**CSEE 5590 0002**

**Web and Mobile Programming**

**Project Source Code**

**Pranathi Nalluri**

**Sathwik Banginimatam**

**Ravindra Reddy Chennapareddy**

1. **Backend**
2. **Index.js**

require('dotenv').config();

const express = require("express");

const cors = require("cors");

const mongoose = require("mongoose");

const passport = require("passport");

const usersRouter = require("./routes/users");

const scrapperRouter = require("./routes/scrapper");

const app = express();

app.use(cors());

app.use(express.json());

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

// Database connection setup

mongoose

  .connect(`${process.env.MONGODB\_URL}`, {

    useNewUrlParser: true,

    useUnifiedTopology: true,

    useCreateIndex: true,

  })

  .then((res) => console.log("Connected to Database"))

  .catch((err) => console.log("Connection failed", err));

// Passport config

app.use(passport.initialize());

require("./config/passport")(passport);

// Routes configuration

app.use("/api/user/", usersRouter);

app.use("/api/scrapper/", scrapperRouter);

app.listen(5000, (err) => {

  if (err) {

    throw err;

  }

  console.log("Server running on 5000");

});

1. **Routes/users.js**

const express = require("express");

const router = express.Router();

const bcrypt = require("bcryptjs");

const jwt = require("jsonwebtoken");

const product = require('./../models/compare').product;

const compare = require('./../models/compare').compare;

// Load input validation

const validateRegisterInput = require("./../validations/register");

const validateLoginInput = require("./../validations/login");

// Load User model

const User = require("./../models/user");

// Register Router

router.post("/register", (req, res, next) => {

  const { errors, isValid } = validateRegisterInput(req.body);

  if (!isValid) {

    return res.status(400).json(errors);

  }

  console.log(req.body.email);

  User.findOne({ email: req.body.email }, (err, user) => {

    if (user) {

      return res.status(400).json({ message: "Email already exists" });

    } else {

      const newUser = new User({

        name: req.body.name,

        email: req.body.email,

        password: req.body.password,

      });

      // Hash Password

      bcrypt.genSalt(10, (err, salt) => {

        if (err) throw err;

        bcrypt.hash(newUser.password, salt, (err, hash) => {

          if (err) throw err;

          newUser.password = hash;

          newUser

            .save()

            .then((user) =>

              res.status(201).json({

                user,

              })

            )

            .catch((err) => console.log(err));

        });

      });

    }

  });

});

// Login User

router.post("/login", (req, res) => {

  const { isValid, errors } = validateLoginInput(req.body);

  console.log(errors, "in login api");

  if (!isValid) {

    return res.status(400).json(errors);

  }

  console.log("in login api");

  const email = req.body.email;

  const password = req.body.password;

  User.findOne({ email: email }).then((user) => {

    if (!user) {

      return res

        .status(400)

        .json({ message: "Email address is not registered" });

    }

    bcrypt.compare(password, user.password, (err, isMatch) => {

      if (err) throw err;

      if (!isMatch) {

        return res

          .status(400)

          .json({ message: "Please provide a correct password" });

      } else {

        //   Create JWT payload

        const payload = {

          id: user.\_id,

          name: user.name,

        };

        // Sign Token

        jwt.sign(

          payload,

          process.env.secretOrKey,

          { expiresIn: 36000 },

          (err, token) => {

            if (err) throw err;

            res.json({

              success: true,

              token: "Bearer " + token,

            });

          }

        );

      }

    });

  });

});

router.post("/save", (req, res) => {

  const { productsArray, userId } = req.body;

  console.log(req.body);

  const products = [];

  productsArray.forEach(prod => {

    const prodObj = new product({

      brand: prod.brand,

      name: prod.name,

      price: prod.price,

      rating: prod.rating,

      image: prod.image,

      link: prod.link,

    })

    products.push(prodObj);

  })

  const newCompareObj = new compare({

    userId: userId,

    products: products

  })

  newCompareObj.save().then(response => {

    return res.status(201).send({

      message: "Comparison saved successully",

    })

  }).catch(err => {

    return res.status(400).send({

      message: "Failed to saved comparison",

    })

  })

})

router.post('/comparisons', (req, res) => {

  const { userId } = req.body;

  compare.find({ userId: userId }).then(comparisons => {

    console.log(comparisons)

    return res.status(200).send({

      data: comparisons

    })

  }).catch(err => {

    console.log(err);

    return res.status(400).send({

      error: "Something went wrong.Please try again."

