

Sri Lanka Institute of Information Technology

**Agri product online purchasing platform**

**Report**

Distributed System

Assignment 02

**Group ID: 2022S1\_REG\_65**

Submitted by:

1. IT20221928– Y.Nirojan
2. IT20223458- T.Arivaran
3. IT20202422 -L.Lavaniyah
4. IT20118822 – K.Sayanthan

**Introduction**

**“Agri Go”** is an online purchasing platform that allows customers and sellers to satisfy their shopping demands by buying and selling things over the Internet. This online platform allows customers to simply look for and purchase one or more products.

This application is developed using Node.js and express.js for the backend services while React Js and Material UI for the frontend development. Redux is used to manage state throughout the client-side application. Mongo dB has been used as the database. In order to get services from backend, user needs to send a token along with each request which is provided when logging in the system, users with the valid token will be allowed to get services. For this purpose, we have used jsonwebtoken library.

Our application “Agri Go” is User-friendly Website where buyers can shop for items uploaded by farmers Who can add/update/delete items. In Our system, there is a search bar that allows users to find what they're looking for quickly and buy the items which they need. Buyers may buy multiple items. Here in our system Once an item or a collection of items is added to the cart, the buyer can select the delivery option, where a request may be sent to a delivery service. The payment for the items bought can be made using credit cards or using the mobile phone number which would be added to the user’s mobile bill. Our system connects to a payment gateway for easy credit card transactions. For Security purposes, the information that should be transferred includes the mobile phone number, a six-digit pin number, and for this purpose the amount to be charged. Once the payment is made the user should be given a confirmation of the appointment via and email.

Several authentications and security mechanisms have been implemented to verify the security and reliability of the shopping platform. Each request from client is gone through some middleware where the token is being checked for its validity. If the token is not valid, then a response with status code 404 will send to client. Some services are restricted only for farmers and the other users cannot access those services.

We developed a good User Interface that can turn potential visitors into buyers as it facilitates interaction between the user and your website or application. It does not only focus only on aesthetics but also maximizes responsiveness, efficiency, and accessibility of a website.

This report will elaborate on the functionalities of this “**Agri Go**" is an online purchasing platform with the aid of diagrams.

**Tools And Technologies**

1. Web Client

* ReactJS
* Material UI
* Redux
* VS Code

1. Rest Api

* NodeJS
* Express (Framework)
* VS Code

1. Stripe payment gateway
2. JSON Web Token
3. Mongo DB [Mongoose]

**High Level Architectural Diagram**

**Services**

1. **User Services**

The front of the web application is developed using React, Redux, and Material UI libraries. The backend of the application communicates using the Axios library. The Screenshot below shows how the buyer component functions.

Text

Description automatically generated

1. **Product service**

The interfaces for the buyer and seller are different. When users request the item service application, they will be sent to pages based on their function and will be able to execute related duties.

1. **GetProducts() –**This API is used to show all of the items on the home page, along with their descriptions. The item's name, description, price, and an image of each item are all included. It also works to serve the searched items.

Text

Description automatically generated

1. Text

   Description automatically generated**GetProduct() –** This API is used to show the item that the seller/client wants to view. This API uses the item id argument, which is automatically assigned to an item when it is added by the seller.
2. **NewProduct() –**This API allows a seller to add a new item to the database and sell it. The item object parameter is used to add a new item to the database using this API.

Text

Description automatically generated

1. **DeleteProduct() –**This API is implemented for the sellers to delete the items as they prefer.

Text

Description automatically generated

1. **UpdateProduct() –**This API is implemented to change the information of an existing item The item id option is used to locate a specific item so that the seller may change its data.

Text

Description automatically generated

1. **Order service**
2. **AddOrder()**- This API is implemented for a new order to be added.

A screenshot of a computer

Description automatically generated with medium confidence

1. Text

   Description automatically generated**GetOrders()**- This API is implemented to get all the order details for the seller to view. Address, city, and zip code of the delivery will be displayed by the API for the seller to carry out the deliveries.
2. **Payment service**

Credit cards or the user's mobile phone number can be used to pay for the things purchased (which will be added to the user's mobile bill). For credit card transactions, the system can link to a payment gateway. The credit card number, amount, CVC number (3-digit number on the back of the credit card), and card holder's name should all be given.

Users can opt to credit their mobile bill with a payment made through a mobile company's service. The mobile phone number, a six-digit pin number, and the amount to be charged are all items that should be transferred. Following payment, the customer should get an appointment confirmation by SMS and email.

1. **Authentication / Security Mechanism**

To improve the level of security and protection of the application, we have developed several security mechanisms using Jsonwebtoken library.

1. **User Authentication**

When entering to the application the users must be registered to the system with valid data after that they have to give their credentials (Username and password) to enter the system. A token that in encrypted from will be given to the client after logging to the system. Every request for any services from the server should carry that token to validate the authentication and authorization.

1. **User Authorization**

There are two categories of users who utilize our system, according to our system. They are farmers and customers, respectively. They will be sent to their appropriate home pages based on their username and password. Buyers may only make changes to their orders, but sellers can make changes to the things they're selling, both can change their details. Some pages are restricted to either user according to their role in the system.

**GitHub Link:** [**https://github.com/Nirojan-99/DS-Agri-web.git**](https://github.com/Nirojan-99/DS-Agri-web.git)

**Appendix-client**

**Ack.js**

import {

  Button,

  Dialog,

  DialogActions,

  DialogContent,

  DialogContentText,

  DialogTitle,

  Slide,

} from "@mui/material";

import \* as React from "react";

const Transition = React.forwardRef(function Transition(props, ref) {

  return <Slide direction="up" ref={ref} {...props} />;

});

function Ack(props) {

  return (

    <div>

      <Dialog

        open={props.open}

        TransitionComponent={Transition}

        keepMounted

        onClose={props.handleClose}

      >

        <DialogTitle>{props.title}</DialogTitle>

        <DialogContent>

          <DialogContentText>{props.msg}</DialogContentText>

        </DialogContent>

        <DialogActions>

          <Button onClick={props.handleYes}>Yes</Button>

          <Button onClick={props.handleClose}>No</Button>

        </DialogActions>

      </Dialog>

    </div>

  );

}

export default Ack;

**Alert.js**

import {

  Button,

  Dialog,

  DialogActions,

  DialogContent,

  DialogContentText,

  DialogTitle,

  Slide,

} from "@mui/material";

import \* as React from "react";

const Transition = React.forwardRef(function Transition(props, ref) {

  return <Slide direction="up" ref={ref} {...props} />;

});

function Alert(props) {

  return (

    <>

      <Dialog

        open={props.open}

        TransitionComponent={Transition}

        keepMounted

        onClose={props.handleClose}

      >

        <DialogTitle>{props.title}</DialogTitle>

        <DialogContent>

          <DialogContentText>{props.msg}</DialogContentText>

        </DialogContent>

        <DialogActions>

          <Button onClick={props.handleClose}>OK</Button>

        </DialogActions>

      </Dialog>

    </>

  );

}

export default Alert;

**Footer.js**

import { Paper, Typography, Button, Divider, Grid } from "@mui/material";

import MessageIcon from "@mui/icons-material/Message";

function Footer() {

  return (

    <>

      <Divider />

      <Grid

        square

        elevation={2}

        component={Paper}

        sx={{ py: 2, px: 1 }}

        container

        justifyContent={"center"}

        alignItems="center"

      >

        <Grid item xs={12} sm={9}>

          <Typography

            variant="subtitle2"

            textAlign={{ xs: "center", sm: "left" }}

          >

            © 2022, made with ❤️ by Silicon Team for a better shopping . All

            copyrights reserved

          </Typography>

        </Grid>

        <Grid item xs={12} sm={3} textAlign={{ xs: "center", sm: "right" }}>

          <Button endIcon={<MessageIcon />} variant="text" href="/contact-us">

            Contact Us

          </Button>

        </Grid>

      </Grid>

    </>

  );

}

export default Footer;

**Header.js**

import { useState } from "react";

import {

  Box,

  AppBar,

  Toolbar,

  IconButton,

  Badge,

  Menu,

  MenuItem,

  Tooltip,

  Button,

  Divider,

  Typography,

} from "@mui/material";

import AccountCircle from "@mui/icons-material/AccountCircle";

import DarkModeIcon from "@mui/icons-material/DarkMode";

import LightModeIcon from "@mui/icons-material/LightMode";

import ShoppingCartIcon from "@mui/icons-material/ShoppingCart";

import FavoriteIcon from "@mui/icons-material/Favorite";

import { dark, light } from "../Store/theme";

import logo from "../Assets/logo.png";

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

import { useSelector, useDispatch } from "react-redux";

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

import axios from "axios";

import { useEffect } from "react";

function Header(props) {

  // user data

  const { token, role, userID } = useSelector((state) => state.loging);

  const [auth, setAuth] = useState(token);

  //theme handler

  const Tmode = useSelector((state) => state.mode.mode);

  const [mode, setMode] = useState(Tmode);

  const [anchorEl, setAnchorEl] = useState(null);

  //hooks

  const dispatch = useDispatch();

  let history = useNavigate();

  //cart count

  const [cart, setcart] = useState(0);

  //useEffect call

  useEffect(() => {

    axios

      .get(`http://localhost:5000/users/carts?\_id=${userID}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        setcart(res.data.length);

      });

  }, []);

  const handleClose = () => {

    setAnchorEl(null);

  };

  props.handler(mode);

  const handleMenu = (event) => {

    setAnchorEl(event.currentTarget);

  };

  const profilehandler = () => {

    history("/profile");

  };

  return (

    <>

      <Box sx={{ flexGrow: 1 }}>

        <AppBar color={"primary"} position="sticky">

          <Toolbar>

            <Button href="/">

              <img alt="Agri logo" src={logo} width="50px" />

              <Typography

                variant="h3"

                sx={{

                  textTransform: "none",

                  ml: 2,

                  fontSize: { xs: 14, sm: 25 },

                }}

              >

                AgriGo

              </Typography>

            </Button>

            <Box sx={{ flexGrow: 1 }} />

            <Box sx={{ display: { xs: "none", md: "flex" } }}></Box>

            <Tooltip title={mode === "dark" ? "light theme" : "dark theme"}>

              <IconButton

                size="large"

                aria-controls="menu-appbar"

                aria-haspopup="true"

                color="inherit"

                onClick={() => {

                  setMode((prevMode) =>

                    prevMode === "light" ? "dark" : "light"

                  );

                  if (Tmode === "light") {

                    dispatch(dark());

                  } else {

                    dispatch(light());

                  }

                }}

              >

                {mode === "light" ? (

                  <DarkModeIcon fontSize="inherit" />

                ) : (

                  <LightModeIcon fontSize="inherit" />

                )}

              </IconButton>

            </Tooltip>

            {!auth && (

              <Box sx={{ flexGrow: 0, display: { xs: "flex", md: "flex" } }}>

                <Button href="/login" key={"login"} sx={{ color: "#eee" }}>

                  Log In

                </Button>

                <Button href="/signup" key={"signup"} sx={{ color: "#eee" }}>

                  Sign up

                </Button>

              </Box>

            )}

            {/\*user profile\*/}

            {auth && (

              <div>

                {role === "client" && (

                  <Tooltip title={"cart"}>

                    <IconButton href="/cart" size="large" color="inherit">

                      <Badge badgeContent={cart} color="error">

                        <ShoppingCartIcon fontSize="inherit" />

                      </Badge>

                    </IconButton>

                  </Tooltip>

                )}

                {role === "client" && (

                  <Tooltip title={"Favorites"}>

                    <IconButton href="/favorites" size="large" color="inherit">

                      <FavoriteIcon fontSize="inherit" />

                    </IconButton>

                  </Tooltip>

                )}

                <Tooltip title="profile">

                  <IconButton size="large" onClick={handleMenu} color="inherit">

                    <AccountCircle fontSize="inherit" />

                  </IconButton>

                </Tooltip>

                <Menu

                  color="#e28743"

                  sx={{ mt: "45px" }}

                  id="menu-appbar"

                  anchorEl={anchorEl}

                  anchorOrigin={{

                    vertical: "top",

                    horizontal: "right",

                  }}

                  keepMounted

                  transformOrigin={{

                    vertical: "bottom",

                    horizontal: "right",

                  }}

                  open={Boolean(anchorEl)}

                  onClose={handleClose}

                >

                  <MenuItem onClick={profilehandler}>Profile</MenuItem>

                  <MenuItem

                    onClick={() => {

                      setAuth(null);

                      dispatch(logout());

                      history("/login", { replace: true });

                    }}

                  >

                    Log Out

                  </MenuItem>

                </Menu>

              </div>

            )}

          </Toolbar>

        </AppBar>

      </Box>

      <Divider />

    </>

  );

}

export default Header;

**PageNotFound.js**

import { Box, Paper, Typography } from "@mui/material";

import Header from "../../Components/Header";

function PageNotFound(props) {

  return (

    <>

      <Header handler={props.handler} />

      <Box

        component={Paper}

        elevation={1}

        square

        minHeight={"82vh"}

        sx={{ display: "flex", flexDirection: "column" }}

      >

        <Typography

          variant="h3"

          textAlign={"center"}

          sx={{ mt: 6, fontFamily: "open sans", color: "#62BB46" }}

        >

          Page Not Found

        </Typography>

      </Box>

    </>

  );

}

export default PageNotFound;

**SignIn.js**

import Avatar from "@mui/material/Avatar";

import Button from "@mui/material/Button";

import TextField from "@mui/material/TextField";

import Link from "@mui/material/Link";

import Paper from "@mui/material/Paper";

import Box from "@mui/material/Box";

import Grid from "@mui/material/Grid";

import Typography from "@mui/material/Typography";

import Header from "../../../Components/Header";

import AgricultureIcon from "@mui/icons-material/Agriculture";

import { useDispatch } from "react-redux";

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

import Alert from "../../../Components/Alert";

import axios from "axios";

import AgriSnackbar from "../../Utils/AgriSnackbar";

import { useState } from "react";

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

export default function SignInSide(props) {

  //data

  const [snack, setSnack] = useState(false);

  const [error, setError] = useState("");

  const [email, setEmail] = useState("");

  const [password, setPassword] = useState("");

  //hooks

  const dispatch = useDispatch();

  const navigate = useNavigate();

  //error display

  const handleClose = () => {

    setSnack(false);

  };

  //login submit

  const handleSubmit = (event) => {

    event.preventDefault();

    if (!email.trim() || !email.includes("@")) {

      setError("Invalid Email");

      return;

    }

    if (!password.trim()) {

      setError("Password Required");

      return;

    }

    axios

      .post("http://localhost:5000/users/login", {

        email: email,

        password: password,

      })

      .then((res) => {

        dispatch(

          login({

            role: res.data.type,

            userID: res.data.\_id,

            token: res.data.token,

          })

        );

        navigate("/dashboard", { replace: true });

      })

      .catch((er) => {

        setSnack(true);

      });

  };

  //close alert msg

  const handleCloseAlert = () => {

    setError("");

  };

  return (

    <>

      <Alert

        open={error}

        handleClose={handleCloseAlert}

        title="Alert"

        msg={error}

      />

      <AgriSnackbar open={snack} msg={"Login Failure"} handler={handleClose} />

      <Header mode={props.mode} handler={props.handler} />

      <Grid container sx={{ height: "83vh" }}>

        <Grid

          item

          xs={false}

          sm={4}

          md={7}

          sx={{

            backgroundImage:

              "url(https://cdn.pixabay.com/photo/2016/09/21/04/46/barley-field-1684052\_\_480.jpg)",

            backgroundRepeat: "no-repeat",

            backgroundSize: "cover",

            backgroundPosition: "center",

          }}

        />

        <Grid item xs={12} sm={8} md={5} component={Paper} elevation={0} square>

          <Box

            sx={{

              my: 8,

              mx: 4,

              display: "flex",

              flexDirection: "column",

              alignItems: "center",

            }}

          >

            <Avatar sx={{ m: 1, bgcolor: "secondary.main" }}>

              <AgricultureIcon fontSize="medium" />

            </Avatar>

            <Typography component="h1" variant="h5">

              Sign in

            </Typography>

            <Box

              component="form"

              noValidate

              onSubmit={handleSubmit}

              sx={{ mt: 1 }}

            >

              <TextField

                margin="normal"

                required

                fullWidth

                value={email}

                onChange={(event) => {

                  setEmail(event.target.value);

                }}

                id="email"

                label="Email ID"

                name="email"

                autoComplete="email"

                autoFocus

              />

              <TextField

                margin="normal"

                required

                fullWidth

                value={password}

                onChange={(event) => {

                  setPassword(event.target.value);

                }}

                name="password"

                label="Password"

                type="password"

                id="password"

                autoComplete="current-password"

              />

              <Button

                type="submit"

                fullWidth

                variant="contained"

                sx={{ mt: 3, mb: 2 }}

              >

                Sign In

              </Button>

              <Grid container>

                <Grid item xs sx={{ textAlign: "left" }}>

                  <Link href="/forget-password" variant="body2">

                    Forgot password?

