration, they can log in with their credentials.
- Once logged in, they can check for the available products in the platform. • Users can add the products they wish to their carts and order.
- They can then proceed by entering address and payment details. • After ordering, they can check them in the profile section.

Admin Flow:

- Admins start by logging in with their credentials.
- Once logged in, they are directed to the Admin Dashboard.
- Admins can access the users list, products, orders, etc.,

Backened Development

- Set Up Project Structure:
  - Create a new directory for your project and set up a package.json file using the npm init command.
  - Install necessary dependencies such as Express.js, Mongoose, and other required packages.
- Database Configuration:
  - Set up a MongoDB database either locally or using a cloud-based MongoDB service like MongoDB Atlas or use locally with MongoDB compass.
  - Create a database and define the necessary collections for admin, users, products, orders and other relevant data.

- Create Express.js Server:
  - Set up an Express.js server to handle HTTP requests and serve API endpoints.
- Define API Routes:
  - Create separate route files for different API functionalities such as users, orders, and authentication.
  - Define the necessary routes for listing products, handling user registration and login,managing orders, etc.
  - Implement route handlers using Express.js to handle requests and interact with the database.
- Implement Data Models:
  - Define Mongoose schemas for the different data entities like products, users, and orders.
  - Create corresponding Mongoose models to interact with the MongoDB database.
  - Implement CRUD operations (Create, Read, Update, Delete) for each model to perform database operations.
- User Authentication:
  - Create routes and middleware for user registration, login, and logout.
  - Set up authentication middleware to protect routes that require user authentication.
- Handle new products and Orders:
  - Create routes and controllers to handle new product listings, including fetching products data from the database and sending it as a response.
  - Implement ordering(buy) functionality by creating routes and controllers to handle order requests, including validation and database updates.
- Admin Functionality:
  - Implement routes and controllers specific to admin functionalities such as adding products, managing user orders, etc.

- Add necessary authentication and authorization checks to ensure only authorized admins can access these routes.

- Error Handling:
  - Implement error handling middleware to catch and handle any errors that occur during the API requests.
  - Return appropriate error responses with relevant error messages and HTTP status codes.

Schema Use Case

- User Schema:
  - Schema: userSchema
  - Model: ‘User’
  - The User schema represents the user data and includes fields such as username, email, and password.
  - It is used to store user information for registration and authentication purposes.
  - The email field is marked as unique to ensure that each user has a unique email address

- Product Schema:
  - Schema: productSchema
  - Model: ‘Product’
  - The Product schema represents the data of all the products in the platform.
  - It is used to store information about the product details, which will later be useful for ordering .

- Orders Schema:
  - Schema: ordersSchema
  - Model: ‘Orders’
  - The Orders schema represents the orders data and includes fields such as userId, product Id, product name, quantity, size, order date, etc.,
  - It is used to store information about the orders made by users.
  - The user Id field is a reference to the user who made the order.

- Cart Schema:
  - Schema: cartSchema
  - Model: ‘Cart’
  - The Cart schema represents the cart data and includes fields such as userId, product Id, produce name, quantity, size, order date, etc.,
  - It is used to store information about the products added to the cart by users. • The user Id field is a reference to the user who has the product in cart.

- Admin Schema:
  - Schema: adminSchema
  - Model: ‘Admin’
  - The admin schema has essential data such as categories, banner.

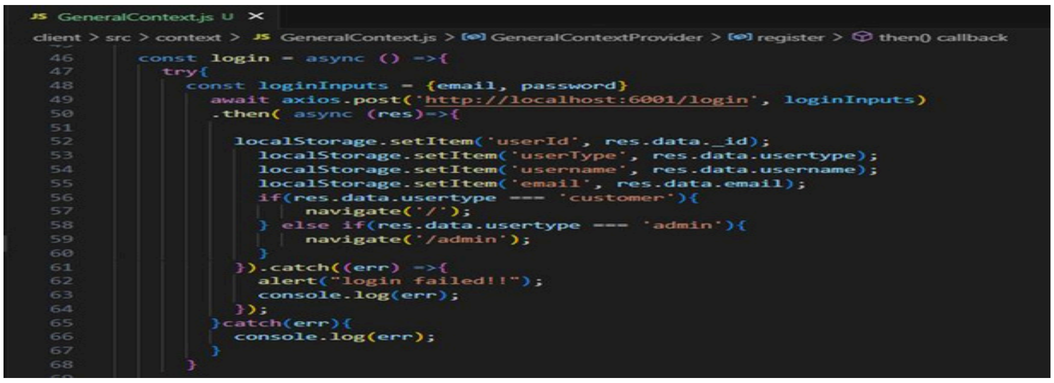
Web Development

- Set up react application
  - Create a React app in the client folder.
  - Install required libraries
  - Create required pages and components and add routes.

- Design UI Components
  - Create Components.
  - Implement layout and styling.
  - Add Navigation