    })

  })

})

module.exports = router;

1. **Routes/scrapper.js**

const express = require("express");

const router = express.Router();

const productScrapper = require('./../controllers/productScrapper');

const dealScrapper = require('./../controllers/dealScrapper');

router.get('/offers', (req, res, next) => {

    dealScrapper.scrapper().then(response => {

        console.log(response);

        return res.status(200).send({

            data: response

        })

    }).catch(err => {

        console.log(err);

        return res.status(400).send({

            error: "Something went wrong.Please try again."

        })

    });

})

router.post("/", (req, res) => {

    const { query } = req.body;

    console.log(query);

    productScrapper.scrapper(query).then(response => {

        console.log(response);

        return res.status(200).send({

            data: response

        })

    }).catch(err => {

        console.log(err);

        return res.status(400).send({

            error: "Something went wrong.Please try again."

        })

    });

})

module.exports = router;

1. **Models/user.js**

const mongoose = require("mongoose");

const userSchema = new mongoose.Schema({

  name: {

    type: String,

    require: true,

  },

  email: {

    type: String,

    require: true,

  },

  password: {

    type: String,

    require: true,

  },

  created\_date: {

    type: Date,

    default: Date.now,

  },

});

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

1. **Models/compare.js**

const mongoose = require('mongoose');

const productSchema = new mongoose.Schema({

    id: String,

    brand: String,

    name: String,

    price: String,

    rating: String,

    image: String,

    link: String,

}, {

    timestamps: true

})

const comparisonSchema = new mongoose.Schema({

    userId: String,

    products: [productSchema],

    created: {

        type: Date, default: Date.now

    }

})

const product = mongoose.model('product', productSchema);

const compare = mongoose.model('compare', comparisonSchema)

module.exports = {

    product: product,

    compare: compare

};

1. **Controllers/dealScrapper.js**

const request = require('request-promise');

const cheerio = require('cheerio');

const websitesToSearch = require('./../config/websites\_data');

const getOffersData = async () => {

    let offersArray = [];

    // for (index = 0; index < websitesToSearch.length; index++) {

    const offersData = await getDataFromEbay(websitesToSearch[1]);

    const offersObj = {

        id: 1,

        brand: websitesToSearch[1].name,

        offers: offersData

    }

    offersArray.push(offersObj);

    console.log(offersArray);

    // }

    return offersArray;

}

const getOfferDataFromEachBrand = (index, websiteDetails) => {

    switch (index) {

        // case 0:

        //     return getDataFromAmazon(websiteDetails);

        case 1:

            return getDataFromEbay(websiteDetails);

        case 2:

            console.log("Getting Best Buy Data");

            return getDataFromBestBuy(websiteDetails);

        default:

            console.log("Default case")

    }

}

var getDataFromAmazon = async (websiteDetails) => {

    const result = await request.get(websiteDetails['deal\_scrapper\_url']);

    const $ = await cheerio.load(result);

    let brandOfferData = [];

    const firstThreeOffers = $('#widgetContent').children;

    // console.log(firstThreeOffers, "In Amazon offers");

    console.log(firstThreeOffers.length);

    return brandOfferData;

}

var getDataFromEbay = async (websiteDetails) => {

    const result = await request.get(websiteDetails['deal\_scrapper\_url']);

    const $ = await cheerio.load(result);

    const offersData = $('.ebayui-dne-item-featured-card').find('.dne-itemtile').slice(0, 10);

    let brandOfferData = [];

    offersData.each((i, el) => {

        const title = $(el).find('.dne-itemtile-title').attr('title');

        const id = $(el).find('.dne-itemtile').attr('data-listing-id');

        const imageLink = $(el).find('.slashui-image-cntr').find('img').attr('src');

        const reducedPrice = $(el).find('.dne-itemtile-price').find('.first').text();

        const link = $(el).find('.dne-itemtile-detail').find('a').attr('href');

        const originalPrice = $(el).find('.dne-itemtile-original-price').find('.itemtile-price-strikethrough').text();

        const offerValue = $(el).find('.dne-itemtile-original-price').find('.itemtile-price-bold').text();

        const datas = { id, title, imageLink, reducedPrice, offerValue, originalPrice, link };

        brandOfferData.push(datas);

    })

    return brandOfferData;

}

var getDataFromBestBuy = async (websiteDetails) => {

    console.log("Getting Buest buy data inside method");

    console.log(websiteDetails['scrapper\_url']);

    request.get(websiteDetails['scrapper\_url']).then(res => {

        console.log(res)

    }).catch(err => console.log(err));

    try {

        const result = await request.get(websiteDetails['scrapper\_url']);

        console.log(result);

        const $ = await cheerio.load(result);

        let brandData = [];