                  </Link>

                </Grid>

                <Grid item>

                  <Link href="/signup" variant="body2">

                    {"Don't have an account? Sign Up"}

                  </Link>

                </Grid>

              </Grid>

            </Box>

          </Box>

        </Grid>

      </Grid>

    </>

  );

}

**Signup.js**

import {

  Paper,

  Typography,

  Box,

  Grid,

  Link,

  TextField,

  Button,

  Avatar,

  Container,

} from "@mui/material";

import AgricultureIcon from "@mui/icons-material/Agriculture";

import axios from "axios";

import { useState } from "react";

import { useDispatch } from "react-redux";

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

import Header from "../../../Components/Header";

import AgriSnakbar from "../../Utils/AgriSnackbar";

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

import Alert from "../../../Components/Alert";

export default function SignUp(props) {

  //data

  const [open, setOpen] = useState(false);

  const [error, setError] = useState("");

  //input darta

  const [FName, setFName] = useState("");

  const [LName, setLName] = useState("");

  const [email, setEmail] = useState("");

  const [password, setPassword] = useState("");

  const [mobile, setMobile] = useState("");

  //hooks

  const dispatch = useDispatch();

  const navigate = useNavigate();

  //close snackbar

  const closeHandler = () => {

    setOpen(false);

  };

  //submit form

  const handleSubmit = (event) => {

    event.preventDefault();

    //validation

    if (!FName.trim() || !LName.trim()) {

      setError("Name Required");

      return;

    }

    if (!email.trim() || !email.includes("@")) {

      setError("Invalid Email ID");

      return;

    }

    if (!password.trim()) {

      setError("Password Required");

      return;

    }

    if (

      !mobile.trim() ||

      isNaN(mobile) ||

      mobile.length > 10 ||

      mobile.length < 9

    ) {

      setError("Invalid Mobile Number");

      return;

    }

    axios

      .post("http://localhost:5000/users", {

        firstName: FName,

        lastName: LName,

        email: email,

        password: password,

        mobile\_number: mobile,

      })

      .then((res) => {

        dispatch(

          login({

            role: res.data.type,

            userID: res.data.\_id,

            token: res.data.token,

          })

        );

        navigate("/dashboard", { replace: true });

      })

      .catch((er) => {

        setOpen(true);

      });

  };

  //close alert

  const handleClose = () => {

    setError("");

  };

  return (

    <>

      <Alert

        handleClose={handleClose}

        title="Alert!"

        open={error}

        msg={error}

      />

      <AgriSnakbar

        msg={"Unable to create Account"}

        open={open}

        handler={closeHandler}

      />

      <Header mode={props.mode} handler={props.handler} />

      <Box minHeight={"83vh"} component="div">

        <Paper square elevation={0}>

          <Container component="main" maxWidth="xs" color="primary">

            <Box

              pb={3}

              component={"paper"}

              sx={{

                minHeight: "83vh",

                display: "flex",

                flexDirection: "column",

                alignItems: "center",

              }}

            >

              <Avatar sx={{ m: 1, mt: 5, bgcolor: "secondary.main" }}>

                <AgricultureIcon fontSize="medium" />

              </Avatar>

              <Typography component="h1" variant="h5">

                Sign up

              </Typography>

              <Box

                component="form"

                noValidate

                onSubmit={handleSubmit}

                sx={{ mt: 3 }}

              >

                <Grid container spacing={2}>

                  <Grid item xs={12} sm={6}>

                    <TextField

                      autoComplete="given-name"

                      name="firstName"

                      required

                      value={FName}

                      onChange={(event) => {

                        setFName(event.target.value);

                      }}

                      fullWidth

                      id="firstName"

                      label="First Name"

                      autoFocus

                    />

                  </Grid>

                  <Grid item xs={12} sm={6}>

                    <TextField

                      required

                      value={LName}

                      onChange={(event) => {

                        setLName(event.target.value);

                      }}

                      fullWidth

                      id="lastName"

                      label="Last Name"

                      name="lastName"

                      autoComplete="family-name"

                    />

                  </Grid>

                  <Grid item xs={12}>

                    <TextField

                      required

                      value={email}

                      onChange={(event) => {

                        setEmail(event.target.value);

                      }}

                      fullWidth

                      id="email"

                      label="Email Address"

                      name="email"

                      autoComplete="email"

                    />

                  </Grid>

                  <Grid item xs={12}>

                    <TextField

                      required

                      value={mobile}

                      onChange={(event) => {

                        setMobile(event.target.value);

                      }}

                      fullWidth

                      id="mobile-number"

                      label="mobile number"

                      name="mobile-number"

                      type="number"

                      autoComplete="mobile-number"

                    />

                  </Grid>

                  <Grid item xs={12}>

                    <TextField

                      required

                      value={password}

                      onChange={(event) => {

                        setPassword(event.target.value);

                      }}

                      fullWidth

                      name="password"

                      label="Password"

                      type="password"

                      id="password"

                      autoComplete="new-password"

                    />

                  </Grid>

                </Grid>

                <Button

                  type="submit"

                  fullWidth

                  variant="contained"

                  sx={{ mt: 3, mb: 2 }}

                >

                  Sign Up

                </Button>

                <Grid container justifyContent="flex-end">

                  <Grid item>

                    <Link href="/login" variant="body2">

                      Already have an account? Sign in

                    </Link>

                  </Grid>

                </Grid>

              </Box>

            </Box>

          </Container>

        </Paper>

      </Box>

    </>

  );

}

**Cart.js**

import {

  Box,

  Button,

  Container,

  Divider,

  Grid,

  Paper,

  Typography,

} from "@mui/material";

import Header from "../../Components/Header";

import AgriCart from "../Utils/AgriCart";

import ShoppingCartCheckoutOutlinedIcon from "@mui/icons-material/ShoppingCartCheckoutOutlined";

import { useEffect, useState } from "react";

import axios from "axios";

import { useSelector } from "react-redux";

import CartSkelton from "../Utils/CartSkelton";

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

import { useDispatch } from "react-redux";

function Cart(props) {

  //data

  const { userID, token } = useSelector((state) => state.loging);

  const [cart, setCart] = useState([]);

  const [isLoaded, setLoaded] = useState(false);

  const [total, setTotal] = useState(0);

  const [cartObj, setCartObj] = useState({});

  //hook

  const dispatch = useDispatch();

  const navigate = useNavigate();

  //checkout

  const checkOutHandler = () => {

    console.log(cartObj);

    axios

      .post(

        `http://localhost:5000/api/orders`,

        {

          products: cartObj,

          user\_id: userID,

          total,

        },

        {

          headers: { Authorization: "Agriuservalidation " + token },

        }

      )

      .then((res) => {

        navigate(`/checkout/${res.data.\_id}`, { replace: true });

      })

      .catch((er) => {});

  };

  //quantity Handler

  const quantityHandler = (operation, price, id, amount) => {

    setCartObj((pre) => {

      let obj = { ...pre, [id]: amount };

      return obj;

    });

    if (operation === "inc") {

      setTotal((pre) => {

        let val;

        val = pre \* 100 + price \* 100;

        return val / 100;

      });

    } else {

      setTotal((pre) => (pre \* 100 - price \* 100) / 100);

    }

  };

  //total amount calculator

  const calTotal = () => {

    let total = 0;

    cart.forEach((row) => {

      total += row.price;

    });

    return total;

  };

  //remove cart element

  const removeCart = (index, id) => {

    axios

      .delete(`http://localhost:5000/users/carts?id=${userID}&pid=${id}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        setTotal((pre) => {

          let data = +pre \* 100;

          let minus = +cartObj[id];

          data -= minus \* (+cart[index].price \* 100);

          return data / 100;

        });

        setCartObj((pre) => {

          console.log(id);

          const { [id]: dummy, ...objectWithoutDeleted } = pre;

          return objectWithoutDeleted;

        });

        setCart((pre) => {

          const array = [...pre];

          array.splice(index, 1);

          return array;

        });

      })

      .catch((er) => {});

  };

  //useEffect call

  useEffect(() => {

    axios

      .get(`http://localhost:5000/users/carts?\_id=${userID}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        axios

          .get(

            `http://localhost:5000/api/products?pagination=${1}&favList=${

              res.data.length != 0 ? res.data : ["1", "2"]

            }`,

            {

              headers: { Authorization: "Agriuservalidation " + token },

            }

          )

          .then((res) => {

            if (res.data) {

              setCart(res.data.data);

              setLoaded(true);

              setCartObj((pre) => {

                let obj = [];

                for (let i = 0; i < res.data.data.length; i++) {

                  obj = { ...obj, [res.data.data[i].\_id]: 1 };

                }

                return obj;

              });

              setTotal((pre) => {

                let data = 0;

                for (let i = 0; i < res.data.data.length; i++) {

                  data += +res.data.data[i].price \* 100;

                }

                return data / 100;

              });

            }

          })

          .catch((er) => {

            setLoaded(true);

          });

      })

      .catch((er) => {});

  }, []);

  return (

    <>

      <Header mode={props.mode} handler={props.handler} />

      <Box py={1} component={Paper} elevation={0} square minHeight={"65vh"}>

        <Container maxWidth="md">

          <Grid

            maxWidth="md"

            container

            justifyContent="center"

            alignItems={"center"}

            spacing={1}

            sx={{ textAlign: "left" }}

          >

            <Grid

              item

              xs={6}

              component={Typography}

              variant="h3"

              fontFamily={"roboto"}

              letterSpacing={1}

              sx={{ color: "#62BB46", my: 2 }}

            >

              Your Cart

            </Grid>

            <Grid

              item

              xs={6}

              component={Typography}

              variant="h4"

              fontFamily={"roboto"}

              letterSpacing={1}

              sx={{ color: "#62BB46", my: 2, textAlign: "right" }}

            >

              {`Total : $${total}`}

            </Grid>

          </Grid>

        </Container>

        <Divider />

        <Container maxWidth="md" sx={{ py: 1 }}>

          <Box minHeight="60vh">

            {isLoaded ? (

              cart.map((row, index) => {

                return (

                  <AgriCart

                    index={index}

                    removeCart={removeCart}

                    key={row.\_id}

                    data={row}

                    quantityHandler={quantityHandler}

                  />

                );

              })

            ) : (

              <>

                <CartSkelton />

                <CartSkelton />

              </>

            )}

            {isLoaded && cart.length === 0 && (

              <Box mt={3}>

                <Typography textAlign={"center"} sx={{ color: "#4d9537" }}>

                  Cart is Empty

                </Typography>

              </Box>

            )}

          </Box>

        </Container>

        <Divider />

        <Container maxWidth="md" sx={{ py: { xs: 1, sm: 3 } }}>

          <Grid container justifyContent={"space-between"} alignItems="center">

            <Grid item></Grid>

            <Grid item>

              <Button

                onClick={checkOutHandler}

                disabled={!calTotal()}

                sx={{

                  textTransform: "none",

                  "&:hover": { bgcolor: "#333", color: "#fff" },

                }}

                variant="contained"

                startIcon={<ShoppingCartCheckoutOutlinedIcon />}

              >

                CheckOut

              </Button>

            </Grid>

          </Grid>

        </Container>

      </Box>

    </>

  );

}

export default Cart;

**Checkout.js**

import Box from "@mui/material/Box";

import Container from "@mui/material/Container";

import Paper from "@mui/material/Paper";

import Stepper from "@mui/material/Stepper";

import Step from "@mui/material/Step";

import StepLabel from "@mui/material/StepLabel";

import Typography from "@mui/material/Typography";

import AddressForm from "./AddressForm";

import PaymentForm from "./PaymentForm";

import Header from "../../Components/Header";

import { useState } from "react";

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

import OTP from "./OTP";

import { Button } from "@mui/material";

const steps = ["Shipping address", "Payment details", "OTP verification"];

export default function Checkout(props) {

  const [activeStep, setActiveStep] = useState(0);

  const { ID } = useParams();

  const handleNext = () => {

    setActiveStep(activeStep + 1);

  };

  const handleBack = () => {

    setActiveStep(activeStep - 1);

  };

  function getStepContent(step) {

    switch (step) {

      case 0:

        return <AddressForm id={ID} handleNext={handleNext} />;

      case 1:

        return (

          <PaymentForm

            id={ID}

            handleNext={handleNext}

            handleBack={handleBack}

          />

        );

      case 2:

        return <OTP id={ID} handleNext={handleNext} handleBack={handleBack} />;

      default:

        throw new Error("Unknown step");

    }

  }

  return (

    <>

      <Header mode={props.mode} handler={props.handler} />

      <Box component={Paper} py={0.1} square elevation={1} minHeight="83vh">

        <Container component="main" maxWidth="sm" sx={{ mb: 4 }}>

          <Paper

            variant="outlined"

            sx={{ my: { xs: 3, md: 6 }, p: { xs: 2, md: 3 } }}

          >

            <Typography

              component="h1"

              variant="h3"

              align="center"

              sx={{ color: "#62BB46", fontFamily: "open sans" }}

            >

              Checkout

            </Typography>

            <Stepper

              activeStep={activeStep}

              sx={{ my: 3, py: 3, border: "2px solid #62BB46" }}

            >

              {steps.map((label) => (

                <Step key={label}>

                  <StepLabel>{label}</StepLabel>

                </Step>

              ))}

            </Stepper>

            <>

              {activeStep === steps.length ? (

                <>

                  <Typography variant="h5" gutterBottom>

                    Thank you for your order.

                  </Typography>

                  <Typography variant="subtitle1">

                    Your order number is{" "}

                    <span style={{ color: "green" }}>#{ID}</span>. We have

                    emailed your order confirmation, and will send you an update

                    when your order has shipped.