- Code Explanation

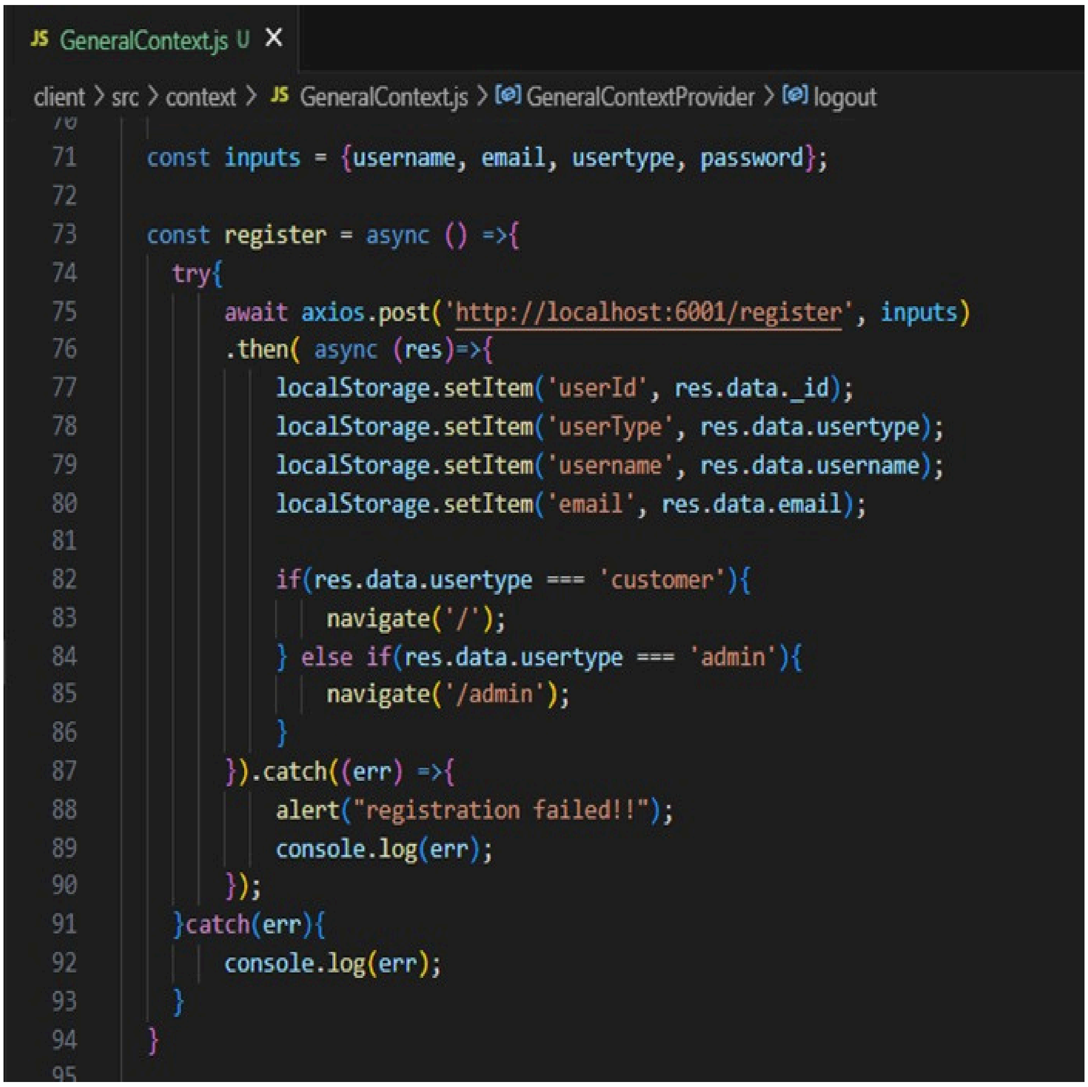
Frontend Login



GeneralContext.js U ×
client > src > context > GeneralContext.js > GeneralContextProvider > register > then() callback

```javascript
  const login = async () =>{
    try{
      const loginInputs = {email, password}
        await axios.post('http://localhost:6001/login', loginInputs)
        .then( async (res)=>{

          localStorage.setItem('userId', res.data._id);
            localStorage.setItem('userType', res.data.usertype);
            localStorage.setItem('username', res.data.username);
            localStorage.setItem('email', res.data.email);
            if(res.data.usertype === 'customer'){
                navigate('/');
            } else if(res.data.usertype === 'admin'){
                navigate('/admin');
            }
          }).catch((err) =>{
            alert("login failed!!");
            console.log(err);
          });
        }catch(err){
          console.log(err);
        }
      }
```

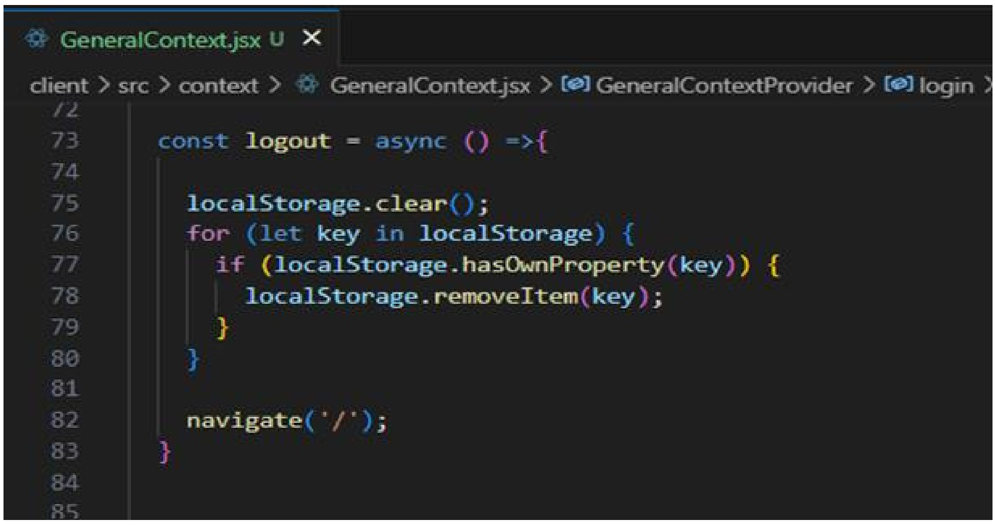
Register



GeneralContext.js U ×
client > src > context > GeneralContext.js > GeneralContextProvider > logout

```javascript
  const inputs = {username, email, usertype, password};

  const register = async () =>{
    try{
        await axios.post('http://localhost:6001/register', inputs)
        .then( async (res)=>{
            localStorage.setItem('userId', res.data._id);
            localStorage.setItem('userType', res.data.usertype);
            localStorage.setItem('username', res.data.username);
            localStorage.setItem('email', res.data.email);

            if(res.data.usertype === 'customer'){
                navigate('/');
            } else if(res.data.usertype === 'admin'){
                navigate('/admin');
            }
        }).catch((err) =>{
            alert("registration failed!!");
            console.log(err);
        });
    }catch(err){
        console.log(err);
    }
  }
```