    } catch (e) {

        console.log(e)

    }

    const firstFiveProducts = $('li.sku-item').slice(0, 5);

    firstFiveProducts.each((i, el) => {

        const id = $(el).attr('.sku-value').text();

        const name = $(el).find('h4.sku-header').find('a').text();

        const price = $(el).find('.priceView-customer-price').find('span').text();

        const image = $(el).find('.product-image').attr('src');

        const link = websiteDetails['website\_link'] + $(el).find('h4.sku-header').find('a').attr('href');

        const datas = { id, name, price, image, link };

        if (name !== '' || price !== '') {

            brandData.push(datas);

        }

    });

    return brandData;

}

module.exports = {

    scrapper: getOffersData

}

1. **Controllers/productScrapper.js**

const request = require('request-promise');

const cheerio = require('cheerio');

const websitesToSearch = require('./../config/websites\_data');

const getProductData = async (query) => {

    console.log(query)

    let productsArray = [];

    for (index = 0; index < websitesToSearch.length; index++) {

        const productData = await getDataFromEachBrand(index, websitesToSearch[index], query);

        const productsObj = {

            id: index + 1,

            brand: websitesToSearch[index].name,

            products: productData

        }

        productsArray.push(productsObj);

        console.log(productsArray);

    }

    return productsArray;

}

const getDataFromEachBrand = (index, websiteDetails, query) => {

    switch (index) {

        case 0:

            return getDataFromAmazon(websiteDetails, query);

        case 1:

            return getDataFromEbay(websiteDetails, query);

        case 2:

            console.log("Getting Best Buy Data");

            return getDataFromBestBuy(websiteDetails, query);

        default:

            console.log("Default case")

    }

}

var getDataFromAmazon = async (websiteDetails, query) => {

    const result = await request.get(websiteDetails['scrapper\_url'] + query);

    const $ = await cheerio.load(result);

    let brandData = [];

    const firstFiveProducts = $('.s-asin').slice(0, 3);

    firstFiveProducts.each((i, el) => {

        const id = $(el).attr('data-asin');

        const name = $(el).find('h2 span').text();

        const price = $(el).find('.a-price-whole').text().replace(',', '');

        const rating = $(el).find('.a-spacing-top-micro span').attr('aria-label');

        const image = $(el).find('.s-image').attr('src');

        const link = websiteDetails['website\_link'] + $(el).find('.a-link-normal').attr('href');

        const datas = { id, name, price: '$' + (+price / 75).toFixed(2), rating, image, link, brand: 'Amazon' };

        brandData.push(datas);

    });

    return brandData;

}

var getDataFromEbay = async (websiteDetails, query) => {

    const result = await request.get(websiteDetails['scrapper\_url'] + query);

    const $ = await cheerio.load(result);

    let brandData = [];

    const firstFiveProducts = $('.s-item').slice(1, 4);

    firstFiveProducts.each((i, el) => {

        const id = i + 1;

        const name = $(el).find('h3.s-item\_\_title').text();

        const price = $(el).find('.s-item\_\_price').text();

        const image = $(el).find('.s-item\_\_image-img').attr('src');

        const link = $(el).find('.s-item\_\_link').attr('href');

        // const id = link.split('?')[1].split('&')[0].split('=')[1];

        const datas = { id, name, price, image, link, brand: 'EBay' };

        if (name !== '' || price !== '') {

            brandData.push(datas);

        }

    });

    return brandData;

}

var getDataFromBestBuy = async (websiteDetails, query) => {

    console.log("Getting Buest buy data inside method");

    console.log(websiteDetails['scrapper\_url'] + query);

    request.get(websiteDetails['scrapper\_url'] + query).then(res => {

        console.log(res)

    }).catch(err => console.log(err));

    try {

        const result = await request.get(websiteDetails['scrapper\_url'] + query);

        console.log(result);

        const $ = await cheerio.load(result);

        let brandData = [];

    } catch (e) {

        console.log(e)

    }

    const firstFiveProducts = $('li.sku-item').slice(0, 5);

    firstFiveProducts.each((i, el) => {

        const id = $(el).attr('.sku-value').text();

        const name = $(el).find('h4.sku-header').find('a').text();

        const price = $(el).find('.priceView-customer-price').find('span').text();

        const image = $(el).find('.product-image').attr('src');

        const link = websiteDetails['website\_link'] + $(el).find('h4.sku-header').find('a').attr('href');

        const datas = { id, name, price, image, link };

        if (name !== '' || price !== '') {

            brandData.push(datas);