                  </Typography>

                  <Box sx={{ display: "flex" }}>

                    <Box sx={{ flexGrow: 1 }} />

                    <Button variant="outlined" href="/">

                      Keep Shoping

                    </Button>

                  </Box>

                </>

              ) : (

                <>

                  {getStepContent(activeStep)}

                  <Box

                    sx={{ display: "flex", justifyContent: "flex-end" }}

                  ></Box>

                </>

              )}

            </>

          </Paper>

        </Container>

      </Box>

    </>

  );

}

**AddressForm.js**

import Grid from "@mui/material/Grid";

import Typography from "@mui/material/Typography";

import TextField from "@mui/material/TextField";

import { Button, Box } from "@mui/material";

import { useSelector } from "react-redux";

import { useState, useEffect } from "react";

import axios from "axios";

export default function AddressForm(props) {

  //user data

  const { userID, token } = useSelector((state) => state.loging);

  // form data

  const [fName, setFName] = useState();

  const [LName, setLName] = useState();

  const [address, setAddess] = useState();

  const [city, setCity] = useState();

  const [province, setProvince] = useState();

  const [postalcode, setPostalCode] = useState();

  const [country, setCountry] = useState();

  //data

  const [isFilled, setFilled] = useState(false);

  const [error, setError] = useState("");

  //useEffect call

  useEffect(() => {

    axios

      .get(`http://localhost:5000/users?ID=${userID}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        setFName(res.data.firstName);

        setLName(res.data.lastName);

      })

      .catch((er) => {

        console.log(er);

      });

    //get order data

    axios

      .get(`http://localhost:5000/api/orders?\_id=${props.id}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        setAddess(res.data.address.address);

        setCity(res.data.address.city);

        setProvince(res.data.address.province);

        setPostalCode(res.data.address.postalcode);

        setCountry(res.data.address.country);

        setFilled(true);

      })

      .catch((er) => {

        console.log(er);

      });

  }, []);

  // submit form

  const handleSubmit = (event) => {

    event.preventDefault();

    if (!fName.trim() || !LName.trim()) {

      setError("Invalid  Name");

      return;

    }

    if (!address.trim()) {

      setError("Address required");

      return;

    }

    if (!city.trim()) {

      setError("City Required");

      return;

    }

    if (!province.trim()) {

      setError("provimce Required");

      return;

    }

    if (!postalcode.trim() || isNaN(postalcode)) {

      setError("Invalid postalcode");

      return;

    }

    if (!country.trim()) {

      setError("Country Required");

      return;

    }

    axios

      .put(

        `http://localhost:5000/api/orders`,

        {

          address,

          city,

          province,

          postalcode,

          country,

          \_id: props.id,

        },

        {

          headers: { Authorization: "Agriuservalidation " + token },

        }

      )

      .then((res) => {

        props.handleNext();

      })

      .catch((er) => {});

  };

  return (

    <>

      <Typography variant="h6" gutterBottom sx={{ color: "#62BB46", mb: 2 }}>

        Shipping address

      </Typography>

      <Grid container spacing={3}>

        <Grid

          item

          xs={12}

          sm={6}

          component="form"

          noValidate

          onSubmit={handleSubmit}

        >

          <TextField

            disabled

            required

            inputProps={{ sx: { color: "#62BB46" } }}

            value={fName}

            onChange={(event) => {

              setFName(event.target.value);

            }}

            autoFocus

            id="firstName"

            placeholder="First name"

            fullWidth

            variant="outlined"

          />

        </Grid>

        <Grid item xs={12} sm={6}>

          <TextField

            disabled

            required

            inputProps={{ sx: { color: "#62BB46" } }}

            id="lastName"

            name="lastName"

            placeholder="Last name"

            fullWidth

            autoComplete="family-name"

            variant="outlined"

            value={LName}

            onChange={(event) => {

              setLName(event.target.value);

            }}

          />

        </Grid>

        <Grid item xs={12}>

          <TextField

            required

            disabled={isFilled}

            inputProps={{ sx: { color: "#62BB46" } }}

            id="address"

            name="address"

            placeholder="Address"

            fullWidth

            variant="outlined"

            value={address}

            onChange={(event) => {

              setAddess(event.target.value);

            }}

          />

        </Grid>

        <Grid item xs={12} sm={6}>

          <TextField

            required

            disabled={isFilled}

            inputProps={{ sx: { color: "#62BB46" } }}

            id="city"

            name="city"

            placeholder="City"

            fullWidth

            autoComplete="city"

            variant="outlined"

            value={city}

            onChange={(event) => {

              setCity(event.target.value);

            }}

          />

        </Grid>

        <Grid item xs={12} sm={6}>

          <TextField

            id="state"

            name="state"

            placeholder="Province"

            required

            disabled={isFilled}

            inputProps={{ sx: { color: "#62BB46" } }}

            fullWidth

            variant="outlined"

            value={province}

            onChange={(event) => {

              setProvince(event.target.value);

            }}

          />

        </Grid>

        <Grid item xs={12} sm={6}>

          <TextField

            required

            disabled={isFilled}

            inputProps={{ sx: { color: "#62BB46" } }}

            id="zip"

            name="zip"

            placeholder="Postal code"

            fullWidth

            autoComplete="shipping postal-code"

            variant="outlined"

            value={postalcode}

            onChange={(event) => {

              setPostalCode(event.target.value);

            }}

          />

        </Grid>

        <Grid item xs={12} sm={6}>

          <TextField

            required

            disabled={isFilled}

            inputProps={{ sx: { color: "#62BB46" } }}

            id="country"

            name="country"

            placeholder="Country"

            fullWidth

            autoComplete="shipping country"

            variant="outlined"

            value={country}

            onChange={(event) => {

              setCountry(event.target.value);

            }}

          />

        </Grid>

        <Grid item xs={12}>

          <Box sx={{ display: "flex", justifyContent: "flex-end" }}>

            <Button

              type="submit"

              variant="contained"

              onClick={isFilled ? props.handleNext : handleSubmit}

              sx={{

                mt: 3,

                ml: 1,

                "&:hover": { bgcolor: "#333", color: "#fff" },

              }}

            >

              Next

            </Button>

          </Box>

        </Grid>

      </Grid>

    </>

  );

}

**PaymentForm.js**

import Typography from "@mui/material/Typography";

import Grid from "@mui/material/Grid";

import TextField from "@mui/material/TextField";

import { useState } from "react";

import {

  Button,

  Box,

  FormControlLabel,

  Radio,

  RadioGroup,

} from "@mui/material";

import { useEffect } from "react";

import axios from "axios";

import { useSelector } from "react-redux";

import validator from "validator";

import Alert from "../../Components/Alert";

export default function PaymentForm(props) {

  //data

  const [method, setMethod] = useState("card");

  const [nameOn, setNameOn] = useState();

  const [cardNum, setCardNum] = useState();

  const [ExMonth, setExMonth] = useState();

  const [ExYear, setExYear] = useState();

  const [mobile, setMobile] = useState();

  const [cvv, setCvv] = useState();

  const [amount, setAmount] = useState(0);

  const [isFilled, setFilled] = useState(false);

  //error indicator

  const [error, setError] = useState("");

  // user data

  const { token, userID } = useSelector((state) => state.loging);

  //submit payment

  const handleSubmit = () => {

    //input validation

    if (method === "card") {

      if (!validator.isCreditCard(cardNum.trim())) {

        setError("Invalid card number");

        return;

      }

      if (!ExMonth.trim() || isNaN(ExMonth) || +ExMonth > 12 || +ExMonth < 1) {

        setError("Invalid Expiry Month");

        return;

      }

      if (!ExYear.trim() || isNaN(ExYear) || ExYear < 2022 || ExYear > 2050) {

        setError("Invalid Expiry Year");

        return;

      }

      if (!cvv.trim() || isNaN(cvv) || cvv.length !== 3) {

        setError("Invalid CVV number");

        return;

      }

      if (!nameOn.trim()) {

        setError("Invalid Name on card");

        return;

      }

    }

    if (method === "mobile") {

      if (isNaN(mobile) || mobile.length > 10 || mobile.length < 9) {

        setError("Invalid Mobile number");

      }

    }

    axios

      .post(

        `http://localhost:5000/api/payments`,

        {

          user\_id: userID,

          order\_id: props.id,

          amount: amount,

          method: method,

          mobile\_number: mobile,

          card\_number: cardNum,

          expiry\_year: ExYear,

          expiry\_month: ExMonth,

          cvv,

          name\_on\_card: nameOn,

        },

        {

          headers: { Authorization: "Agriuservalidation " + token },

        }

      )

      .then((res) => {

        props.handleNext();

      })

      .catch((er) => {});

  };

  //useEffect call

  useEffect(() => {

    axios

      .get(`http://localhost:5000/api/payments?order\_id=${props.id}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        if (res.data.exist) {

          setFilled("true");

        }

      })

      .catch((er) => {});

    axios

      .get(`http://localhost:5000/api/orders?\_id=${props.id}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        setAmount(res.data.total);

      })

      .catch((er) => {});

  }, []);

  //payment method handler

  const methodHandler = (event) => {

    setMethod(event.target.value);

  };

  //handleClose

  const handleClose = () => {

    setError("");

  };

  return (

    <>

      {error && (

        <Alert

          open={true}

          msg={error}

          title={"Alert!"}

          handleClose={handleClose}

        />

      )}

      <Typography variant="h6" gutterBottom sx={{ mb: 2, color: "#62BB46" }}>

        Payment method

      </Typography>

      <RadioGroup

        disabled={isFilled}

        value={method}

        onChange={methodHandler}

        row

        sx={{ mb: 2 }}

      >

        <FormControlLabel

          value="card"

          control={<Radio />}

          label="Card Payment"

        />

        <FormControlLabel

          value="mobile"

          control={<Radio />}

          label="Add to Mobile"

        />

      </RadioGroup>

      {method === "card" ? (

        <Grid container spacing={3}>

          <Grid item xs={12} md={12}>

            <TextField

              required

              value={nameOn}

              onChange={(event) => {

                setNameOn(event.target.value);

              }}

              id="cardName"

              label="Name on card"

              fullWidth

              autoComplete="cc-name"

              variant="outlined"

              disabled={isFilled}

            />

          </Grid>

          <Grid item xs={12} md={12}>

            <TextField

              required

              value={cardNum}

              onChange={(event) => {

                setCardNum(event.target.value);

              }}

              id="cardNumber"

              label="Card number"

              fullWidth

              autoComplete="cc-number"

              variant="outlined"

              type="number"

              disabled={isFilled}

            />

          </Grid>

          <Grid item xs={6} sm={4}>

            <TextField

              required

              value={ExYear}

              onChange={(event) => {

                setExYear(event.target.value);

              }}

              id="expDate"

              label="Expiry year"

              fullWidth

              autoComplete="cc-exp"

              variant="outlined"

              disabled={isFilled}

            />

          </Grid>

          <Grid item xs={6} sm={4}>

            <TextField

              value={ExMonth}

              onChange={(event) => {

                setExMonth(event.target.value);

              }}

              required

              id="expDate"

              label="Expiry month"

              fullWidth

              autoComplete="cc-exp"

              variant="outlined"

              disabled={isFilled}

            />

          </Grid>

          <Grid item xs={12} sm={4}>

            <TextField

              value={cvv}

              onChange={(event) => {

                setCvv(event.target.value);

              }}

              required

              id="cvv"

              label="CVV"

              fullWidth

              autoComplete="cc-csc"

              variant="outlined"

              disabled={isFilled}

            />

          </Grid>

        </Grid>

      ) : (

        <Grid container spacing={3}>

          <Grid item xs={12} md={12}>

            <TextField

              required

              value={mobile}

              onChange={(event) => {

                setMobile(event.target.value);

              }}

              disabled={isFilled}

              id="mobileNo"

              label="mobile Number"

              fullWidth

              type="tel"

              autoComplete="mobile"

              variant="outlined"

            />

          </Grid>

        </Grid>

      )}

      <Box sx={{ display: "flex", justifyContent: "flex-end" }}>

        <Button

          variant="contained"

          onClick={isFilled ? props.handleNext : handleSubmit}

          sx={{

            mt: 3,

            ml: 1,

            "&:hover": { bgcolor: "#333", color: "#fff" },

          }}

        >

          Next

        </Button>

      </Box>

    </>

  );

}

**OTP.js**

import Grid from "@mui/material/Grid";

import Typography from "@mui/material/Typography";

import TextField from "@mui/material/TextField";

import { Button, Box } from "@mui/material";

import { useSelector } from "react-redux";

import { useState } from "react";

import axios from "axios";

import Alert from "../../Components/Alert";

export default function OTP(props) {

  //user data

  const { userID, token } = useSelector((state) => state.loging);

  //form data

  const [OTP, setOTP] = useState();

  //eror indicator

  const [error, setError] = useState("");

  // close alert

  const handleClose = () => {

    setError("");

  };

  //submit otp

  const handleSubmit = (event) => {

    event.preventDefault();

    if (!OTP.trim() || OTP.length !== 6 || isNaN(OTP)) {

      setError("Invalid OTP");

      return;

    }

    axios

      .put(

        `http://localhost:5000/api/payments`,

        { OTP, order\_id: props.id, userID },

        {

          headers: { Authorization: "Agriuservalidation " + token },

        }

      )

      .then((res) => {

        if (res.data.ok) {

          props.handleNext();

        }

      })

      .catch((er) => {});

  };

  return (

    <>

      <Alert

        handleClose={handleClose}

        title="Alert!"

        msg={error}

        open={error}

      />

      <Typography variant="h6" gutterBottom sx={{ color: "#62BB46", mb: 2 }}>

        OTP verifications

      </Typography>

      <Grid container spacing={3} justifyContent="center">

        <Grid

          item

          xs={12}

          sm={6}

          component="form"

          noValidate

          onSubmit={handleSubmit}

        >

          <TextField

            required

            inputProps={{ sx: { color: "#62BB46" } }}

            autoFocus

            value={OTP}

            onChange={(event) => {

              setOTP(event.target.value);

            }}

            id="otp"

            placeholder="OTP"

            fullWidth

            type="password"

            variant="outlined"

            sx={{ textAlign: "center", letterSpacing: 3 }}

          />

        </Grid>

        <Grid item xs={12}>

          <Box sx={{ display: "flex", justifyContent: "flex-end" }}>

            <Button

              type="submit"

              variant="contained"

              onClick={handleSubmit}

              sx={{

                mt: 3,

                ml: 1,

                "&:hover": { bgcolor: "#333", color: "#fff" },

              }}

            >

              Next

            </Button>

          </Box>

        </Grid>

      </Grid>

    </>

  );

}

**Dashboard.js**

import {

  Box,

  Container,

  Divider,

  Grid,

  IconButton,

  Pagination,

  Paper,

  TextField,

  Typography,

} from "@mui/material";

import Header from "../../Components/Header";

import AgriCard from "../Utils/AgriCard";

import LocationSearchingOutlinedIcon from "@mui/icons-material/LocationSearchingOutlined";

import axios from "axios";

import { useEffect, useState } from "react";

import { useSelector } from "react-redux";

import AgriSkelton from "../Utils/AgriSkelton";

import Ack from "../../Components/Ack";

function Dashboard(props) {

  //product data

  const [products, setProducts] = useState([]);

  const [count, setCount] = useState(1);

  const [page, setPage] = useState(1);

  const [favorites, setFavorites] = useState([]);

  const [isLoaded, setLoaded] = useState(false);

  const [search, setSearch] = useState();

  const [isEmpty, setEmpty] = useState(false);

  //popup

  const [open, setOpen] = useState(false);

  //user data

  const { token, userID } = useSelector((state) => state.loging);

  //product delete

  const [id, setID] = useState();

  const [index, setIndex] = useState();

  //search product

  const searchHandler = () => {

    setPage(1);

    setLoaded(false);

    setEmpty(false);

    axios

      .get(

        `http://localhost:5000/api/products?pagination=${page}&title=${search}`,

        {

          headers: { Authorization: "Agriuservalidation " + token },

        }

      )

      .then((res) => {

        if (res.data.data.length !== 0) {

          setProducts(res.data.data);

          setLoaded(true);