Logout



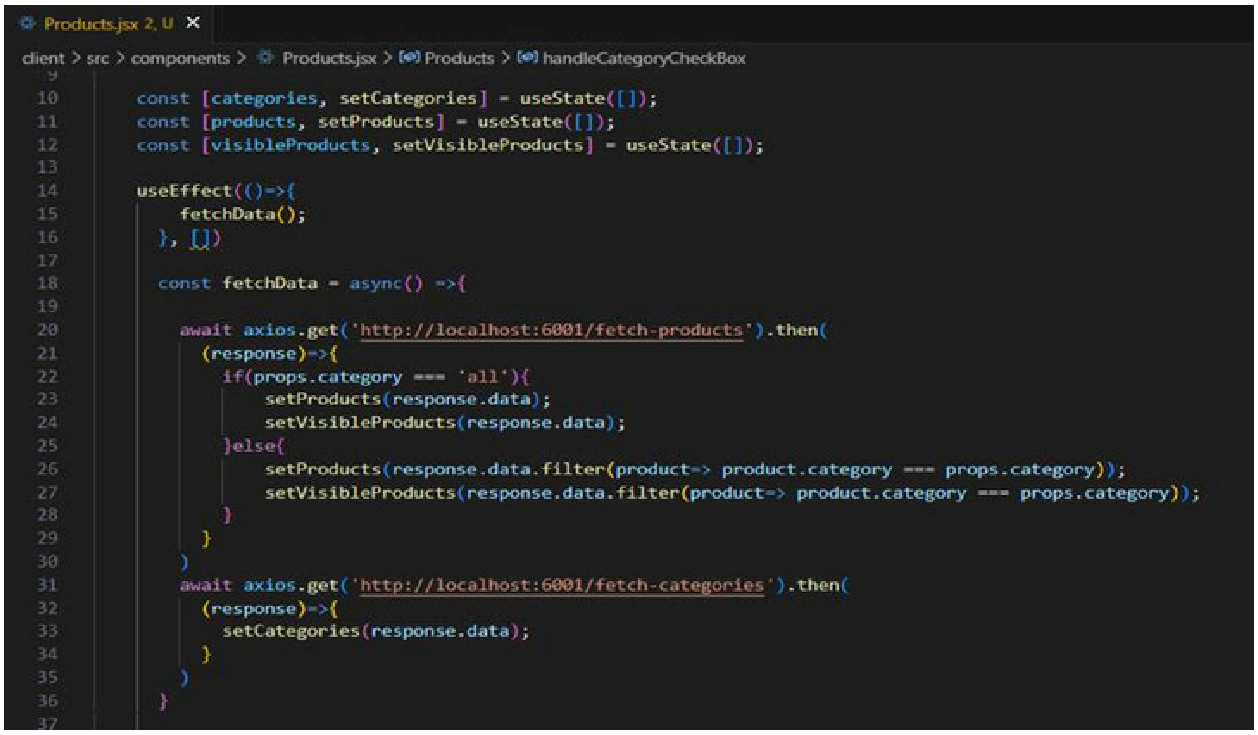
GeneralContext.jsx U ×

client > src > context > GeneralContext.jsx > GeneralContextProvider > login >

```
  const logout = async () =>{

    localStorage.clear();
    for (let key in localStorage) {
      if (localStorage.hasOwnProperty(key)) {
        localStorage.removeItem(key);
      }
    }

    navigate('/');
  }
```