        }

    });

    return brandData;

}

module.exports = {

    scrapper: getProductData

}

1. **Config/passport.js**

require('dotenv').config();

const JWTStrategy = require("passport-jwt").Strategy;

const ExtractJwt = require("passport-jwt").ExtractJwt;

const User = require("./../models/user");

let options = {};

options.jwtFromRequest = ExtractJwt.fromAuthHeaderAsBearerToken();

options.secretOrKey = process.env.secretOrKey;

const passportAuth = (passport) => {

  passport.use(

    new JWTStrategy(options, (jwt\_payload, done) => {

      User.findById(jwt\_payload.id)

        .then((user) => {

          if (user) {

            return done(null, user);

          }

          return done(null, false);

        })

        .catch((err) => console.log(err));

    })

  );

};

module.exports = passportAuth;

1. **Config/websites\_data.js**

const websitesToSearch = [

    {

        id: 1,

        name: 'amazon',

        scrapper\_url: "https://www.amazon.in/s?k=",

        deal\_scrapper\_url: "https://www.amazon.in/gp/goldbox",

        website\_link: "https://www.amazon.in/",

    },

    {

        id: 2,

        name: 'ebay',

        deal\_scrapper\_url: "https://www.ebay.com/deals",

        scrapper\_url: "https://www.ebay.com/sch/i.html?&\_nkw=",

    },

]

module.exports = websitesToSearch;

1. **Frontend**
2. **Index.js**

import React from 'react';

import ReactDOM from 'react-dom';

import App from './App';

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

import { Provider } from "react-redux";

import store from "./store/store";

import './index.css';

ReactDOM.render(<Provider store={store}>

  <Router>

    <React.StrictMode>

      <App />

    </React.StrictMode>

  </Router>

</Provider>, document.getElementById('root'));

1. **App.js**

import React from "react";

import "./App.css";

import Navbar from './components/Navbar';

import Home from './components/Home';

import About from './pages/About/About';

import Landing from './pages/Landing/Landing';

import Register from './pages/Authentication/Register'

import Login from './pages/Authentication/Login';

import { Redirect, Route, Switch } from 'react-router-dom';

import jwt\_decode from "jwt-decode";

import setAuthToken from "./utils/setAuthToken";

import store from "./store/store";

import { setCurrentUser, logoutUser } from "./store/actions/authActions";

import Offers from "./components/Offers";

import MyAccount from "./pages/MyAccount/MyAccount";

class App extends React.Component {

  componentDidMount() {

    // Check for token to keep user logged in

    if (localStorage.jwtToken) {

      // Set auth token header auth

      const token = localStorage.jwtToken;

      setAuthToken(token);

      // Decode token and get user info and exp

      const decoded = jwt\_decode(token);

      // Set user and isAuthenticated

      store.dispatch(setCurrentUser(decoded));

      // Check for expired token

      const currentTime = Date.now() / 1000; // to get in milliseconds

      if (decoded.exp < currentTime) {

        // Logout user

        store.dispatch(logoutUser());

        // Redirect to login

        window.location.href = "./login";

      }

    }

  }

  render() {

    return (

      <div className="App" >

        <Navbar />

        <div className="main-section">

          <Switch>

            <Route path="/about" exact><About /></Route>

            <Route path="/home" exact><Home /></Route>

            <Route path="/account" exact><MyAccount /></Route>

            <Route path="/offers" exact><Offers /></Route>

            <Route path="/landing" exact><Landing /></Route>

            <Route exact path="/register"><Register /></Route>

            <Route exact path="/login"><Login /></Route>

            <Redirect to="/landing" path="/"></Redirect>

          </Switch>

        </div>

      </div>

    )

  };

}

export default App;

1. **Src/pages/About**

import React from "react";

//import { Image, ScrollView, Text } from "react-native";

import "./About.css"

import Projectdemo from "./../../assets/images/projectdemo.png";

import AmazonLogo from "./../../assets/images/amazon1.jpg";

import EbayLogo from './../../assets/images/ebay.png';

import Walmartlogo from "./../../assets/images/walmart.png";

import Bestbuylogo from "./../../assets/images/bestbuylogo.png";

import Jeanine from "./../../assets/images/jeanine.png";

const About = () => {

    return (

        <>

            <div className="mainpage">

                <div>

                    <h1 id="aboutit"> About us </h1>

                    <p className="text-para"> The main objective of application is to provide users a platform which collects the

                    information about the product of user interest like price, reviews and if there are any offers

                    available in those ecommerce sites. The application can save both time and money for the

                    customers and can also help the ecommerce platforms to get customer attention if they can

        offer great deals. </p>

                    <p className="text-para">In recent years, the usage of e-commerce platforms for purchasing the needs became a

                    daily routine of many people as it saves them lot of time and provides many options which

                    can be ordered from online. But the main problem that a lot of customers face is spending

                    much money for the product as they don’t get time to check different websites to compare

the deals. It would be a good idea to have an application which he</p>

                </div>

                <div className="benfits">

                    <img className="img-demo" src={Projectdemo} />

                    <p className="img-para" >

                        Usually following methods are followed to get the product data from ecommerce websites:<br></br><br></br>