        } else {

          setEmpty(true);

          setLoaded(true);

        }

      })

      .catch((er) => {

        setLoaded(true);

        setEmpty(true);

      });

  };

  //find favorite products

  const findfav = (array, id) => {

    let val = false;

    array.forEach((data) => {

      if (data === id) {

        val = true;

      }

    });

    return val;

  };

  //page change[pagination]

  const handleChange = (event, value) => {

    setPage(value);

  };

  //useEffect call

  useEffect(() => {

    axios

      .get(`http://localhost:5000/api/products?pagination=${page}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        if (res.data) {

          setProducts(res.data.data);

          setLoaded(true);

          const p\_count = Math.ceil(+res.data.cdata / 6);

          setCount(p\_count);

        }

      })

      .catch((er) => {

        setLoaded(true);

        setEmpty(true);

      });

    axios

      .get(`http://localhost:5000/users/favorites?\_id=${userID}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        if (res.data) {

          setFavorites(res.data);

        }

      })

      .catch(() => {});

  }, [page]);

  //popup handler

  const handleClose = () => {

    setOpen(false);

  };

  //delete product

  const handleYes = () => {

    setOpen(false);

    console.log("ddd");

    axios

      .delete(`http://localhost:5000/api/products?\_id=${id}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        setProducts((pre) => {

          const array = [...pre];

          array.splice(index, 1);

          return array;

        });

      })

      .catch((er) => {});

  };

  //handle click delete btn

  const clickDelete = (id, index) => {

    setOpen(true);

    setID(id);

    setIndex(index);

  };

  return (

    <>

      <Header mode={props.mode} handler={props.handler} />

      <Ack

        title={"Alert"}

        open={open}

        handleClose={handleClose}

        msg="Are you sure to delete"

        handleYes={handleYes}

      />

      <Box py={2} component={Paper} elevation={0} square minHeight={"83vh"}>

        <Container maxWidth="md">

          <Box component="form" noValidate>

            <Grid

              container

              justifyContent="end"

              alignItems={"center"}

              spacing={1}

            >

              <Grid

                item

                xs={false}

                display={{ xs: "none", sm: "block" }}

                sm={3}

                component={Typography}

                variant="h3"

                fontFamily={"roboto"}

                letterSpacing={1}

                sx={{ color: "#62BB46" }}

              >

                AgriGo

              </Grid>

              <Grid item xs={10} sm={7}>

                <TextField

                  onKeyUp={searchHandler}

                  value={search}

                  onChange={(event) => {

                    setSearch(event.target.value);

                  }}

                  margin="none"

                  required

                  fullWidth

                  id="search"

                  label="Search Product"

                  name="search"

                  autoComplete="search"

                  autoFocus

                />

              </Grid>

              <Grid item xs={2} sm={2}>

                <IconButton

                  onClick={searchHandler}

                  sx={{

                    bgcolor: "#62BB46",

                    borderRadius: 1,

                    p: 2,

                    "&:hover": {

                      backgroundColor: "#fff",

                      color: "#62BB46",

                    },

                  }}

                  size="large"

                >

                  <LocationSearchingOutlinedIcon size="large" />

                </IconButton>

              </Grid>

            </Grid>

          </Box>

        </Container>

        <Divider sx={{ my: 2 }} />

        <Container maxWidth="lg">

          <Grid

            container

            direction={"row"}

            justifyContent="center"

            alignItems={"stretch"}

            spacing={4}

            minHeight="50vh"

          >

            {isEmpty && (

              <>

                <Box mt={7} component={Typography} sx={{ textAlign: "center" }}>

                  No products to show

                </Box>

              </>

            )}

            {!isEmpty &&

              isLoaded &&

              products.map((row, index) => {

                const val = findfav(favorites, row.\_id);

                return (

                  <AgriCard

                    clickDelete={clickDelete}

                    key={row.\_id}

                    fav={val}

                    data={row}

                    index={index}

                  />

                );

              })}

            {!isLoaded && (

              <>

                <AgriSkelton />

                <AgriSkelton />

                <AgriSkelton />

              </>

            )}

          </Grid>

          <Box mt={3} />

          <Box sx={{ display: "flex", alignItems: "center" }}>

            <Box sx={{ flexGrow: 1 }} />

            <Pagination count={count} color="primary" onChange={handleChange} />

            <Box sx={{ flexGrow: 1 }} />

          </Box>

        </Container>

      </Box>

    </>

  );

}

export default Dashboard;

**favorites.js**

import {

  Box,

  Container,

  Divider,

  Grid,

  Pagination,

  Paper,

  Typography,

} from "@mui/material";

import Header from "../../Components/Header";

import AgriCard from "../Utils/AgriCard";

import { useEffect, useState } from "react";

import { useSelector } from "react-redux";

import axios from "axios";

import AgriSkelton from "../Utils/AgriSkelton";

function Favorites(props) {

  //state data

  const [products, setProducts] = useState([]);

  const [isLoaded, setLoaded] = useState(false);

  const [page, setPage] = useState(1);

  const [count, setCount] = useState(1);

  //user data

  const { token, userID } = useSelector((state) => state.loging);

  //pagination

  const handleChange = (event, value) => {

    setPage(value);

  };

  //remove favorite handler

  const removeFav = (index) => {

    setProducts((pre) => {

      const array = [...pre];

      array.splice(index, 1);

      return array;

    });

  };

  useEffect(() => {

    //get fav id

    axios

      .get(`http://localhost:5000/users/favorites?\_id=${userID}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        if (res.data) {

          //get products for fav id

          axios

            .get(

              `http://localhost:5000/api/products?pagination=${page}&favList=${

                res.data.length != 0 ? res.data : ["1", "2"]

              }`,

              {

                headers: { Authorization: "Agriuservalidation " + token },

              }

            )

            .then((res) => {

              if (res.data) {

                setProducts(res.data.data);

                const pcount = Math.ceil(+res.data.cdata / 6);

                setCount(pcount);

                setLoaded(true);

              }

            })

            .catch((er) => {

              setLoaded(true);

            });

        }

      })

      .catch(() => {});

  }, [page]);

  return (

    <>

      <Header mode={props.mode} handler={props.handler} />

      <Box py={1} component={Paper} elevation={0} square minHeight={"83vh"}>

        <Container maxWidth="lg">

          <Grid

            textAlign={"left"}

            item

            xs={12}

            component={Typography}

            variant="h3"

            fontFamily={"roboto"}

            letterSpacing={1}

            sx={{ color: "#62BB46", my: 2 }}

          >

            Favorites

          </Grid>

        </Container>

        <Divider sx={{ mb: 1 }} />

        <Container maxWidth="lg">

          <Box minHeight={"55vh"}>

            <Grid

              container

              direction={"row"}

              justifyContent="center"

              alignItems={"center"}

              spacing={4}

            >

              {isLoaded ? (

                products.map((row, index) => {

                  return (

                    <AgriCard

                      removeFav={removeFav}

                      index={index}

                      key={row.\_id}

                      data={row}

                      fav={true}

                    />

                  );

                })

              ) : (

                <>

                  <AgriSkelton />

                  <AgriSkelton />

                  <AgriSkelton />

                </>

              )}

            </Grid>

          </Box>

          <Box my={4} sx={{ display: "flex", alignItems: "center" }}>

            <Box sx={{ flexGrow: 1 }} />

            <Pagination count={count} color="primary" onChange={handleChange} />

            <Box sx={{ flexGrow: 1 }} />

          </Box>

          <Box mt={3} />

        </Container>

      </Box>

    </>

  );

}

export default Favorites;

**ForgetPassword.js**

import {

  Avatar,

  Box,

  Button,

  Paper,

  TextField,

  Typography,

} from "@mui/material";

import Header from "../../Components/Header";

import AgricultureIcon from "@mui/icons-material/Agriculture";

import { useState } from "react";

import axios from "axios";

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

function ForgotPassword(props) {

  //data

  const [email, setEmail] = useState();

  const [isSent, setSent] = useState(false);

  const [OTP, setOTP] = useState("");

  //hook

  const navigate = useNavigate();

  //submit

  const submitHandler = (event) => {

    event.preventDefault();

    if (!email.trim() || !email.includes("@")) {

      return;

    }

    axios

      .post(`http://localhost:5000/users/password`, { email })

      .then((res) => {

        setSent(true);

      })

      .catch((er) => {});

  };

  //otp submit

  const otpHandler = (event) => {

    event.preventDefault();

    axios

      .post(`http://localhost:5000/users/password`, { email, OTP })

      .then((res) => {

        setTimeout(() => {

          navigate("/forget-password/" + res.data.\_id, { replace: true });

        }, 1000);

      })

      .catch((er) => {});

  };

  return (

    <>

      <Header mode={props.mode} handler={props.handler} />

      <Box

        component={Paper}

        variant="elevation"

        square

        elevation={1}

        sx={{ minHeight: "81vh", display: "flex", flexDirection: "column" }}

      >

        <Box

          sx={{

            my: 8,

            mx: 4,

            display: "flex",

            flexDirection: "column",

            alignItems: "center",

          }}

        >

          <Avatar sx={{ m: 1, bgcolor: "secondary.main" }}>

            <AgricultureIcon fontSize="medium" />

          </Avatar>

          <Typography component="h1" variant="h5">

            Reset Pasword

          </Typography>

          <Box

            component="form"

            noValidate

            onSubmit={isSent ? otpHandler : submitHandler}

            sx={{ mt: 1 }}

          >

            {!isSent && (

              <TextField

                margin="normal"

                required

                fullWidth

                id="email"

                label="Email ID"

                name="email"

                autoComplete="email"

                autoFocus

                valuse={email}

                onChange={(event) => {

                  setEmail(event.target.value);

                }}

              />

            )}

            {isSent && (

              <TextField

                margin="normal"

                required

                fullWidth

                name="otp"

                label="OTP"

                type="password"

                id="otp"

                value={OTP}

                onChange={(event) => {

                  setOTP(event.target.value);

                }}

              />

            )}

            <Button

              type="submit"

              fullWidth

              variant="contained"

              sx={{ mt: 3, mb: 2 }}

            >

              Submit

            </Button>

          </Box>

        </Box>

      </Box>

    </>

  );

}

export default ForgotPassword;

**ResetPassword.js**

import {

  Avatar,

  Box,

  Button,

  Paper,

  TextField,

  Typography,

} from "@mui/material";

import Header from "../../Components/Header";

import AgricultureIcon from "@mui/icons-material/Agriculture";

import { useState, useEffect } from "react";

import axios from "axios";

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

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

import Alert from "../../Components/Alert";

function ResetPassword(props) {

  //data

  const [newPassword, setNewPassword] = useState();

  const [reTypePassword, setReTypePassword] = useState();

  const [error, setError] = useState("");

  //hooks

  const { id } = useParams();

  const navigate = useNavigate();

  //useeffect call

  useEffect(() => {

    axios

      .get(`http://localhost:5000/users/password/${id}`)

      .then((res) => {

        if (!res.data.found) {

          navigate("login", { replace: true });

        }

      })

      .catch((er) => {

        navigate("login", { replace: true });

      });

  }, []);

  const submitHandler = (event) => {

    event.preventDefault();

    if (!newPassword.trim() || !reTypePassword.trim()) {

      setError("Passwords required");

      return;

    }

    if (newPassword !== reTypePassword) {

      setError("Passwords did not match");

      return;

    }

    axios

      .post(`http://localhost:5000/users/password/${id}`, {

        password: newPassword,

      })

      .then((res) => {

        navigate("/login", { replace: true });

      })

      .catch((er) => {});

  };

  return (

    <>

      <Alert

        open={error}

        msg={error}

        title={"Alert"}

        handleClose={() => {

          setError(false);

        }}

      />

      <Header mode={props.mode} handler={props.handler} />

      <Box

        component={Paper}

        variant="elevation"

        square

        elevation={1}

        sx={{ minHeight: "81vh", display: "flex", flexDirection: "column" }}

      >

        <Box

          sx={{

            my: 8,

            mx: 4,

            display: "flex",

            flexDirection: "column",

            alignItems: "center",

          }}

        >

          <Avatar sx={{ m: 1, bgcolor: "secondary.main" }}>

            <AgricultureIcon fontSize="medium" />

          </Avatar>

          <Typography component="h1" variant="h5">

            Reset Pasword

          </Typography>

          <Box

            component="form"

            noValidate

            onSubmit={submitHandler}

            sx={{ mt: 1 }}

          >

            <TextField

              margin="normal"

              required

              fullWidth

              id="newpassword"

              label="New Password"

              name="newpassword"

              autoFocus

              type="password"

              valuse={newPassword}

              onChange={(event) => {

                setNewPassword(event.target.value);

              }}

            />

            <TextField

              margin="normal"

              required

              fullWidth

              name="passRE"

              label="Re-type Password"

              type="password"

              value={reTypePassword}

              onChange={(event) => {

                setReTypePassword(event.target.value);

              }}

            />

            <Button

              type="submit"

              fullWidth

              variant="contained"

              sx={{ mt: 3, mb: 2 }}

            >

              Submit

            </Button>

          </Box>

        </Box>

      </Box>

    </>

  );

}

export default ResetPassword;

**NewProduct.js**

import {

  Avatar,

  Box,

  Button,

  Container,

  Grid,

  Paper,

  TextField,

  Tooltip,

  Typography,

  Zoom,

} from "@mui/material";

import Header from "../../Components/Header";

import { useState } from "react";

import CloudUploadIcon from "@mui/icons-material/CloudUpload";

import { useSelector } from "react-redux";

import axios from "axios";

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

import { useEffect } from "react";

import Alert from "../../Components/Alert";

function NewProduct(props) {

  //data

  const [previewUrl, setPreviewUrl] = useState("");

  const [error, setError] = useState("");

  //useEffect call on edit

  useEffect(() => {

    if (id) {

      axios

        .get(`http://localhost:5000/api/products/${id}`, {

          headers: { Authorization: "Agriuservalidation " + token },

        })

        .then((res) => {

          if (res.data.user\_id !== userID) {

            navigate("/", { replace: true });

            return;

          }

          if (res.data) {

            setPreviewUrl(res.data.images);

            setID(res.data.id);

            setTitle(res.data.title);

            setPrice(res.data.price);

            setDescription(res.data.description);

            setCategory(res.data.category);

          }

        })

        .catch((er) => {

          navigate("/", { replace: true });

        });

    }

  }, []);

  //hooks

  const { id } = useParams();

  const navigate = useNavigate();

  //form data

  const [image, setImage] = useState(null);

  const [ID, setID] = useState();

  const [title, setTitle] = useState();

  const [price, setPrice] = useState();

  const [description, setDescription] = useState();

  const [category, setCategory] = useState();

  //user data

  const { token, userID } = useSelector((state) => state.loging);

  //file handle

  const handleFile = (file) => {

    setImage(file);

    setPreviewUrl(URL.createObjectURL(file));

  };

  //image drag and drop handler

  const handleDragOver = (event) => {

    event.preventDefault();

  };

  //image drag and drop handler

  const handleOnDrop = (event) => {

    event.preventDefault();

    event.stopPropagation();

    let imageFile = event.dataTransfer.files[0];

    handleFile(imageFile);

  };

  //image drag and drop handler

  const handleOnChange = (event) => {

    let imageFile = event.target.files[0];

    handleFile(imageFile);

  };

  //submit handler

  const handleSubmit = (event) => {

    event.preventDefault();

    if (

      !ID.trim() ||

      !title.trim() ||

      !price.trim() ||

      !description.trim() ||

      !category.trim()

    ) {

      setError("All Fields are required");

      return;