- All Products

Fetching Products

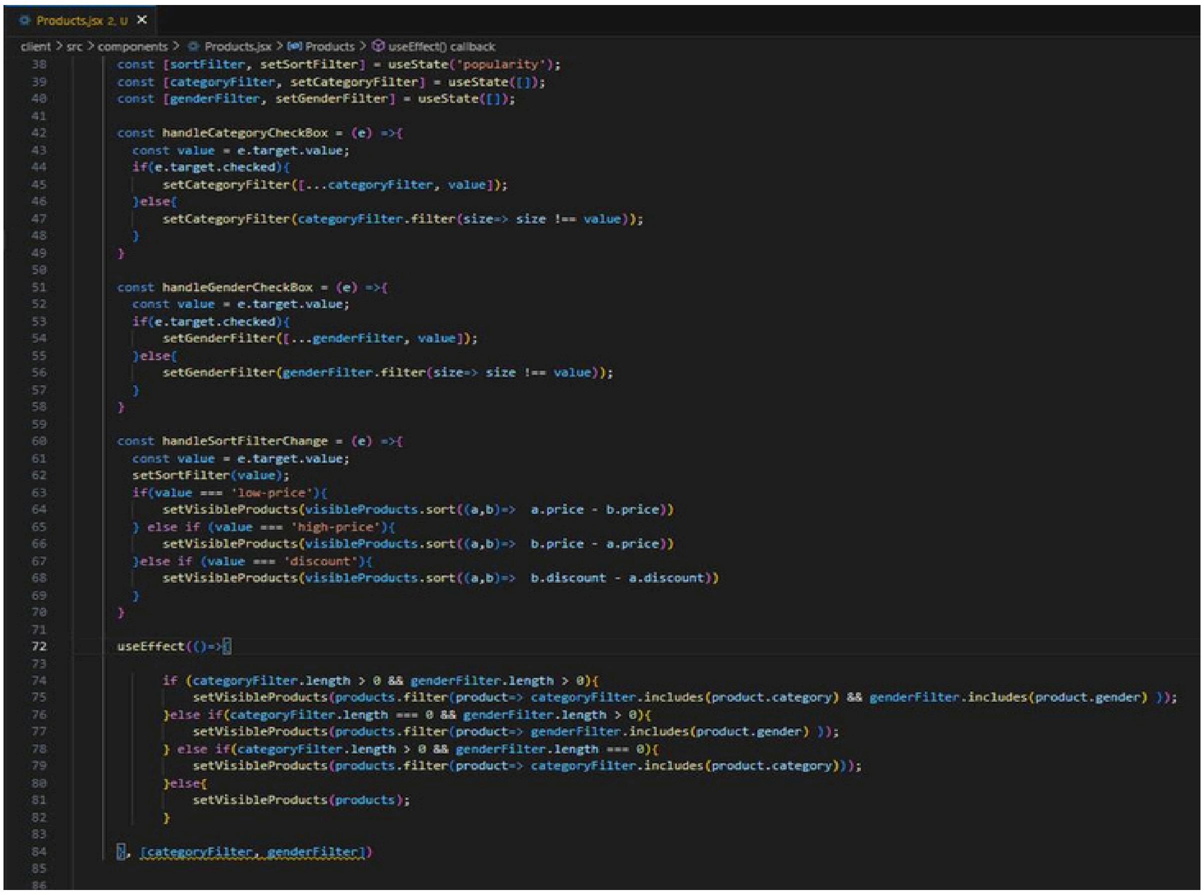


Products.jsx 2, U ×

client > src > components > Products.jsx > Products > handleCategoryCheckBox

```
    const [categories, setCategories] = useState([]);
    const [products, setProducts] = useState([]);
    const [visibleProducts, setVisibleProducts] = useState([]);

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

      const fetchData = async() =>{

        await axios.get('http://localhost:6001/fetch-products').then(
          (response)=>{
            if(props.category === 'all'){
                setProducts(response.data);
                setVisibleProducts(response.data);
            }else{
                setProducts(response.data.filter(product=> product.category === props.category));
                setVisibleProducts(response.data.filter(product=> product.category === props.category));
            }
          }
        )
        await axios.get('http://localhost:6001/fetch-categories').then(
          (response)=>{
            setCategories(response.data);
          }
        )
      }
```

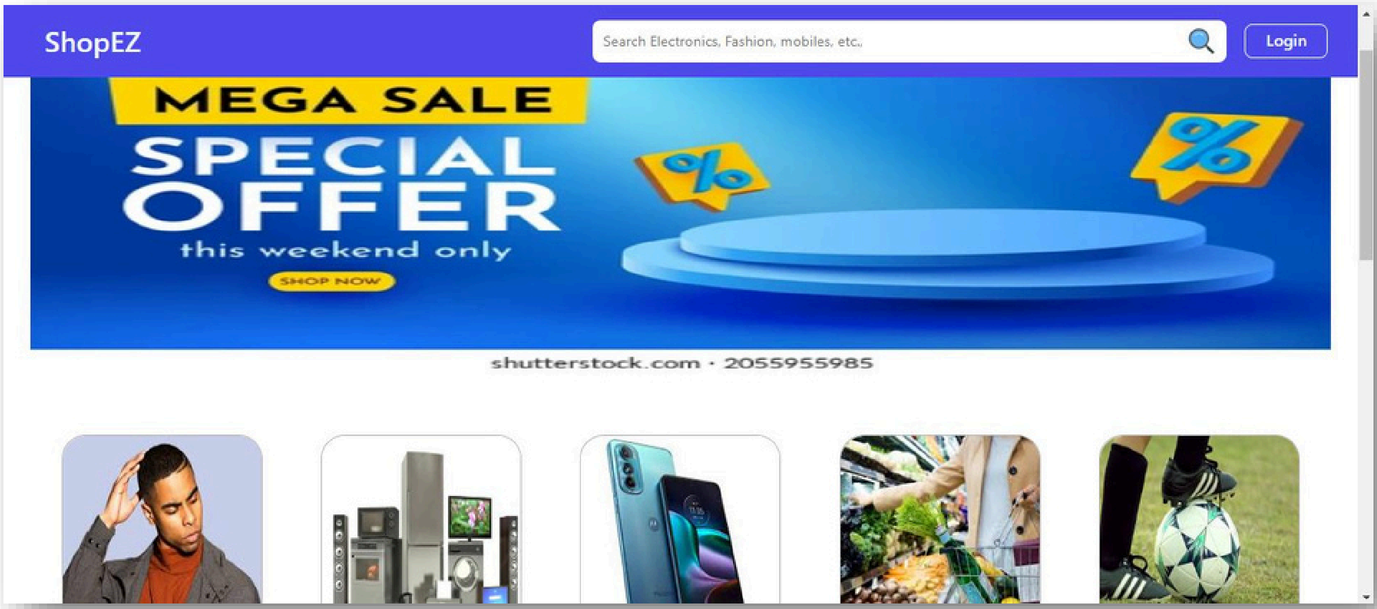
Filtering Products


Products.jsx 2, U
client > src > components > Products.jsx > Products > useEffect() callback