                        1.  Feed from Sellers<br></br>

                        2.  Third party API’s<br></br>

                        3.  Web Scrapping<br></br><br></br>

                        We will use Web Scraping, which is a secure and efficient method of obtaining data from various ecommerce sites, as obtaining data from Sellers and third-party APIs is difficult and costly.

            </p>

                </div>

                <div className="aboutus-brands">

                    <h3 className="heading"> Brands </h3>

                    <div className="brand-logos">

                        <img className="logoimg" src={AmazonLogo} />

                        <img className="logoimg" src={EbayLogo} />

                        <img className="logoimg" src={Walmartlogo} />

                        <img className="logoimg" src={Bestbuylogo} />

                    </div>

                </div>

                <div className="usecases">

                    <h3 className="heading">Stories</h3>

                    <div className="reviewsetup">

                        <img className="userimg" src={Jeanine} />

                        <p class="parafont">“Jeanine, working as a Team Manager in a MNC and a mother of two children who likes to visit different stores for shopping during her teenage. In recent years she is not able to find time in between her professional and family life to go for shopping. So she started purchasing the things online, but she is not able to spend much time to check different e-commerce platforms to compare the deals in each site due to her busy schedules”</p>

                        <p class="parafont">“Jeanine got an idea thinking about her problem and asked her team whether any websites are available to compare the products across different ecommerce websites, the team suggested her few options available. She has gone through those websites and liked the overall features available. However, she felt like there are few more features that can be added to the existing functionality”.</p>

                    </div>

                </div>

            </div>

        </>

    );

}

export default About;

.mainpage {

    padding: 0 2rem;

}

.text-para{

    text-indent:50px;

    text-align: left;

    text-align: justify;

    width: 75%;

    margin: 0 auto;

}

    .text-para::first-letter{

        font-size:20px;

        font-style: italic;

    }

.img-demo{

    width: 350px;

}

.img-para{

    width: 40rem;

}

.brand-logos {

display: flex;

justify-content: space-between;

align-items: center;

}

.logoimg {

    margin-right:25px;

    width:200px;

    height:200px;

    border-radius: 50%;

    object-fit: cover;

}

.heading{

    color: rgb(58,44,44);

}

.parafont{

    /\* width:40rem; \*/

    margin-left: 1rem;

    font-style:italic ;

    font-family: cursive;

   display: table-cell;

    /\*flex:coloumn;\*/

}

#aboutit{

    color: rgb(58,44,44);

    text-align: center;

}

.userimg{

    border-radius: 50%;;

    width:25%;

    height:25%;

}

#theend{

    padding: 100px;

}

.hi{

    display:flex;

    padding: 50px;;

    align-items: center;

    justify-content: center;

}

.aboutus-brands {

    padding: 0 2rem;

    width: 75%;

    margin: 0 auto;

}

.usecases {

    padding: 0 2rem;

    width: 75%;

    margin: 4rem auto;

}

.benfits {

    display:flex;

    justify-content: space-between;

    align-items: center;

    width: 75%;

    margin: 4rem auto;

}

.benfits img {

    margin-right: 2rem;

}

.reviewsetup {

    display:flex;

    justify-content: space-between;

    align-items: center;

}

1. **Src/pages/Authentication**

import React from "react";

import { Link, withRouter } from "react-router-dom";

import "./Authentication.css";

import PropTypes from "prop-types";

import { connect } from "react-redux";

import { loginUser } from "./../../store/actions/authActions";

class Login extends React.Component {

  constructor(props) {

    super(props);

    this.state = {

      email: "",

      password: "",

      errors: {},

      visible: false,

    };

    console.log(this.props);

  }

  onDismiss = () =>

    this.setState({

      visible: false,

    });

  componentWillReceiveProps(nextProps) {

    if (nextProps.auth.isAuthenticated) {

      this.props.history.push("/dashboard"); // push user to dashboard when they login

    }

    if (nextProps.errors) {

      this.setState({

        errors: nextProps.errors,

        visible: true,

      });