    }

    if (isNaN(price)) {

      setError("Invalid Price");

      return;

    }

    const data = new FormData();

    data.append("title", title);

    data.append("price", price);

    data.append("description", description);

    data.append("category", category);

    data.append("id", ID);

    data.append("user\_id", userID);

    data.append("image", image);

    data.append("\_id", id ? id : "");

    if (id) {

      axios

        .put(`http://localhost:5000/api/products`, data, {

          headers: { Authorization: "Agriuservalidation " + token },

        })

        .then((res) => {

          navigate("/");

        })

        .catch((er) => {

          console.log(er);

        });

    } else {

      axios

        .post(`http://localhost:5000/api/products`, data, {

          headers: { Authorization: "Agriuservalidation " + token },

        })

        .then((res) => {

          navigate("/");

        })

        .catch((er) => {

          console.log(er);

        });

    }

  };

  //close Alert

  const handleClose = () => {

    setError("");

  };

  return (

    <>

      <Alert

        open={error}

        handleClose={handleClose}

        title="Alert!"

        msg={error}

      />

      <Header mode={props.mode} handler={props.handler} />

      <Box component={Paper} elevation={0} square minHeight={"82vh"} py={3}>

        <Container maxWidth="md">

          <Typography

            variant="h3"

            sx={{ mb: 3, color: "#62BB46", fontFamily: "open sans" }}

          >

            {id ? "Update Product" : "New Product"}

          </Typography>

          <Grid

            component={Paper}

            elevation={2}

            container

            justifyContent={"center"}

            alignItems="center"

            sx={{ border: "2px solid #62BB46", p: 2, borderRadius: 2 }}

          >

            <Grid item xs={12} md={5}>

              <Box p={{ md: 0, xs: 3 }}>

                <Box

                  component={Tooltip}

                  title="change image"

                  sx={{}}

                  TransitionComponent={Zoom}

                >

                  <label

                    htmlFor="image-dp"

                    onDragOver={handleDragOver}

                    onDrop={handleOnDrop}

                    onChange={handleOnDrop}

                  >

                    <Avatar

                      src={previewUrl && previewUrl}

                      variant="square"

                      sx={{

                        borderRadius: "4px",

                        color: "#333",

                        width: { ...{ xs: "70%", md: "100%" } },

                        maxHeight: 400,

                        height: "auto",

                        cursor: "pointer",

                        pl: { xs: "15%", md: "0" },

                      }}

                    >

                      {!previewUrl && (

                        <CloudUploadIcon

                          sx={{

                            color: "#333",

                            width: 300,

                            height: 300,

                          }}

                        />

                      )}

                    </Avatar>

                  </label>

                </Box>

                <input

                  hidden

                  id="image-dp"

                  type={"file"}

                  onChange={handleOnChange}

                />

              </Box>

            </Grid>

            <Grid item xs={12} md={7}>

              <Box

                component="form"

                noValidate

                onSubmit={handleSubmit}

                sx={{ p: 3 }}

              >

                <Grid container spacing={2}>

                  <Grid item xs={12} sm={12}>

                    <TextField

                      autoComplete="given-id"

                      name="id"

                      required

                      fullWidth

                      inputProps={{ sx: { color: "#62BB46" } }}

                      value={ID}

                      onChange={(event) => {

                        setID(event.target.value);

                      }}

                      id="id"

                      placeholder="Product ID"

                      autoFocus

                    />

                  </Grid>

                  <Grid item xs={12} sm={12}>

                    <TextField

                      autoComplete="given-name"

                      name="Profuct name"

                      required

                      fullWidth

                      inputProps={{ sx: { color: "#62BB46" } }}

                      value={title}

                      onChange={(event) => {

                        setTitle(event.target.value);

                      }}

                      id="name"

                      placeholder="Product Name"

                    />

                  </Grid>

                  <Grid item xs={12} sm={12}>

                    <TextField

                      required

                      fullWidth

                      inputProps={{ sx: { color: "#62BB46" } }}

                      value={price}

                      onChange={(event) => {

                        setPrice(event.target.value);

                      }}

                      id="price"

                      placeholder="Price"

                      name="price"

                      autoComplete="price"

                      type="number"

                    />

                  </Grid>

                  <Grid item xs={12}>

                    <TextField

                      required

                      fullWidth

                      inputProps={{ sx: { color: "#62BB46" } }}

                      value={category}

                      onChange={(event) => {

                        setCategory(event.target.value);

                      }}

                      id="category"

                      placeholder="Category"

                      name="category"

                      autoComplete="category"

                    />

                  </Grid>

                  <Grid item xs={12}>

                    <TextField

                      height={1000}

                      maxRows={4}

                      required

                      fullWidth

                      inputProps={{ sx: { color: "#62BB46" } }}

                      value={description}

                      onChange={(event) => {

                        setDescription(event.target.value);

                      }}

                      id="description"

                      placeholder="description"

                      name="description"

                      autoComplete="description"

                      multiline

                    />

                  </Grid>

                </Grid>

                <Button

                  type="submit"

                  fullWidth

                  variant="contained"

                  sx={{

                    mt: 3,

                    "&:hover": { bgcolor: "#333", color: "#fff" },

                  }}

                >

                  {id ? "Update" : "Add"}

                </Button>

              </Box>

            </Grid>

          </Grid>

        </Container>

      </Box>

    </>

  );

}

export default NewProduct;

**ProductView.js**

import {

  Avatar,

  Box,

  Container,

  Grid,

  Paper,

  Skeleton,

  Typography,

} from "@mui/material";

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

import Header from "../../Components/Header";

import { useEffect, useState } from "react";

import axios from "axios";

import { useSelector } from "react-redux";

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

function ProductView(props) {

  //product id

  const { id } = useParams();

  //user data

  const { token, userID } = useSelector((state) => state.loging);

  //product data

  const [image, setImage] = useState(null);

  const [ID, setID] = useState();

  const [title, setTitle] = useState();

  const [price, setPrice] = useState();

  const [description, setDescription] = useState();

  const [category, setCategory] = useState();

  //loading data

  const [isLoaded, setloaded] = useState(false);

  const navigate = useNavigate();

  //useeffect call

  useEffect(() => {

    axios

      .get(`http://localhost:5000/api/products/${id}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        if (res.data) {

          setImage(res.data.images);

          setID(res.data.id);

          setTitle(res.data.title);

          setPrice(res.data.price);

          setDescription(res.data.description);

          setCategory(res.data.category);

          setloaded(true);

        }

      })

      .catch((er) => {

        setloaded(true);

        navigate("/", { replace: true });

      });

  }, []);

  return (

    <>

      <Header mode={props.mode} handler={props.handler} />

      <Box pt={3} minHeight={"81vh"} component={Paper} elevation={1} square>

        <Container maxWidth="md" sx={{ pb: { xs: 3 } }}>

          {isLoaded ? (

            <Grid container component={Paper} variant="outlined">

              <Grid item xs={12} sm={4}>

                <Avatar

                  src={image}

                  variant="square"

                  sx={{

                    borderRadius: "4px",

                    color: "#333",

                    width: "100%",

                    height: "100%",

                    objectFit: "cover",

                  }}

                />

              </Grid>

              <Grid item xs={12} sm={8}>

                <Box p={2}>

                  <Typography variant="h3" sx={{ color: "#62BB46" }}>

                    {title}

                  </Typography>

                  <Typography sx={{ color: "#aaa", mb: 2 }}>

                    ${price}

                  </Typography>

                  <Typography sx={{}}>{category}</Typography>

                  <Typography

                    sx={{

                      color: "#62BB46",

                      textAlign: "justify",

                      fontWeight: "400",

                    }}

                  >

                    {description}

                  </Typography>

                </Box>

              </Grid>

            </Grid>

          ) : (

            <>

              <Skeleton

                animation="pulse"

                variant="rectangular"

                sx={{ borderRadius: 1 }}

                width={"100%"}

                height={250}

              />

            </>

          )}

        </Container>

      </Box>

    </>

  );

}

export default ProductView;

**Profile.js**

import { Box, Container, Divider, Paper } from "@mui/material";

import Header from "../../Components/Header";

import Tab from "@mui/material/Tab";

import TabContext from "@mui/lab/TabContext";

import TabList from "@mui/lab/TabList";

import TabPanel from "@mui/lab/TabPanel";

import { useState } from "react";

import Account from "./Components/Account";

import Products from "./Components/Products";

import { useSelector } from "react-redux";

import Orders from "./Components/Orders";

import CustomerOrders from "./Components/CustomerOrders";

import Payments from "./Components/Payments";

function Profile(props) {

  //initial vaue for tab bar

  const [value, setValue] = useState("1");

  //tab bar handler

  const handleChange = (event, newValue) => {

    setValue(newValue);

  };

  //user data

  const { role } = useSelector((state) => state.loging);

  return (

    <>

      <Header mode={props.mode} handler={props.handler} />

      <Box

        component={Paper}

        sx={{ typography: "body1" }}

        elevation={0}

        square

        minHeight={"83vh"}

      >

        <TabContext value={value}>

          <Container maxWidth="md">

            <Box component={Paper} elevation={2}>

              <TabList onChange={handleChange}>

                <Tab label="Account" value="1" />

                <Tab label="Orders" value="4" />

                {role === "client" && <Tab label="Payment" value="3" />}

                {role === "farmer" && <Tab label="Products" value="2" />}

              </TabList>

            </Box>

            <TabPanel value="1" sx={{ m: 0, p: 0, mt: 2 }}>

              <Account />

            </TabPanel>

            <TabPanel value="2" sx={{ m: 0, p: 0, mt: 2 }}>

              <Products />

            </TabPanel>

            <TabPanel value="3" sx={{ m: 0, p: 0, mt: 2 }}>

              <Payments />

            </TabPanel>

            <TabPanel value="4" sx={{ m: 0, p: 0, mt: 2 }}>

              {role === "farmer" ? <Orders /> : <CustomerOrders />}

            </TabPanel>

          </Container>

        </TabContext>

      </Box>

    </>

  );

}

export default Profile;

**Account.js**

import ChangeCircleIcon from "@mui/icons-material/ChangeCircle";

import {

  Avatar,

  Badge,

  Box,

  Button,

  Container,

  Grid,

  IconButton,

  Paper,

  TextField,

  Tooltip,

} from "@mui/material";

import CollectionsOutlinedIcon from "@mui/icons-material/CollectionsOutlined";

import PersonOutlineOutlinedIcon from "@mui/icons-material/PersonOutlineOutlined";

import ImageModal from "../../Utils/ImageModal";

import { useState, useEffect } from "react";

import axios from "axios";

import { useSelector, useDispatch } from "react-redux";

import AgriSnackbar from "../../Utils/AgriSnackbar";

import LoadingButton from "@mui/lab/LoadingButton";

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

import Alert from "../../../Components/Alert";

function Account() {

  const [open, setOpen] = useState(false);

  const [Sopen, setSOpen] = useState(false);

  const [msg, setMsg] = useState("");

  //handlers

  const handleOpen = () => setOpen(true);

  const handleClose = () => setOpen(false);

  const handleSClose = () => setSOpen(false);

  const [btnDisable, setDisable] = useState(false);

  const [loading, setLoading] = useState(false);

  //hook

  const dispatch = useDispatch();

  //form data

  const [firstname, setFName] = useState("");

  const [lastName, setlName] = useState("");

  const [number, setNumber] = useState("");

  const [email, setEmail] = useState("");

  const [address, setAddress] = useState("");

  const [password, setPassword] = useState("");

  const [newPassword, setNewPassword] = useState("");

  const [ReNewPassword, setReNewPassword] = useState("");

  const [role, setRole] = useState("");

  const [dp, setDp] = useState("");

  const [error, seterror] = useState("");

  //user data

  const { token, userID } = useSelector((state) => state.loging);

  //change role

  const sendRequest = (role) => {

    setLoading(true);

    axios

      .put(

        "http://localhost:5000/users/role",

        { \_id: userID, role },

        {

          headers: { Authorization: "Agriuservalidation " + token },

        }

      )

      .then((res) => {

        setLoading(false);

        setRole(role);

        dispatch(login({ role: role, userID: userID, token }));

      })

      .catch((er) => {

        setLoading(false);

        setMsg("Unable to become " + role);

        setSOpen(true);

      });

  };

  //useEffect call

  useEffect(() => {

    axios

      .get(`http://localhost:5000/users?ID=${userID}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        setFName(res.data.firstName);

        setlName(res.data.lastName);

        setEmail(res.data.email);

        setNumber(res.data.mobile\_number);

        setRole(res.data.role);

        setDp(res.data.images);

        setAddress(res.data.address && res.data.address);

      })

      .catch((er) => {

        console.log(er);

      });

  }, []);

  //update details

  const handleSubmit = (event) => {

    event.preventDefault();

    if (

      !number ||

      number.length > 10 ||

      number.length < 9 ||

      isNaN(number)

    ) {

      seterror("Invalid Mobile Number");

      return;

    }

    if (

      !firstname.trim() ||

      !lastName.trim() ||

      !email.trim() ||

      !address.trim()

    ) {

      seterror("All Fields are Required");

      return;

    }

    if (!email.includes("@")) {

      seterror("Invalid Email ID");

      return;

    }

    setDisable(true);

    axios

      .put(

        "http://localhost:5000/users",

        {

          firstName: firstname,

          lastName,

          address,

          mobile\_number: number,

          \_id: userID,

        },

        {

          headers: { Authorization: "Agriuservalidation " + token },

        }

      )

      .then((res) => {

        window.location.reload();

        setDisable(false);

      })

      .catch((er) => {

        setMsg("Unable to Upadate");

        setSOpen(true);

        setDisable(false);

      });

  };

  //change password

  const handleSubmitPassword = (event) => {

    event.preventDefault();

    if (newPassword !== ReNewPassword) {

      setMsg("Passwords did not match");

      setSOpen(true);

      return;

    }

    setDisable(true);

    axios

      .put(

        "http://localhost:5000/users",

        { password, newPassword, \_id: userID },

        {

          headers: { Authorization: "Agriuservalidation " + token },

        }

      )

      .then((res) => {

        setMsg("Password updated");

        setSOpen(true);

        setPassword("");

        setNewPassword("");

        setReNewPassword("");

        setDisable(false);

      })

      .catch(() => {

        setMsg("Unable to Upadate");

        setSOpen(true);

        setDisable(false);

      });

  };

  //close Alert

  const handleCloseAlert = () => {

    seterror("");

  };

  return (

    <>

      <Alert

        open={error}

        handleClose={handleCloseAlert}

        msg={error}

        title="Alert!"