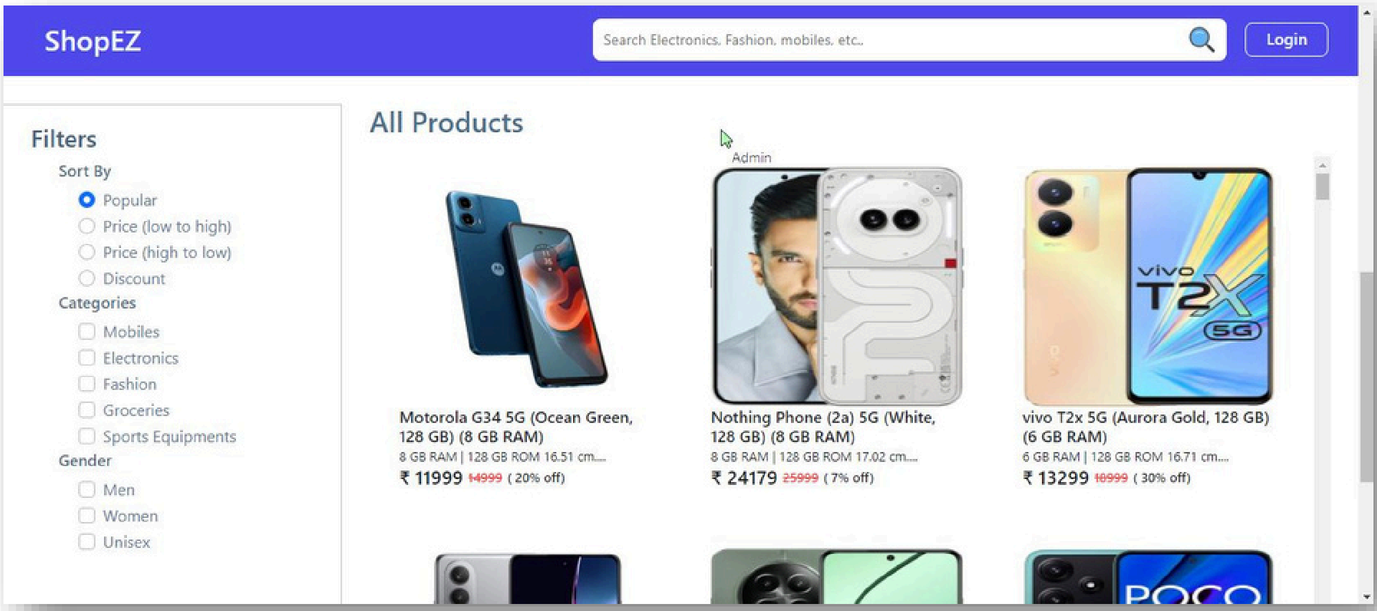
```
    const [sortFilter, setSortFilter] = useState('popularity');
    const [categoryFilter, setCategoryFilter] = useState([]);
    const [genderFilter, setGenderFilter] = useState([]);

    const handleCategoryCheckBox = (e) =>{
      const value = e.target.value;
      if(e.target.checked){
          setCategoryFilter([...categoryFilter, value]);
      }else{
          setCategoryFilter(categoryFilter.filter(size=> size !== value));
      }
    }

    const handleGenderCheckBox = (e) =>{
      const value = e.target.value;
      if(e.target.checked){
          setGenderFilter([...genderFilter, value]);
      }else{
          setGenderFilter(genderFilter.filter(size=> size !== value));
      }
    }

    const handleSortFilterChange = (e) =>{
      const value = e.target.value;
      setSortFilter(value);
      if(value === 'low-price'){
          setVisibleProducts(visibleProducts.sort((a,b)=>  a.price - b.price))
      } else if (value === 'high-price'){
          setVisibleProducts(visibleProducts.sort((a,b)=>  b.price - a.price))
      }else if (value === 'discount'){
          setVisibleProducts(visibleProducts.sort((a,b)=>  b.discount - a.discount))
      }
    }

    useEffect(()=>{

          if (categoryFilter.length > 0 && genderFilter.length > 0){
              setVisibleProducts(products.filter(product=> categoryFilter.includes(product.category) && genderFilter.includes(product.gender) ));
          }else if(categoryFilter.length === 0 && genderFilter.length > 0){
              setVisibleProducts(products.filter(product=> genderFilter.includes(product.gender) ));
          } else if(categoryFilter.length > 0 && genderFilter.length === 0){
              setVisibleProducts(products.filter(product=> categoryFilter.includes(product.category)));
          }else{
              setVisibleProducts(products);
          }

    }, [categoryFilter, genderFilter])
```

Demo UI Images

Landing Page


ShopEZ
Search Electronics, Fashion, mobiles, etc..
Login
MEGA SALE
SPECIAL OFFER
this weekend only
SHOP NOW
shutterstock.com · 2055955985

Products



ShopEZ

Search Electronics, Fashion, mobiles, etc.,

Login

Filters

Sort By

- [✓ Popular]
- Price (low to high)
- Price (high to low)
- Discount

Categories

- Mobiles
- Electronics
- Fashion
- Groceries
- Sports Equipments

Gender

- Men
- Women
- Unisex

All Products

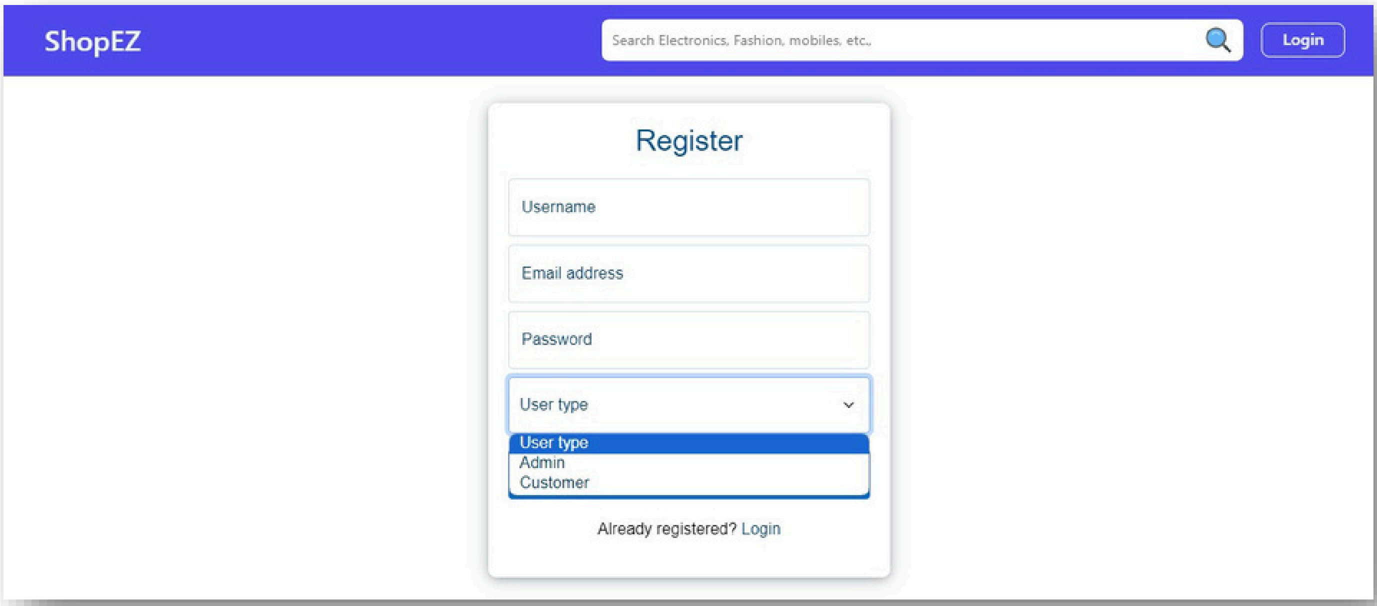
Admin

Motorola G34 5G (Ocean Green, 128 GB) (8 GB RAM)
8 GB RAM | 128 GB ROM 16.51 cm....
₹ 11999 ~~14999~~ (20% off)