    }

  }

  componentDidMount() {

    // If logged in and user navigates to Register page, should redirect them to dashboard

    if (this.props.auth.isAuthenticated) {

      this.props.history.push("/dashboard");

    }

  }

  onChange = (event) => {

    this.setState({

      [event.target.name]: event.target.value,

    });

  };

  loginHandler = (e) => {

    e.preventDefault();

    const userObj = {

      email: this.state.email,

      password: this.state.password,

    };

    console.log(userObj);

    this.props.loginUser(userObj);

  };

  render() {

    const errorMessages = Object.keys(this.state.errors).map((el) => {

      return (

        <div

          key={el}

          className="alert alert-warning alert-dismissible fade show"

          role="alert"

        >

          {this.state.errors[el]}

        </div>

      );

    });

    return (

      <div className="auth-container">

        <h2 className="section-title">Sign In</h2>

        <form className="section-form">

          {errorMessages.length > 0 && (

            <div className="errors row">

              <div className="messages">{errorMessages}</div>

            </div>

          )}

          <div className="form-group row">

            <div className="col-md-4 offset-md-1">

              <label htmlFor="email" className="col-form-label">

                Email Address

              </label>

            </div>

            <div className="col-md-6">

              <input

                type="text"

                name="email"

                id="email"

                value={this.state.email}

                className="form-control"

                placeholder="Enter Email address"

                onChange={this.onChange}

              />

            </div>

          </div>

          <div className="form-group row">

            <div className="col-md-4 offset-md-1">

              <label htmlFor="password" className="col-form-label">

                Password

              </label>

            </div>

            <div className="col-md-6">

              <input

                type="password"

                name="password"

                id="password"

                value={this.state.password}

                className="form-control"

                placeholder="Enter password"

                onChange={this.onChange}

              />

            </div>

          </div>

          <div className="row align-items-center">

            <div className="col-md-3 offset-md-5">

              <button

                type="submit"

                className="btn btn-info btn-Login"

                onClick={this.loginHandler}

              >

                Login

              </button>

            </div>

            <div className="col-md-4">

              <p className="forgot-pwd mt-2">

                <Link to="/register" className="btn-link">

                  Forgot Password?

                </Link>

              </p>

            </div>

          </div>

        </form>

        <div className="btn-section row mt-2">

          <div className="col-md-12 offset-md-5">

            <p className="login-link">

              Don't have an account ?{" "}

              <Link to="/register" className="btn-link">

                Register

              </Link>

            </p>

          </div>

        </div>

      </div>

    );

  }

}

Login.propTypes = {

  loginUser: PropTypes.func.isRequired,

  auth: PropTypes.object.isRequired,

  errors: PropTypes.object,

};

const mapStateToProps = (state) => ({

  auth: state.authReducer,

  errors: state.errorReducer["errors"],

});

export default connect(mapStateToProps, { loginUser })(withRouter(Login));

import React from "react";

import { Link, withRouter } from "react-router-dom";

import "./Authentication.css";

import { connect } from "react-redux";

import PropTypes from "prop-types";

import { registerUser } from "./../../store/actions/authActions";

class Register extends React.Component {

  constructor(props) {

    super(props);

    this.state = {

      name: "",

      email: "",

      password: "",

      password2: "",

      errors: {},

    };

    console.log(this.props.history, "in Register component");

    this.onChange = this.onChange.bind(this);

    // this.registerHandler = this.registerHandler.bind(this);

  }

  componentWillReceiveProps(nextProps) {

    if (nextProps.errors) {

      this.setState({

        errors: nextProps.errors,

      });

    }

  }

  onChange(event) {

    this.setState({

      [event.target.name]: event.target.value,

    });

  }

  componentDidMount() {

    // If logged in and user navigates to Register page, should redirect them to dashboard

    if (this.props.auth.isAuthenticated) {

      this.props.history.push("/home");

    }

  }

  registerHandler = (event) => {

    event.preventDefault();

    const userObj = {

      name: this.state.name,

      email: this.state.email,

      password: this.state.password,

      password2: this.state.password2,

    };

    console.log(userObj);

    this.props.registerUser(userObj, this.props.history);

  };

  render() {

    const errorMessages = Object.keys(this.state.errors).map((el) => {

      return (

        <div

          key={el}

          className="alert alert-warning alert-dismissible fade show"

          role="alert"

        >

          {this.state.errors[el]}

        </div>

      );