      />

      <AgriSnackbar open={Sopen} handler={handleSClose} msg={msg} />

      <ImageModal

        userID={userID}

        token={token}

        open={open}

        handleClose={handleClose}

      />

      <Box p={2} m={0} component={Paper} elevation={2} square>

        <Container maxWidth="sm">

          <Box

            pb={4}

            sx={{

              display: "flex",

              flexDirection: "column",

              alignItems: "center",

            }}

          >

            <Badge

              overlap="circular"

              anchorOrigin={{ vertical: "bottom", horizontal: "right" }}

              badgeContent={

                <Tooltip title="change image">

                  <IconButton

                    onClick={handleOpen}

                    sx={{

                      bgcolor: "#62BB46",

                      "&:hover": { bgcolor: "#333", color: "#fff" },

                    }}

                  >

                    <CollectionsOutlinedIcon />

                  </IconButton>

                </Tooltip>

              }

            >

              <Avatar

                src={dp}

                sx={{ height: 100, width: 100, border: "1px solid #333" }}

              >

                <PersonOutlineOutlinedIcon />

              </Avatar>

            </Badge>

          </Box>

          <Box

            component="form"

            noValidate

            onSubmit={handleSubmit}

            sx={{ mt: 3 }}

          >

            <Grid container spacing={2}>

              <Grid item xs={12} sm={6}>

                <TextField

                  autoComplete="given-name"

                  name="firstName"

                  required

                  fullWidth

                  value={firstname}

                  onChange={(event) => {

                    setFName(event.target.value);

                  }}

                  id="firstName"

                  label="First Name"

                  autoFocus

                  inputProps={{ sx: { color: "#62BB46" } }}

                />

              </Grid>

              <Grid item xs={12} sm={6}>

                <TextField

                  required

                  fullWidth

                  inputProps={{ sx: { color: "#62BB46" } }}

                  value={lastName}

                  onChange={(event) => {

                    setlName(event.target.value);

                  }}

                  id="lastName"

                  label="Last Name"

                  name="lastName"

                  autoComplete="family-name"

                />

              </Grid>

              <Grid item xs={12}>

                <TextField

                  contentEditable={false}

                  required

                  fullWidth

                  inputProps={{ sx: { color: "#62BB46" } }}

                  value={email}

                  id="email"

                  label="Email Address"

                  name="email"

                  autoComplete="email"

                  type={"email"}

                />

              </Grid>

              <Grid item xs={12}>

                <TextField

                  required

                  fullWidth

                  inputProps={{ sx: { color: "#62BB46" } }}

                  value={number}

                  onChange={(event) => {

                    setNumber(event.target.value);

                  }}

                  id="mobile-number"

                  label="mobile number"

                  name="mobile-number"

                  type="number"

                  autoComplete="mobile-number"

                />

              </Grid>

              <Grid item xs={12}>

                <TextField

                  required

                  fullWidth

                  inputProps={{ sx: { color: "#62BB46" } }}

                  value={address}

                  onChange={(event) => {

                    setAddress(event.target.value);

                  }}

                  name="address"

                  label="address"

                  id="address"

                  autoComplete="address"

                />

              </Grid>

            </Grid>

            <Button

              disabled={btnDisable}

              type="submit"

              fullWidth

              variant="contained"

              sx={{

                mt: 3,

                mb: 2,

                "&:hover": { bgcolor: "#333", color: "#fff" },

              }}

            >

              Save Changes

            </Button>

          </Box>

          <Box

            component="form"

            noValidate

            onSubmit={handleSubmitPassword}

            sx={{ mt: 3 }}

          >

            <Grid container spacing={2}>

              <Grid item xs={12}>

                <TextField

                  required

                  fullWidth

                  value={password}

                  onChange={(event) => {

                    setPassword(event.target.value);

                  }}

                  name="current-password"

                  label="current password"

                  type="password"

                  id="current-password"

                  autoComplete="current-password"

                />

              </Grid>

              <Grid item xs={12}>

                <TextField

                  required

                  fullWidth

                  value={newPassword}

                  onChange={(event) => {

                    setNewPassword(event.target.value);

                  }}

                  name="new-password"

                  label="new password"

                  type="password"

                  id="new-password"

                  autoComplete="new-password"

                />

              </Grid>

              <Grid item xs={12}>

                <TextField

                  required

                  fullWidth

                  value={ReNewPassword}

                  onChange={(event) => {

                    setReNewPassword(event.target.value);

                  }}

                  name="new-password"

                  label="re-type Password"

                  type="password"

                  id="rnew-password"

                  autoComplete="new-password"

                />

              </Grid>

            </Grid>

            <Button

              disabled={btnDisable}

              type="submit"

              fullWidth

              variant="contained"

              sx={{

                mt: 3,

                mb: 2,

                "&:hover": { bgcolor: "#333", color: "#fff" },

              }}

            >

              Change Password

            </Button>

          </Box>

          <Box my={{ xs: 1, md: 4 }} sx={{ textAlign: "right" }}>

            {role === "client" ? (

              <LoadingButton

                onClick={() => {

                  sendRequest("farmer");

                }}

                loading={loading}

                loadingPosition="start"

                startIcon={<ChangeCircleIcon />}

                variant="outlined"

              >

                Become Farmer

              </LoadingButton>

            ) : (

              <>

                <LoadingButton

                  onClick={() => {

                    sendRequest("client");

                  }}

                  loading={loading}

                  loadingPosition="start"

                  startIcon={<ChangeCircleIcon />}

                  variant="outlined"

                >

                  Become Customer

                </LoadingButton>

              </>

            )}

          </Box>

        </Container>

      </Box>

    </>

  );

}

export default Account;

**CustomerOrder.js**

import { Box, Paper, Skeleton, Typography } from "@mui/material";

import axios from "axios";

import { useSelector } from "react-redux";

import { useState, useEffect } from "react";

function CustomerOrders() {

  //user data

  const { token, userID } = useSelector((state) => state.loging);

  //data

  const [orders, setOrders] = useState([]);

  const [isLoaded, setLoaded] = useState(false);

  //useEffect call

  useEffect(() => {

    axios

      .get(`http://localhost:5000/api/orders?userID=${userID}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        if (res.data) {

          setOrders(res.data);

          setLoaded(true);

        }

      })

      .catch((er) => {

        setLoaded(true);

      });

  }, []);

  //single order data

  const SingleOrder = (props) => {

    return (

      <Box

        variant="outlined"

        px={1}

        py={2}

        mt={2}

        component={Paper}

        sx={{ display: "flex" ,justifyContent:"space-between"}}

      >

        <Typography variant="h4">

          <span style={{ color: "#62BB46" }}>Order ID :</span>

          {props.data.\_id}

        </Typography>

        <Box sx={{ flexGrow: 1 }} />

        <Typography variant="h4">

          <span style={{ color: "#62BB46" }}>Total : </span>

          {props.data.total}

        </Typography>

        <Box sx={{ flexGrow: 1 }} />

        <Typography variant="h4">

          <span style={{ color: "#62BB46" }}>Status : </span>

          {props.data.status ? "Completed" : "Pending"}

        </Typography>

      </Box>

    );

  };

  // generate array

  const N = 6;

  const count = Array.from({ length: N }, (\_, index) => index + 1);

  return (

    <>

      <Box component={Paper} p={2} elevation={2} square minHeight={"70vh"}>

        <Box sx={{ display: "flex", alignItems: "center" }}>

          <Typography sx={{ color: "#62BB46" }} variant="h4">

            Your Orders

          </Typography>

        </Box>

        <Box>

          {isLoaded &&

            orders.length > 0 &&

            orders.map((row, index) => {

              return <SingleOrder data={row} key={index} />;

            })}

          {!isLoaded &&

            count.map((row, index) => {

              return (

                <Skeleton

                  key={index}

                  nimation="pulse"

                  variant="rectangular"

                  sx={{ borderRadius: 1, mb: 2 }}

                  width={"100%"}

                  height={50}

                />

              );

            })}

        </Box>

      </Box>

    </>

  );

}

export default CustomerOrders;

**Orders.js**

import { Box, Paper, Skeleton, Typography } from "@mui/material";

import axios from "axios";

import { useEffect, useState } from "react";

import AgriOrdersFarmer from "../../Utils/AgriOrdersFarmer";

import { useSelector } from "react-redux";

function Orders() {

  //user data

  const { token, userID } = useSelector((state) => state.loging);

  //data

  const [orders, setOrders] = useState({});

  const [isLoaded, setLoaded] = useState(false);

  // useEffect call

  useEffect(() => {

    axios

      .get(`http://localhost:5000/api/orders/${userID}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        setOrders(res.data);

        setLoaded(true);

      })

      .catch((er) => {

        setLoaded(true);

      });

  }, []);

  const LoopOrders = () => {

    let Element = [];

    for (let key in orders) {

      if (orders[key][0]) {

        Element.push(key);

      }

    }

    return Element;

  };

  return (

    <>

      <Box component={Paper} p={2} elevation={2} square minHeight={"70vh"}>

        <Box sx={{ display: "flex", alignItems: "center" }}>

          <Typography sx={{ color: "#62BB46" }} variant="h3">

            Your Orders

          </Typography>

        </Box>

        {orders &&

          isLoaded &&

          LoopOrders().map((row) => {

            return <AgriOrdersFarmer id={row} value={orders[row]} key={row} />;

          })}

        {!isLoaded && (

          <>

            <Skeleton

              nimation="pulse"

              variant="rectangular"

              sx={{ borderRadius: 1, mb: 2 }}

              width={"100%"}

              height={200}

            />

            <Skeleton

              nimation="pulse"

              variant="rectangular"

              sx={{ borderRadius: 1, mb: 2 }}

              width={"100%"}

              height={200}

            />

          </>

        )}

        {isLoaded && !orders && (

          <>

            <Typography sx={{ textAlign: "center", my: 2 }}>

              No Orders

            </Typography>

          </>

        )}

      </Box>

    </>

  );

}

export default Orders;

**Payment.js**

import { Box, Button, Paper, Typography } from "@mui/material";

import axios from "axios";

import { useSelector } from "react-redux";

import { useState, useEffect } from "react";

function Payments() {

  //user data

  const { token, userID } = useSelector((state) => state.loging);

  //data

  const [payment, setpayment] = useState({});

  const [isLoaded, setLoaded] = useState(false);

  //useEffect call

  useEffect(() => {

    axios

      .get(`http://localhost:5000/api/payments?userID=${userID}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        setpayment(res.data);

        setLoaded(true);

      })

      .catch((er) => {

        setLoaded(true);

      });

  }, []);

  //remover handler

  const removehandler = () => {

    axios

      .delete(`http://localhost:5000/api/payments?\_id=${payment.\_id}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        setpayment({});

      })

      .catch((er) => {});

  };

  return (

    <>

      <Box component={Paper} p={2} elevation={2} square minHeight={"70vh"}>

        <Box sx={{ display: "flex", alignItems: "center" }}>

          <Typography sx={{ color: "#62BB46" }} variant="h4">

            Your Payment Data

          </Typography>

        </Box>

        {isLoaded && payment.\_id && (

          <Box

            component={Paper}

            variant="outlined"

            sx={{ display: "flex", alignItems: "center" }}

            mt={3}

            p={1.2}

          >

            <Typography variant="h4">{`xxxx xxxx xxxx ${payment.card\_number

              .toString()

              .substring(12, 17)}`}</Typography>

            <Typography variant="h4" sx={{ ml: 3 }}>

              {`${payment.expiry\_year}/${payment.expiry\_month}`}

            </Typography>

            <Box sx={{ flexGrow: 1 }} />

            <Button onClick={removehandler}>Remove</Button>

          </Box>

        )}

        {isLoaded && !payment.\_id && (

          <Typography sx={{textAlign:"center",my:3}}>No Data available</Typography>

        )}

      </Box>

    </>

  );

}

export default Payments;

**Products.js**

import {

  Box,

  Button,

  Container,

  Divider,

  Pagination,

  Paper,

  Typography,

} from "@mui/material";

import AgriCard from "../../Utils/AgriCard";

import { useEffect, useState } from "react";

import { useSelector } from "react-redux";

import Ack from "../../../Components/Ack";

import axios from "axios";

import AgriSkelton from "../../Utils/AgriSkelton";

function Products() {

  //user data

  const { token, userID } = useSelector((state) => state.loging);

  //pagination data

  const [page, setPage] = useState(1);

  const [count, setCount] = useState(1);

  //product data

  const [products, setProducts] = useState([]);

  const [isEmpty, setEmpty] = useState(false);

  //loading hendler

  const [isLoaded, setLoaded] = useState(false);

  //ack popup

  const [open, setOpen] = useState(false);

  //delete handler

  const [id, setID] = useState();

  const [index, setIndex] = useState();

  const handleChange = (event, value) => {

    setPage(value);

  };

  //useEffect hook

  useEffect(() => {

    axios

      .get(

        `http://localhost:5000/api/products?pagination=${page}&owner=${userID}`,

        {

          headers: { Authorization: "Agriuservalidation " + token },

        }

      )

      .then((res) => {

        if (res.data) {

          setProducts(res.data.data);

          setLoaded(true);

          const pcount = Math.ceil(+res.data.cdata / 6);

          setCount(pcount);

        }

      })

      .catch((er) => {

        setLoaded(true);

        setEmpty(true);

      });

  }, [page]);

  const handleClose = () => {

    setOpen(false);

  };

  //delete handler

  const handleYes = () => {

    setOpen(false);

    axios

      .delete(`http://localhost:5000/api/products?\_id=${id}`, {

        headers: { Authorization: "Agriuservalidation " + token },

      })

      .then((res) => {

        setProducts((pre) => {

          const array = [...pre];

          array.splice(index, 1);

          return array;

        });

      })

      .catch((er) => {});

  };

  //delete btn click handler

  const clickDelete = (id, index) => {

    setOpen(true);

    setID(id);

    setIndex(index);

  };

  return (

    <>

      <Ack

        title={"Alert"}

        open={open}

        handleClose={handleClose}

        msg="Are you sure to delete"

        handleYes={handleYes}

      />

      <Box component={Paper} p={2} elevation={2} square minHeight={"70vh"}>

        <Box sx={{ display: "flex", alignItems: "center" }}>

          <Typography variant="h5" sx={{ color: "#62BB46" }}>

            Your Products

          </Typography>

          <Box sx={{ flexGrow: 1 }} />

          <Button

            href="/product/add"

            variant="contained"

            disableElevation

            sx={{ "&:hover": { bgcolor: "#333", color: "#62BB46" } }}

          >

            Add Product

          </Button>

        </Box>

        <Divider sx={{ my: 2 }} />

        <Container maxWidth="xs">

          {isEmpty && (

            <>

              <Box mt={7} component={Typography} sx={{ textAlign: "center" }}>

                No products to show

              </Box>

            </>

          )}

          {!isEmpty &&

            isLoaded &&

            products.map((row, index) => {

              // const val = findfav(favorites, row.\_id);

              return (

                <AgriCard

                  clickDelete={clickDelete}

                  key={row.\_id}

                  // fav={val}

                  data={row}

                  index={index}

                />

              );

            })}

          {!isLoaded && (

            <>

              <AgriSkelton />

              <AgriSkelton />

              <AgriSkelton />

            </>

          )}

          <Box sx={{ display: "flex", alignItems: "center" }} mt={6}>

            <Box sx={{ flexGrow: 1 }} />

            <Pagination count={count} color="primary" onChange={handleChange} />

            <Box sx={{ flexGrow: 1 }} />

          </Box>

        </Container>

      </Box>

    </>

  );

}

export default Products;

**AgriCard.js**

import \* as React from "react";

import Card from "@mui/material/Card";

import CardActions from "@mui/material/CardActions";

import CardContent from "@mui/material/CardContent";

import CardMedia from "@mui/material/CardMedia";

import Button from "@mui/material/Button";

import Typography from "@mui/material/Typography";

import { Grid, IconButton } from "@mui/material";

import FavoriteBorderOutlinedIcon from "@mui/icons-material/FavoriteBorderOutlined";

import FavoriteOutlinedIcon from "@mui/icons-material/FavoriteOutlined";

import ShoppingCartOutlinedIcon from "@mui/icons-material/ShoppingCartOutlined";

import ModeEditOutlineOutlinedIcon from "@mui/icons-material/ModeEditOutlineOutlined";

import DeleteOutlineOutlinedIcon from "@mui/icons-material/DeleteOutlineOutlined";

import { red } from "@mui/material/colors";

import { useSelector } from "react-redux";

import { useState } from "react";

import axios from "axios";

import AgriSnackbar from "../Utils/AgriSnackbar";