Nothing Phone (2a) 5G (White, 128 GB) (8 GB RAM)
8 GB RAM | 128 GB ROM 17.02 cm....
₹ 24179 ~~25999~~ (7% off)

vivo T2x 5G (Aurora Gold, 128 GB) (6 GB RAM)
6 GB RAM | 128 GB ROM 16.71 cm....
₹ 13299 ~~18999~~ (30% off)

Authentication



ShopEZ

Search Electronics, Fashion, mobiles, etc.,

Login

Register

Username

Email address

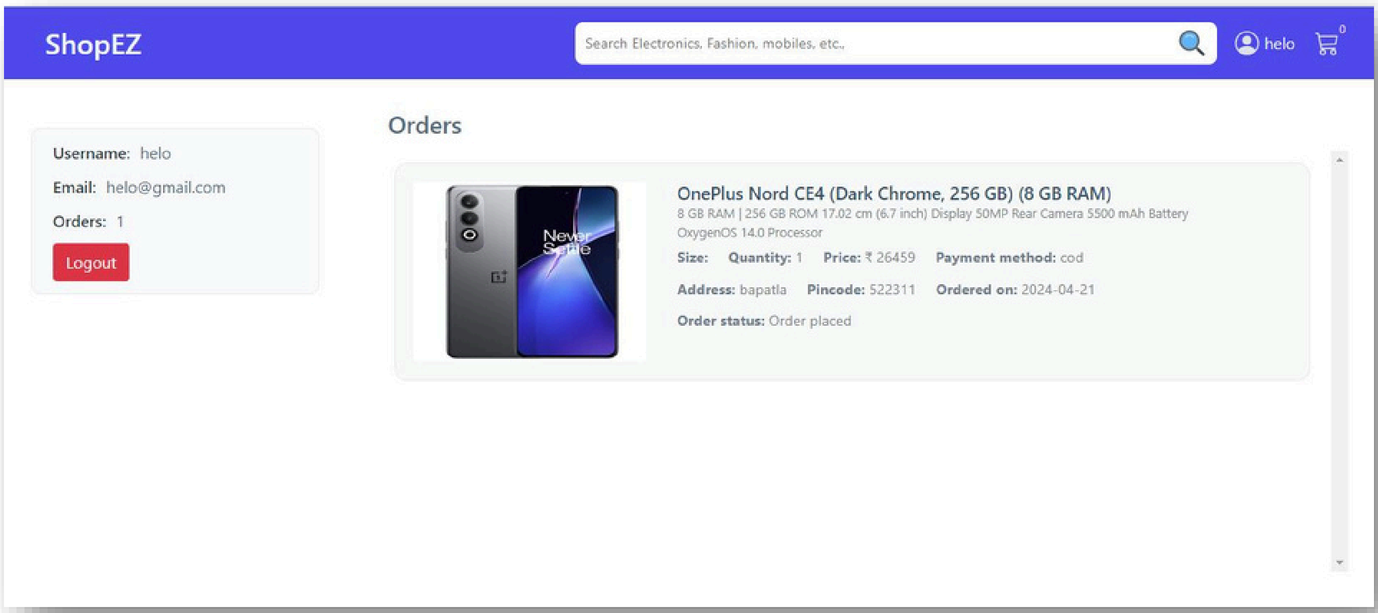
Password

User type

- User type
- Admin
- Customer

Already registered? Login

User Profile



ShopEZ

Search Electronics, Fashion, mobiles, etc.,

helo

Username: helo

Email: helo@gmail.com

Orders: 1

Logout

Orders

OnePlus Nord CE4 (Dark Chrome, 256 GB) (8 GB RAM)

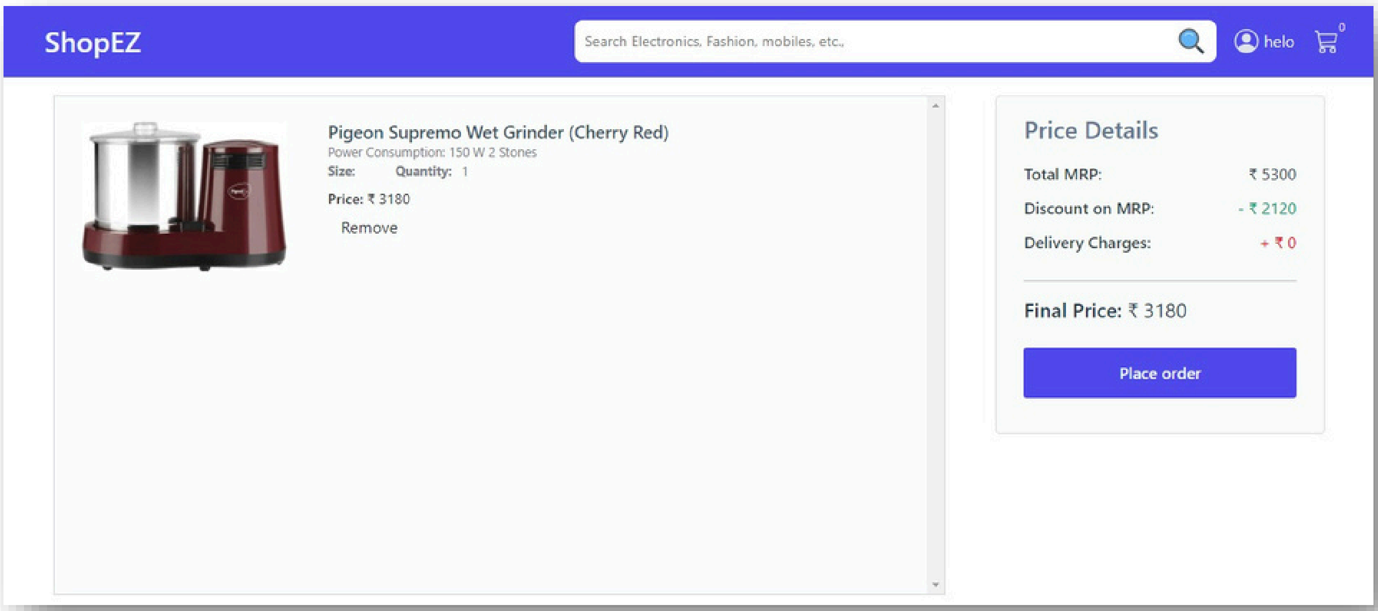
8 GB RAM | 256 GB ROM 17.02 cm (6.7 inch) Display 50MP Rear Camera 5500 mAh Battery OxygenOS 14.0 Processor