    });

    return (

      <div className="auth-container">

        <h2 className="section-title">Sign Up</h2>

        <form className="reg-form">

          {errorMessages.length > 0 && (

            <div className="errors row">

              <div className="messages">{errorMessages}</div>

            </div>

          )}

          <div className="form-group row">

            <div className="col-md-4 offset-md-1">

              <label htmlFor="name" className="col-form-label">

                Name

              </label>

            </div>

            <div className="col-md-6">

              <input

                type="text"

                name="name"

                id="name"

                value={this.state.name}

                className="form-control"

                placeholder="Enter name"

                onChange={this.onChange}

              />

            </div>

          </div>

          <div className="form-group row">

            <div className="col-md-4 offset-md-1">

              <label htmlFor="email" className="col-form-label">

                Email Address

              </label>

            </div>

            <div className="col-md-6">

              <input

                type="text"

                name="email"

                id="email"

                value={this.state.email}

                className="form-control"

                placeholder="Enter Email address"

                onChange={this.onChange}

              />

            </div>

          </div>

          <div className="form-group row">

            <div className="col-md-4 offset-md-1">

              <label htmlFor="password" className="col-form-label">

                Password

              </label>

            </div>

            <div className="col-md-6">

              <input

                type="password"

                name="password"

                id="password"

                value={this.state.password}

                className="form-control"

                placeholder="Enter password"

                onChange={this.onChange}

              />

            </div>

          </div>

          <div className="form-group row">

            <div className="col-md-4 offset-md-1">

              <label htmlFor="password2" className="col-form-label">

                Confirm Password

              </label>

            </div>

            <div className="col-md-6">

              <input

                type="password"

                name="password2"

                id="password2"

                value={this.state.password2}

                className="form-control"

                placeholder="Confirm Password"

                onChange={this.onChange}

              />

            </div>

          </div>

          <div className="row">

            <div className="col-md-12 offset-md-5">

              <button

                type="submit"

                className="btn btn-info btn-register"

                onClick={this.registerHandler}

              >

                Register

              </button>

            </div>

          </div>

        </form>

        <div className="row mt-2 btn-section">

          <div className="col-md-12 offset-md-5">

            <p className="login-link">

              Already Registered?{" "}

              <Link to="/login" className="btn-link">

                Login

              </Link>

            </p>

          </div>

        </div>

      </div>

    );

  }

}

const mapStateToProps = (state) => {

  return {

    auth: state.authReducer,

    errors: state.errorReducer["errors"],

  };

};

Register.propTypes = {

  registerUser: PropTypes.func.isRequired,

  auth: PropTypes.object.isRequired,

  errors: PropTypes.object,

};

const mapDispatchToProps = {

  registerUser: registerUser,

};

export default connect(

  mapStateToProps,

  mapDispatchToProps

)(withRouter(Register));

.auth-container {

    width: 50%;

    margin: 0 auto;

    background: #fff;

    margin-top: 2rem;

    border-radius: 10px;

}

.section-title {

    text-align: center;

    margin-bottom: 1rem;

    color: #3B4258;

}

.section-form {

    margin-top: 1rem;

    color: #3B4258;

}

a.btn-link {

    color: #3B4258;

    text-decoration: none;

    font-size: 700;

}

a.btn-link:hover {

    color: #95D43D;

    text-decoration: none;

}

.form-group.row {

    display: flex;

    justify-content: space-between;

    align-items: center;

    margin-bottom: 1.5rem;

}

.form-group label{

    /\* margin-right: 2rem; \*/

    width: 55%;

}

.col-md-6 {

    width:70%;

}

.form-control {

    width: 95%;

    height: 1.2rem;

    border: 1px solid #ccc;

    outline: none;

    border-radius: 3px;

    padding: 0.5rem;

    font-size: 1rem;

    box-shadow: 0 3px 3px #ccc;

}

.btn.btn-info {

    background: #101522;

    color: #fff;

    padding: 0.7rem 2.5rem;

    outline: none;

    font-size: 1rem;

    border: none;

    border-radius: 2px;

    cursor: pointer;

}

.btn:hover{

    background:#4b59f7;

}

.row {

    display: flex;

    justify-content: flex-end;

    align-items: center;

}

.row.errors {

    justify-content: center;

}

a.btn-link {

    display: inline-block;

}

.alert {

    width: 100%;

}

.alert-warning {

    color: #721c24;

    margin-top: 1rem;

}

.errors {

    background: #f8d7da;

    padding: 1rem;

    margin-bottom: 1rem;

}

.messages {

    width: 80%;

    margin: 0 auto;

    text-transform: capitalize;