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

export default function AgriCard(props) {

  //user data

  const { token, role, userID } = useSelector((state) => state.loging);

  //favorite indicator

  const [fav, setFav] = useState(props.fav);

  //popup indicator

  const [open, setOpen] = useState(false);

  //hooks

  const navigate = useNavigate();

  //add to cart handler

  const addTocart = () => {

    axios

      .put(

        `http://localhost:5000/users/carts`,

        {

          pid: props.data.\_id,

          \_id: userID,

          set: true,

        },

        {

          headers: { Authorization: "Agriuservalidation " + token },

        }

      )

      .then((res) => {

        setOpen(true);

      })

      .catch((er) => {});

  };

  //handle favorite click

  const handlefavorite = (val) => {

    axios

      .put(

        `http://localhost:5000/users/favorites`,

        {

          \_id: userID,

          pid: props.data.\_id,

          val: val,

        },

        {

          headers: { Authorization: "Agriuservalidation " + token },

        }

      )

      .then((res) => {

        setFav((pre) => !pre);

        props.removeFav(props.index);

      })

      .catch(() => {});

  };

  return (

    <Grid item md={4} sm={6} xs={12} sx={{ mt: { xs: 1, sm: 2 } }}>

      <AgriSnackbar

        msg={"Added to cart"}

        open={open}

        handler={() => {

          setOpen(false);

        }}

      />

      <Card

        sx={{

          minWidth: 270,

          border: "2px solid #62BB46",

          "&:hover": {

            boxShadow: "0 0 5px 2px #62BB46",

            transitionDuration: ".5s",

          },

        }}

      >

        <CardMedia

          component="img"

          height="160"

          image={props.data.images}

          alt="green iguana"

          onClick={() => {

            navigate(`/product/view/${props.data.\_id}`);

          }}

          sx={{ cursor: "pointer" }}

        />

        <CardContent>

          <Grid container justifyContent={"space-between"} alignItems="center">

            <Grid item>

              <Typography

                gutterBottom

                variant="h5"

                component="div"

                textAlign={"left"}

              >

                {props.data.title}

              </Typography>

            </Grid>

          </Grid>

          <Typography

            gutterBottom

            variant="body1"

            textAlign={"left"}

            color="primary"

          >

            {`$${props.data.price}`}

          </Typography>

          <Typography

            variant="body2"

            color="text.secondary"

            textAlign={"justify"}

          >

            {props.data.description.substring(0, 200) + "..."}

          </Typography>

        </CardContent>

        <CardActions>

          {role === "client" ? (

            <>

              <IconButton

                sx={{ color: red[800], mr: 2 }}

                onClick={() => {

                  handlefavorite(!fav);

                }}

              >

                {fav ? (

                  <FavoriteOutlinedIcon />

                ) : (

                  <FavoriteBorderOutlinedIcon />

                )}

              </IconButton>

              <Button

                onClick={addTocart}

                component={Typography}

                variant="contained"

                size="small"

                startIcon={<ShoppingCartOutlinedIcon />}

                sx={{

                  textTransform: "none",

                  "&:hover": {

                    backgroundColor: "#333",

                    color: "#62BB46",

                  },

                }}

              >

                ADD to CART

              </Button>

            </>

          ) : (

            <>

              <Button

                disabled={userID !== props.data.user\_id}

                href={`/product/edit/${props.data.\_id}`}

                component={Button}

                variant="contained"

                size="small"

                startIcon={<ModeEditOutlineOutlinedIcon />}

                sx={{

                  textTransform: "none",

                  "&:hover": {

                    backgroundColor: "#444",

                    color: "#62BB46",

                  },

                  mr: 1,

                }}

              >

                Edit

              </Button>

              <Button

                disabled={userID !== props.data.user\_id}

                onClick={() => {

                  props.clickDelete(props.data.\_id, props.index);

                }}

                color="error"

                component={Typography}

                variant="contained"

                size="small"

                startIcon={<DeleteOutlineOutlinedIcon />}

                sx={{

                  textTransform: "none",

                  "&:hover": {

                    backgroundColor: "#444",

                    color: "#62BB46",

                  },

                }}

              >

                Delete

              </Button>

            </>

          )}

        </CardActions>

      </Card>

    </Grid>

  );

}

**AgriCart.js**

import \* as React from "react";

import Box from "@mui/material/Box";

import Card from "@mui/material/Card";

import CardContent from "@mui/material/CardContent";

import CardMedia from "@mui/material/CardMedia";

import IconButton from "@mui/material/IconButton";

import Typography from "@mui/material/Typography";

import CloseOutlinedIcon from "@mui/icons-material/CloseOutlined";

import RemoveOutlinedIcon from "@mui/icons-material/RemoveOutlined";

import AddOutlinedIcon from "@mui/icons-material/AddOutlined";

import { useState } from "react";

export default function AgriCart(props) {

  //data

  const [val, setVal] = useState(1);

  //count increment

  const incVal = () => {

    setVal((pre) => ++pre);

    props.quantityHandler("inc", +props.data.price, props.data.\_id, val + 1);

  };

  //count decrement

  const decVal = () => {

    if (val >= 2) {

      props.quantityHandler("dec", +props.data.price, props.data.\_id, val - 1);

    }

    setVal((pre) => {

      if (pre > 2) {

        return --pre;

      } else {

        return 1;

      }

    });

  };

  return (

    <Card sx={{ display: "flex", my: 3, border: "2px solid #62BB46" }}>

      <CardMedia

        component="img"

        sx={{ width: "30%" }}

        image={props.data.images}

        alt="img"

      />

      <Box sx={{ display: "flex", flexDirection: "column" }}>

        <CardContent sx={{ flex: "2 0 auto", textAlign: "left" }}>

          <Typography

            fontFamily={"open sans"}

            fontWeight="bold"

            component="div"

            variant="h5"

            sx={{ color: "#62BB46" }}

          >

            {`$${props.data.price}`}

          </Typography>

          <Typography

            variant="h6"

            color="text.secondary"

            component="div"

            fontFamily={"open sans"}

            fontWeight="bold"

          >

            {props.data.title}

          </Typography>

          <Typography

            variant="subtitle2"

            color="text.secondary"

            component="div"

            fontFamily={"open sans"}

            fontWeight="bold"

          >

            {props.data.description}

          </Typography>

        </CardContent>

        <Box sx={{ display: "flex", alignItems: "center", pl: 1, pb: 1 }}>

          <IconButton

            onClick={decVal}

            sx={{

              bgcolor: "#aaa",

              borderRadius: 1,

              mr: 3,

              ml: 1,

              "&:hover": { bgcolor: "#62BB46" },

            }}

          >

            <RemoveOutlinedIcon />

          </IconButton>

          <Typography>{val}</Typography>

          <IconButton

            onClick={incVal}

            sx={{

              bgcolor: "#aaa",

              borderRadius: 1,

              ml: 3,

              "&:hover": { bgcolor: "#62BB46" },

            }}

          >

            <AddOutlinedIcon />

          </IconButton>

        </Box>

      </Box>

      <Box sx={{ flexGrow: 1 }} />

      <Box p={0.5}>

        <IconButton

          onClick={() => {

            props.removeCart(props.index, props.data.\_id);

          }}

        >

          <CloseOutlinedIcon />

        </IconButton>

      </Box>

    </Card>

  );

}

**AgriOrdersfarmer.js**

import { Box, Button, Paper, Typography } from "@mui/material";

const SingleOrder = (props) => {

  return (

    <Box sx={{ display: "flex" }} py={1}>

      <Typography>Quantity : {props.data.count}</Typography>

      <Box sx={{ flexGrow: 1 }} />

      <Typography>

        {`${props.data.address.address} / ${props.data.address.city} / ${props.data.address.province}`}

      </Typography>

    </Box>

  );

};

function AgriOrdersFarmer(props) {

  return (

    <>

      <Box component={Paper} variant="outlined" my={2} p={2}>

        <Box

          pb={2}

          mb={2}

          sx={{

            display: "flex",

            alignItems: "center",

            borderBottom: "2px solid #62BB46",

          }}

        >

          <Typography sx={{ color: "#62BB46" }} variant="h4">

            Product ID : {`${props.id}`}

          </Typography>

          <Box sx={{ flexGrow: 1 }} />

          <Button href={`/product/view/${props.id}`} variant="outlined">

            View Product

          </Button>

        </Box>

        <Box>

          {props.value.map((row) => {

            return <SingleOrder data={row} />;

          })}

        </Box>

      </Box>

    </>

  );

}

export default AgriOrdersFarmer;

**AgriSkelton.js**

import { Box, Grid, Skeleton } from "@mui/material";

function AgriSkelton() {

  return (

    <>

      <Grid

        item

        md={4}

        sm={6}

        xs={12}

        sx={{ mt: { xs: 1, sm: 2 }, textAlign: "center" }}

      >

        <Skeleton

          animation="pulse"

          variant="rectangular"

          sx={{ borderRadius: 1 }}

          width={"100%"}

          height={200}

        />

        <Skeleton

          animation="pulse"

          variant="text"

          sx={{ borderRadius: 1 }}

          width={"100%"}

        />

        <Skeleton

          animation="pulse"

          variant="text"

          sx={{ borderRadius: 1 }}

          width={"100%"}

        />

        <Skeleton

          animation="pulse"

          variant="text"

          sx={{ borderRadius: 1 }}

          width={"100%"}

        />

        <Skeleton

          animation="pulse"

          variant="rectangular"

          sx={{ borderRadius: 1 }}

          width={"100%"}

          height={30}

        />

      </Grid>

    </>

  );

}

export default AgriSkelton;

**AgriSnackbar.js**

import { IconButton, Snackbar } from "@mui/material";

import CloseIcon from "@mui/icons-material/Close";

function AgriSnackbar(props) {

  return (

    <>

      <Snackbar

        open={props.open}

        anchorOrigin={{ vertical: "bottom", horizontal: "center" }}

        message={props.msg}

        onClose={props.handler}

        key={"Agri"}

        autoHideDuration={2000}

        action={

          <IconButton sx={{ color: "#333" }} onClick={props.handler}>

            <CloseIcon />

          </IconButton>

        }

      />

    </>

  );

}

export default AgriSnackbar;

**ImageModal.js**

import { Avatar, Box, Button, Grid, Modal } from "@mui/material";

import CloudUploadIcon from "@mui/icons-material/CloudUpload";

import { grey, red } from "@mui/material/colors";

import DeleteIcon from "@mui/icons-material/Delete";

import UploadIcon from "@mui/icons-material/Upload";

import { useEffect, useState } from "react";

import axios from "axios";

//css

const style = {

  textAlign: "center",

  position: "absolute",

  top: "50%",

  left: "50%",

  transform: "translate(-50%, -50%)",

  bgcolor: "#62BB46",

  boxShadow: 24,

  p: 3,

  borderRadius: 2,

};

function ImageModal(props) {

  //data

  const [image, setImage] = useState(null);

  const [previewUrl, setPreviewUrl] = useState("");

  //handler

  const handleFile = (file) => {

    setImage(file);

    setPreviewUrl(URL.createObjectURL(file));

  };

  //drag and drop handlers

  const handleDragOver = (event) => {

    event.preventDefault();

  };

  const handleOnDrop = (event) => {

    event.preventDefault();

    event.stopPropagation();

    let imageFile = event.dataTransfer.files[0];

    handleFile(imageFile);

  };

  const handleOnChange = (event) => {

    let imageFile = event.target.files[0];

    handleFile(imageFile);

  };

  //upload dp

  const uploadImage = () => {

    const data = new FormData();

    data.append("image", image);

    data.append("id", props.userID);

    axios

      .post(`http://localhost:5000/users/dp/${props.userID}`, data, {

        headers: { Authorization: "Agriuservalidation " + props.token },

      })

      .then((res) => {

        window.location.reload();

      })

      .catch((er) => {});

  };

  //delete dp

  const removeImage = () => {

    axios

      .delete(`http://localhost:5000/user/dp/${props.userID}`, {

        headers: { Authorization: "Agriuservalidation " + props.token },

      })

      .then((res) => {

        setPreviewUrl("");

        window.location.reload();

      })

      .catch((er) => {});

  };

  //useEffect hook

  useEffect(() => {

    axios

      .get(`http://localhost:5000/users/dp/${props.userID}`, {

        headers: { Authorization: "Agriuservalidation " + props.token },

      })

      .then((res) => {

        setPreviewUrl(res.data.images);

      })

      .catch((er) => {

        console.log(er);

      });

  }, []);

  return (

    <>

      <Modal open={props.open} onClose={props.handleClose}>

        <Box sx={style}>

          <Box sx={{ textAlign: "center" }} width="100%">

            <label

              style={{ width: "100%", bgcolor: "red" }}

              htmlFor="image-dp"

              onDragOver={handleDragOver}

              onDrop={handleOnDrop}

              onChange={handleOnDrop}

            >

              <Avatar

                src={previewUrl && previewUrl}

                variant="square"

                sx={{

                  borderRadius: "4px",

                  bgcolor: grey[500],

                  color: "#333",

                  width: 330,

                  minHeight: 330,

                  maxHeight: 450,

                  height: "auto",

                  cursor: "pointer",

                }}

              >

                {!previewUrl && (

                  <CloudUploadIcon

                    sx={{

                      color: "#333",

                      width: 300,

                      height: 300,

                    }}

                  />

                )}

              </Avatar>

            </label>

            <br />

            <input

              hidden

              id="image-dp"

              type={"file"}

              onChange={handleOnChange}

            />

            <Grid

              container

              direction="row"

              alignItems={"center"}

              justifyContent="space-between"

            >

              <Grid item>

                <Button

                  onClick={uploadImage}

                  disableElevation

                  sx={{

                    bgcolor: "#333",

                    color: "#fff",

                    fontFamily: "open sans",

                    "&:hover": {

                      backgroundColor: "#fff",

                      color: "#3c52b2",

                    },

                  }}

                  variant="contained"

                  endIcon={<UploadIcon fontSize="small" />}

                >

                  Upload

                </Button>

              </Grid>

              <Grid item>

                <Button

                  onClick={removeImage}

                  disableElevation

                  sx={{

                    color: "#fff",

                    fontFamily: "open sans",

                    "&:hover": {

                      backgroundColor: "#fff",

                      color: "#3c52b2",

                    },

                  }}

                  color="error"

                  endIcon={<DeleteIcon fontSize="small" />}

                  variant="contained"

                >

                  Remove

                </Button>

              </Grid>

              <Grid item>

                <Button

                  sx={{ color: red[900], fontFamily: "open sans" }}

                  onClick={props.handleClose}

                >

                  Cancel

                </Button>

              </Grid>

            </Grid>

          </Box>

        </Box>

      </Modal>

    </>

  );

}

export default ImageModal;