Size: **Quantity:** 1 **Price:** ₹ 26459 **Payment method:** cod

Address: bapatla **Pincode:** 522311 **Ordered on:** 2024-04-21

Order status: Order placed

Cart



ShopEZ

Search Electronics, Fashion, mobiles, etc.,

helo

Pigeon Supremo Wet Grinder (Cherry Red)

Power Consumption: 150 W 2 Stones

Size: **Quantity:** 1

Price: ₹ 3180

Remove

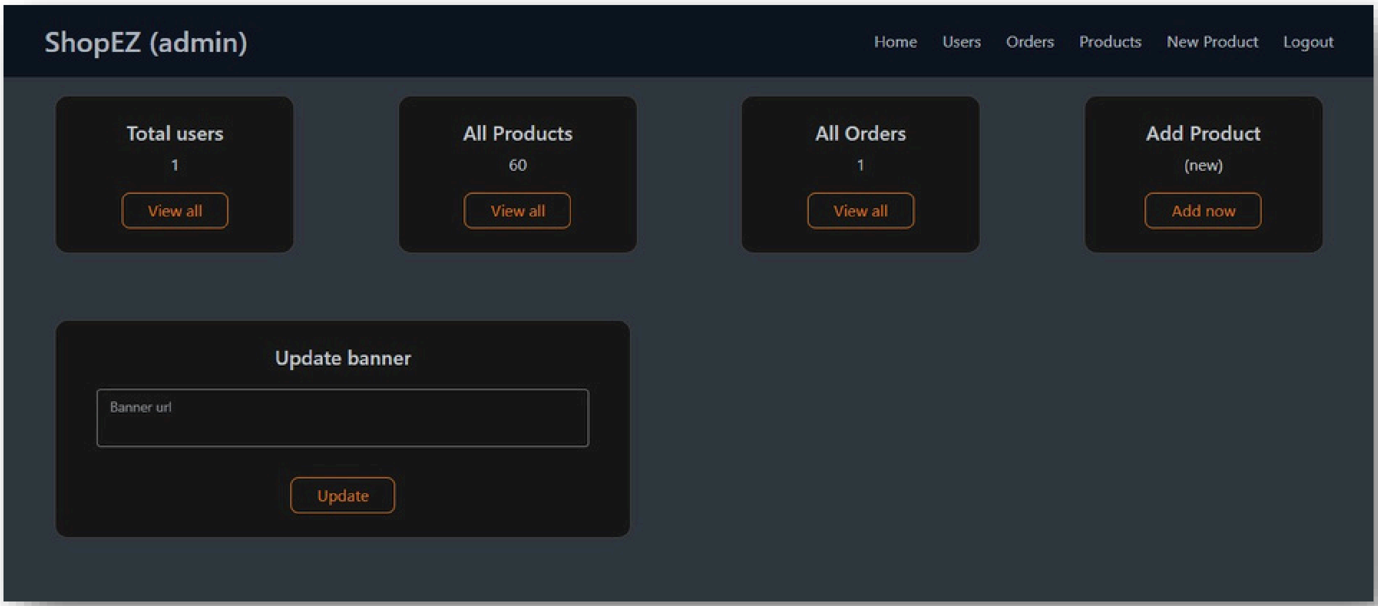
Price Details

Total MRP:	₹ 5300
Discount on MRP:	- ₹ 2120
Delivery Charges:	+ ₹ 0

Final Price: ₹ 3180

Place order

Admin Dashboard



Total users
1
View all

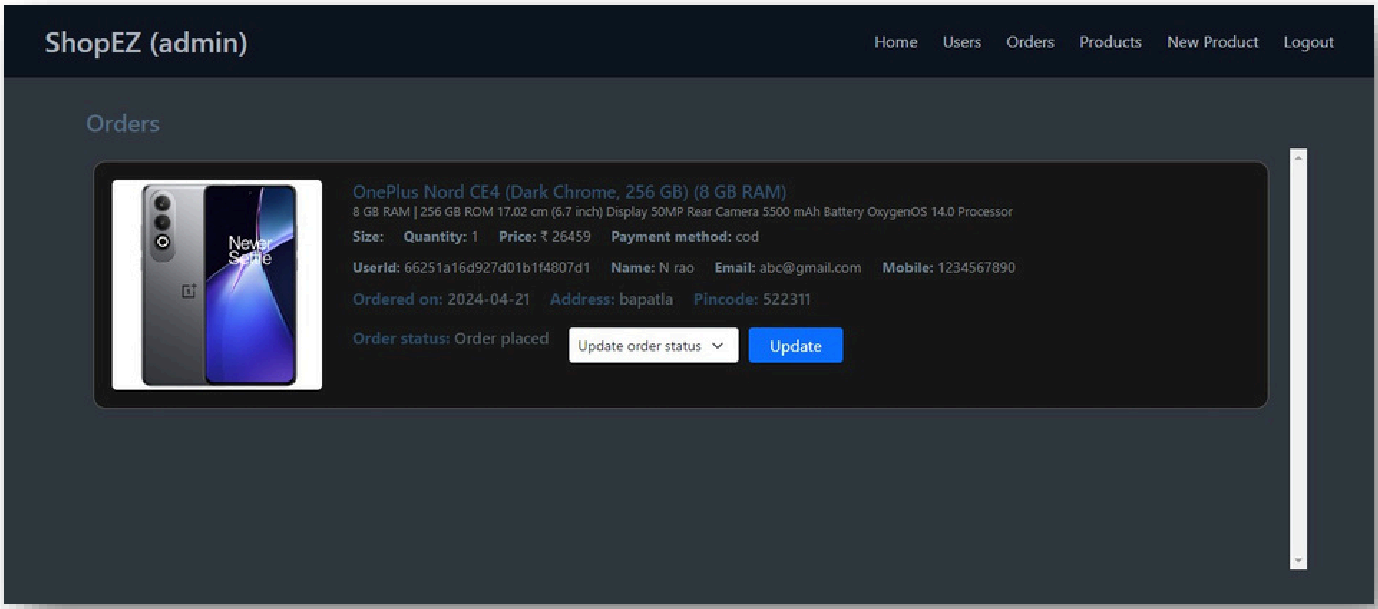
All Products
60
View all

All Orders
1
View all

Add Product
(new)
Add now

Update banner
Banner url
Update

All Orders



Orders

OnePlus Nord CE4 (Dark Chrome, 256 GB) (8 GB RAM)
8 GB RAM | 256 GB ROM 17.02 cm (6.7 inch) Display 50MP Rear Camera 5500 mAh Battery OxygenOS 14.0 Processor

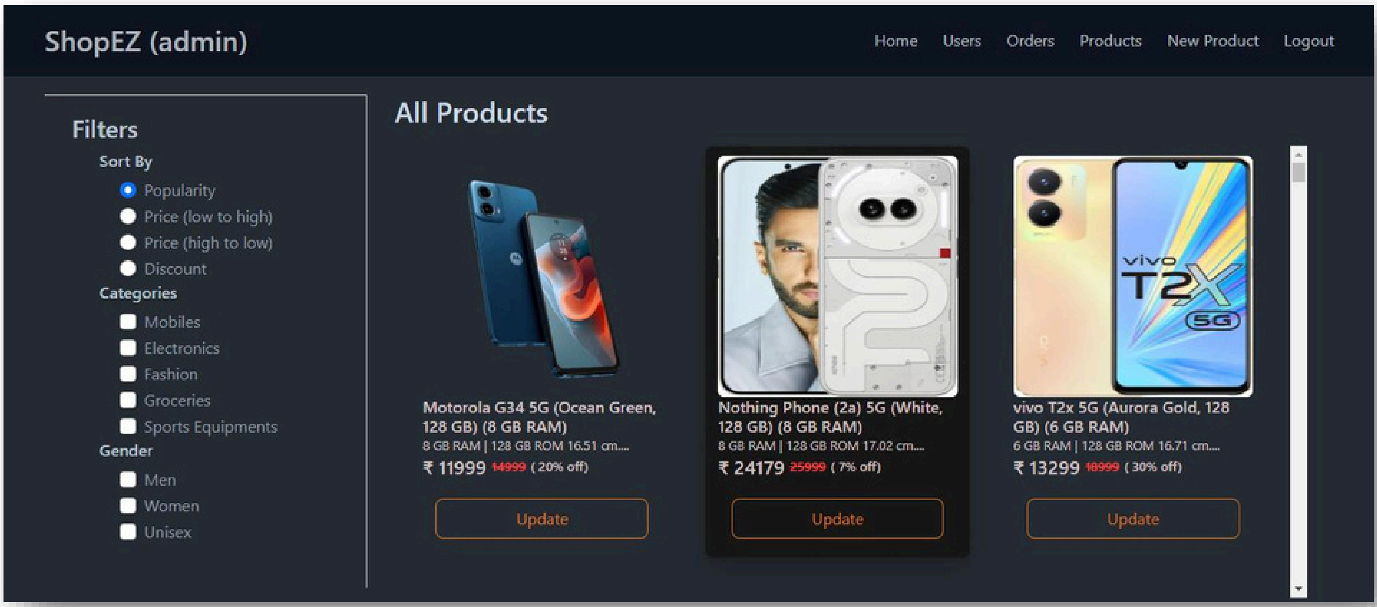
Size: **Quantity:** 1 **Price:** ₹ 26459 **Payment method:** cod

UserId: 66251a16d927d01b1f4807d1 **Name:** N rao **Email:** abc@gmail.com **Mobile:** 1234567890

Ordered on: 2024-04-21 Address: bapatla Pincode: 522311

Order status: Order placed Update order status Update

All Products



All Products

Motorola G34 5G (Ocean Green, 128 GB) (8 GB RAM)