**Pages.js**

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

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

import Signin from "./Auth/Signin/Signin";

import SignUp from "./Auth/Signup/Signup";

import Cart from "./Cart/Cart";

import Checkout from "./Checkout/Checkout";

import Dashboard from "./Dashboard/Dashboard";

import Favorites from "./Favorites/Favorites";

import NewProduct from "./Products/NewProduct";

import Profile from "./Profile/Profile";

import { useSelector } from "react-redux";

import PageNotFound from "./404/PageNotFound";

import ForgotPassword from "./ForgetPassword/ForgetPassword";

import ResetPassword from "./ForgetPassword/ResetPassword";

import ProductView from "./ProductView/ProductView";

function Pages(props) {

  const { token, role } = useSelector((state) => state.loging);

  return (

    <>

      <Routes>

        {token && (

          <>

            {role === "farmer" && (

              <>

                <Route

                  eaxct

                  path="/product/add"

                  element={<NewProduct handler={props.modeHandler} />}

                />

                <Route

                  eaxct

                  path="/product/edit/:id"

                  element={<NewProduct handler={props.modeHandler} />}

                />

              </>

            )}

            <Route

              eaxct

              path="product/view/:id"

              element={<ProductView handler={props.modeHandler} />}

            />

            <Route

              eaxct

              path="/dashboard"

              element={<Dashboard handler={props.modeHandler} />}

            />

            {role === "client" && (

              <>

                <Route

                  eaxct

                  path="/checkout/:ID"

                  element={<Checkout handler={props.modeHandler} />}

                />

                <Route

                  eaxct

                  path="/favorites"

                  element={<Favorites handler={props.modeHandler} />}

                />

                <Route

                  eaxct

                  path="/cart"

                  element={<Cart handler={props.modeHandler} />}

                />

              </>

            )}

            <Route

              eaxct

              path="/profile"

              element={<Profile handler={props.modeHandler} />}

            />

            <Route

              eaxct

              path="/login"

              element={<Signin handler={props.modeHandler} />}

            />

            <Route

              eaxct

              path="/404"

              element={<PageNotFound handler={props.modeHandler} />}

            />

            <Route

              exact

              path="/"

              element={<Navigate replace to="/dashboard" />}

            />

            <Route

              exact

              path="/signup"

              element={<Navigate replace to="/dashboard" />}

            />

            <Route exact path="\*" element={<Navigate replace to="/404" />} />

          </>

        )}

        {!token && (

          <>

            <Route

              eaxct

              path="/forget-password"

              element={<ForgotPassword handler={props.modeHandler} />}

            />

            <Route

              eaxct

              path="/forget-password/:id"

              element={<ResetPassword handler={props.modeHandler} />}

            />

            <Route

              eaxct

              path="/signup"

              element={<SignUp handler={props.modeHandler} />}

            />

            <Route

              eaxct

              path="/login"

              element={<Signin handler={props.modeHandler} />}

            />

            <Route exact path="\*" element={<Navigate replace to="/login" />} />

          </>

        )}

      </Routes>

    </>

  );

}

export default Pages;

**Auth.js**

import { createSlice } from "@reduxjs/toolkit";

const initial = {

  token: localStorage.getItem("token"),

  role: localStorage.getItem("role"),

  userID: localStorage.getItem("userID"),

};

const authStore = createSlice({

  name: "loging",

  initialState: initial,

  reducers: {

    login(state, action) {

      state.role = action.payload.role;

      state.userID = action.payload.userID;

      state.token = action.payload.token;

      localStorage.setItem("token", state.token);

      localStorage.setItem("role", state.role);

      localStorage.setItem("userID", state.userID);

    },

    logout(state) {

      localStorage.removeItem("token");

      localStorage.removeItem("role");

      localStorage.removeItem("userID");

    },

  },

});

export default authStore;

export const { login, logout } = authStore.actions;

**Theme.js**

import { createSlice } from "@reduxjs/toolkit";

if (!localStorage.getItem("mode")) {

  localStorage.setItem("mode", "light");

}

const initial = {

  mode: localStorage.getItem("mode"),

};

const themeStore = createSlice({

  name: "mode",

  initialState: initial,

  reducers: {

    dark(state, action) {

      state.mode = "dark";

      localStorage.setItem("mode", "dark");

    },

    light(state) {

      state.mode = "light";

      localStorage.setItem("mode", "light");

    },

  },

});

export default themeStore;

export const { dark, light } = themeStore.actions;

**Store/index.js**

import { configureStore } from "@reduxjs/toolkit";

import authStore from "./auth";

import themeStore from "./theme";

const store = configureStore({

  reducer: {

    loging: authStore.reducer,

    mode: themeStore.reducer,

  },

});

export default store;

**App.js**

import { useState } from "react";

import { BrowserRouter as Router } from "react-router-dom";

import { useSelector } from "react-redux";

//pages

import Pages from "./Pages/Pages";

import Footer from "./Components/Footer";

//mui

import {

  ThemeProvider,

  createTheme,

  responsiveFontSizes,

} from "@mui/material/styles";

import { grey } from "@mui/material/colors";

function App() {

  const Tmode = useSelector((state) => state.mode.mode);

  const [mode, setMode] = useState(Tmode);

  let theme1 = createTheme({

    typography: {

      mode,

      primary: {

        main: "#073050",

      },

    },

    palette: {

      mode: mode,

      ...(mode === "light"

        ? {

            primary: {

              main: "#62BB46",

            },

            status: {

              danger: "#E28743",

            },

            background: {

              default: "#073050",

              paper: "#fff",

              button: "#073050",

            },

            divider: "#62BB46",

            secondary: {

              main: "#62BB46",

            },

            text: {

              primary: "#073050",

              secondary: "#073050",

            },

            success: {

              main: "#073050",

            },

            info: {

              main: "#1597BB",

            },

          }

        : {

            primary: {

              main: "#62BB46",

              dark: "#073050",

            },

            secondary: {

              main: "#62BB46",

            },

            divider: "#62BB46",

            background: {

              default: "#073050",

              paper: "#273443",

            },

            components: {

              MuiButton: {

                styleOverrides: {

                  outlined: {

                    backgroundColor: "green",

                  },

                },

              },

            },

            text: {

              primary: "#ddd",

              secondary: grey[400],

            },

            typography: {

              primary: "#fff",

              fontSize: 12,

            },

            success: {

              main: "#ECB365",

            },

            info: {

              main: "#aaa",

            },

          }),

    },

  });

  theme1 = responsiveFontSizes(theme1);

  theme1.typography.h3 = {

    fontSize: "1.2rem",

    "@media (min-width:100px)": {

      fontSize: "1.5rem",

    },

    [theme1.breakpoints.up("md")]: {

      fontSize: "2rem",

    },

  };

  theme1.typography.h4 = {

    fontSize: ".6rem",

    "@media (min-width:100px)": {

      fontSize: ".75rem",

    },

    [theme1.breakpoints.up("md")]: {

      fontSize: "1rem",

    },

  };

  theme1.typography.subtitle2 = {

    fontSize: ".5rem",

    "@media (min-width:100px)": {

      fontSize: ".5rem",

    },

    [theme1.breakpoints.up("md")]: {

      fontSize: ".75rem",

    },

  };

  const modeHandler = (value) => {

    setMode(value);

  };

  return (

    <div className="App">

      <ThemeProvider theme={theme1}>

        <Router>

          <Pages modeHandler={modeHandler} />

        </Router>

        <Footer />

      </ThemeProvider>

    </div>

  );

}

export default App;

**Appendix-Server**

**Db.js**

const mongoose = require("mongoose");

const mongoDbUrl =

  "mongodb+srv://nirojan:sliit2020@cluster0.8piet.mongodb.net/DS-Agri?retryWrites=true&w=majority";

let \_db;

const initDb = (callback) => {

  if (\_db) {

    return callback(null, \_db);

  }

  mongoose.connect(mongoDbUrl)

    .then((client) => {

      \_db = client;

      callback(null, \_db);

    })

    .catch((err) => {

      callback(err);

    });

};

const getDb = () => {

  if (!\_db) {

    throw Error("Database not initialzed");

  }

  return \_db;

};

module.exports = {

  initDb,

  getDb,

};

**App.js**

const express = require("express");

const cors = require("cors");

const BodyParser = require("body-parser");

const app = express();

//routes

const User = require("./Routes/User");

const Product = require("./Routes/Product");

const Order = require("./Routes/Order");

const Payment = require("./Routes/Payment");

//database

const db = require("./db");

//middlewares

app.use(BodyParser.urlencoded({ extended: false }));

app.use(express.json());

app.use(BodyParser.json());

app.use(cors());

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

  res.setHeader("Access-Control-Allow-Origin", "\*");

  res.setHeader(

    "Access-Control-Allow-Methods",

    "GET,POST,PUT,PATCH,DELETE,OPTIONS"

  );

  res.setHeader("Access-Control-Allow-Headers", "Content-Type");

  next();

});

app.use("/Uploads", express.static("Uploads"));

//routes

app.use("/users", User);

app.use("/api", Product);

app.use("/api", Order);

app.use("/api", Payment);

//initialize server and database

db.initDb((err, db) => {

  if (err) {

    console.log(err);

  } else {

    app.listen(5000);

  }

});

**Auth.js**

const jwt = require("jsonwebtoken");

const Users = require("../models/userModel");

module.exports = async (req, res, next) => {

  //check header

  const authHeader = req.get("Authorization");

  if (!authHeader) {

    res.status(400).json({ auth: "fail" });

    return;

  }

  const token = authHeader.split(" ")[1];

  if (!token || token === "") {

    res.status(400).json({ auth: "fail" });

    return;

  }

  let decodedToken;

  try {

    //check token

    decodedToken = jwt.verify(token, "Agriuservalidation");

  } catch (err) {

    res.status(400).json({ auth: "fail" });

    return;

  }

  if (!decodedToken) {

    res.status(400).json({ auth: "fail" });

    return;

  }

  req.userID = decodedToken.userID;

  userID = decodedToken.userID;

  const resp = await Users.findById(userID);

  if (resp) {

    req.role = resp.role;

  } else {

    res.status(400).json({ auth: "fail" });

    return;

  }

  req.auth = true;

  next();

};

**AuthAdmin.js**

module.exports = (req, res, next) => {

  if (req.role !== "farmer") {

    res.status(400).json({ auth: "fail" });

    return;

  }

  next();

};

**mailSender.js**

const nodemailer = require("nodemailer");

var transporter = nodemailer.createTransport({

  service: "gmail",

  auth: {

    user: "project2020sliit@gmail.com",

    pass: "sliit2022",

  },

  tls: {

    rejectUnauthorized: false,

  },

});

exports.mailSender = (to, subject, text) => {

  //mailing details

  var mailOptions = {

    from: "project2020sliit@gmail.com",

    to: to,

    subject: subject,

    html: text,

  };

  //send mail

  transporter.sendMail(mailOptions, function (error, info) {

    if (error) {

      console.log(error);

      return false;

    } else {

      return true;

    }

  });

};

**OrderModel.js**

const mongoose = require("mongoose");

const orderSchema = new mongoose.Schema(

  {

    user\_id: {

      type: String,

      required: true,

    },

    total: {

      type: Number,

      required: true,

    },

    payment: {

      type: Boolean,

      default: false,

    },

    address: {

      type: Object,

      default: {

        address: "",

        city: "",

        province: "",

        postalcode: "",

        country: "",

      },

    },

    products: {

      type: Object,

      default: {},

    },

    status: {

      type: Boolean,

      default: false,

    },

    date\_time: {

      type: String,

      default: Date.now(),

    },

  },

  {

    timestamps: true,

  }

);

module.exports = mongoose.model("orders", orderSchema);

**PaymentModel.js**

const mongoose = require("mongoose");

const paymentSchema = new mongoose.Schema(

  {

    user\_id: {

      type: String,

      required: true,

    },

    order\_id: {

      type: String,

      required: true,

      unique: true,

    },

    OTP: {

      type: Number,

      required: true,

    },

    amount: {

      type: Number,

      required: true,

    },

    method: {

      type: String,

      required: true,

    },

    mobile\_number: {

      type: Number,

      default: 0,

    },

    card\_number: {

      type: Number,

      default: 0,

    },

    expiry\_year: {

      type: Number,

      default: 0,

    },

    expiry\_month: {

      type: Number,

      default: 0,

    },

    cvv: {

      type: Number,

      default: 0,

    },

    name\_on\_card: {

      type: String,

      default: "",

    },

  },

  {

    timestamps: true,

  }

);

module.exports = mongoose.model("Payments", paymentSchema);

**ProductModel.js**

const mongoose = require("mongoose");

const productSchema = new mongoose.Schema(

  {

    id: {

      type: String,

      unique: true,

      required: true,

    },

    user\_id: {

      type: String,

      required: true,

    },

    title: {

      type: String,

      trim: true,

      required: true,

    },

    price: {

      type: Number,

      trim: true,

      required: true,

    },

    description: {

      type: String,

      required: true,

    },

    images: {

      type: String,

      required: true,

    },

    category: {

      type: String,

      required: true,

    },

    sold: {

      type: Number,

      default: 0,

    },

  },

  {

    timestamps: true,

  }

);

module.exports = mongoose.model("Products", productSchema);

**UserModel.js**

const mongoose = require("mongoose");

const userSchema = new mongoose.Schema(

  {

    OTP: {

      type: String,

      default:"",

    },

    firstName: {

      type: String,

      required: true,

      trim: true,

    },

    lastName: {

      type: String,

      required: true,

      trim: true,

    },

    email: {

      type: String,

      required: true,

      unique: true,

    },

    password: {

      type: String,

      required: true,

    },

    mobile\_number: {

      type: Number,

      default: 0,

    },

    role: {

      type: String,

      default: "client",

    },

    cart: {

      type: Array,

      default: [],

    },

    favorites: {

      type: Array,

      default: [],

    },

    address: {

      type: String,

      default: "",

    },

    images: {

      type: String,

      default: "",

    },

  },

  {

    timestamps: true,

  }

);

module.exports = mongoose.model("Users", userSchema);

**Routes/Order.js**

const Router = require("express").Router;

const router = Router();

const Order = require("../Controller/OrderCtrl");

const auth = require("../Middleware/auth");

router

  .route("/orders")

  .post(auth, Order.AddOrder)

  .get(auth, Order.GetOrder)

  .put(auth, Order.UpdateOrder);

router.get("/orders/:id", Order.GetOrders);

module.exports = router;

**Routes/Payment.js**

const Router = require("express").Router;

const router = Router();

const Payment = require("../Controller/PaymentCtrl");

const auth = require("../Middleware/auth");

router

  .route("/payments")

  .post(auth, Payment.AddPayment)

  .get(auth, Payment.GetPayment)

  .put(auth, Payment.CheckOTP)

  .delete(auth, Payment.DeletePayment);

module.exports = router;

**Routes/Product.js**

const Router = require("express").Router;

const router = Router();

const Product = require("../Controller/ProductCtrl");

const auth = require("../Middleware/auth");

const authAdmin = require("../Middleware/authAdmin");

const fileUpload = require("express-fileupload");

router.use(fileUpload());

router.get("/products/:\_id", auth, Product.GetProduct);

router

  .route("/products")

  .post(auth, authAdmin, Product.NewProduct)

  .get(auth, Product.GetProducts)

  .delete(auth, authAdmin, Product.DeleteProduct)

  .put(auth, authAdmin, Product.UpdateProduct);

module.exports = router;

**Routes/User.js**

const Router = require("express").Router;

const router = Router();

const User = require("../Controller/UserCtrl");

const auth = require("../Middleware/auth");

const fileUpload = require("express-fileupload");

router.use(fileUpload());

router.post("/login", User.Login);

router.put("/role", auth, User.ChangeRole);

router.post("/password", User.SendOtp);

router

  .route("/password/:\_id")

  .post(User.ResetPassword)

  .get(User.CheckResetValidity);

router

  .route("/favorites")

  .get(auth, User.GetFavorites)

  .put(auth, User.SetFavorite);

router

  .route("/carts")

  .put(auth, User.AddCart)

  .delete(auth, User.RemoveCartEle)

  .get(auth, User.getCart);

router

  .route("/dp/:id")

  .get(auth, User.GetDP)

  .delete(auth, User.DeleteDp)

  .post(auth, User.UploadDp);

router

  .route("/")

  .get(auth, User.GetUser)

  .put(auth, User.UpdateUser)

  .post(User.Register);

module.exports = router;

**//controllers**