8 GB RAM | 128 GB ROM 16.51 cm....
₹ 11999 ~~14999~~ (20% off)

Nothing Phone (2a) 5G (White, 128 GB) (8 GB RAM)
8 GB RAM | 128 GB ROM 17.02 cm....
₹ 24179 ~~25999~~ (7% off)

vivo T2x 5G (Aurora Gold, 128 GB) (6 GB RAM)
6 GB RAM | 128 GB ROM 16.71 cm....
₹ 13299 ~~18999~~ (30% off)

Conclusion

In conclusion, Shopez is a versatile e-commerce application that offers a seamless shopping experience for customers. The platform continues to innovate and enhance its features to meet the evolving needs of users and stay competitive in the market. With a focus on customer satisfaction, product variety, and secure transactions, Shopez aims to become a leading e-commerce destination for online shoppers. Shopez is a fantastic e-commerce application that offers a seamless shopping experience. It allows you to explore a wide variety of products and make purchases with ease. With detailed product descriptions and customer reviews, you can make informed decisions. The app provides secure payment options and reliable delivery services. Shopez simplifies the process of shopping, allowing you to browse, search, and purchase items from the comfort of your own home. It's a convenient and efficient way to fulfill all your shopping needs. So why wait? Start exploring Shopez and enjoy the joys of online shopping.