}

**c) src/pages/Landing**

import React from 'react';

import './Landing.css';

import one from './../../assets/images/one.svg';

import two from './../../assets/images/two.svg';

import currency from './../../assets/images/currency.svg';

import discount from './../../assets/images/discount.svg';

import share from './../../assets/images/share.svg';

const Landing = () => {

    return (

        <div className="landing-container">

            <section className="info info1">

                <div className="info-text">

                    <h2 className="info-header">Online support to compare prices </h2>

                    <p className="info-desc">The main objective of application is to provide users a platform which collects the information about the product of user interest like price, reviews and if there are any offers available in those ecommerce sites. The application can save both time and money for the customers and can also help the ecommerce platforms to get customer attention if they can offer great deals.</p>

                </div>

                <div className="info-image">

                    <img src={one} alt="image-1" className="image image1"/>

                </div>

            </section>

            <section className="info info2">

                <div className="info-text">

                    <h2 className="info-header">Help End-users save time and money </h2>

                    <p className="info-desc">“Jeanine, working as a Team Manager in a MNC and a mother of two children who likes to visit different stores for shopping during her teenage. In recent years she is not able to find time in between her professional and family life to go for shopping. So she started purchasing the things online, but she is not able to spend much time to check different e-commerce platforms to compare the deals in each site due to her busy schedules"</p>

                </div>

                <div className="info-image">

                    <img src={two} alt="image-1" className="image image1"/>

                </div>

            </section>

            <div className="services">

                <div className="services-wrapper">

                    <h2 className="services-title">

                        Our Services

                    </h2>

                    <div className="services-card">

                        <div className="service-card">

                            <div className="service-icon">

                                <img src={currency} alt="image-1" className="image image1" width="150px" height="100px"/>

                            </div>

                            <h3 className="service-label">

                                Price

                            </h3>

                            <p className="service-desc">

                                Get price of the product from different e-commerce sites.

                            </p>

                        </div>

                        <div className="service-card">

                            <div className="service-icon">

                                <img src={discount} alt="image-1" className="image image1" width="150px" height="100px"/>

                            </div>

                            <h3 className="service-label">

                                Offers

                            </h3>

                            <p className="service-desc">

                                Know the offers on e-commerce sites and shop.

                            </p>

                        </div>

                        <div className="service-card">

                            <div className="service-icon">

                                <img src={share} alt="image-1" className="image image1" width="150px" height="100px"/>

                            </div>

                            <h3 className="service-label">

                                Share

                            </h3>

                            <p className="service-desc">

                                It is a great idea to share the comparison with friends.

                            </p>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    )

}

export default Landing;

.landing-container {

    width: 100%;

    /\* max-width: 1100px; \*/

    margin: 0 auto;

}

.landing-container .info {

    background: #101522;

    color:white;

    padding: 5rem 3rem;

    display: flex;

    flex-direction: row;

    align-items: center;

    justify-content: center;

}

.info div {

    width: 40%;

    margin-right:2rem;

}

.info-header {

    font-size: 3rem;

}

.info-image img {

    width: 100%;

}

.landing-container .info2 {

    background: #fff;

    color: #101522;

    flex-direction: row-reverse;

}

.services {

    width: 100%;

    background: #4b59f7;

    color: #fff;

}

.services-wrapper {

    max-width: 80%;

    margin: 0 auto;

    padding: 3rem;

}

h2.services-title {

    margin-top:0;

    text-align: center;

    font-size: 2rem;

}

.services-card {

    display: flex;

    justify-content: center;

    align-items: center;

}

.service-card {

    margin-right: 2rem;

    text-align: center;

    margin-bottom: 1rem;

    padding: 1rem;

}

1. **Src/pages/MyAccount**

import React from 'react';

import { connect } from 'react-redux';

import { withRouter } from 'react-router-dom';

import ComparisonModal from '../../components/ComparisonModal';

import { getComparisons } from '../../store/actions/accountActions';

class MyAccount extends React.Component {

    constructor(props) {

        super(props);

        this.state = {

            showComparisonModal: false,

            selectedComp: null

        }

    }

    componentDidMount() {

        if (this.props.user) {

            this.props.getComparisons(this.props.user.id)

        } else {

            this.props.history.push('/login');

        }

    }

    openDetails = (event, comp) => {

        event.preventDefault();

        this.setState({

            showComparisonModal: true,

            selectedComp: comp

        })

    }

    closeModal = () => {

        this.setState({

            showComparisonModal: false,

        })

    }

    componentDidUpdate() {

        console.log(this.props.comparisons);

    }

    render() {

        return (

            <>

                <div className="account-container">

                    <h3 className="section-title">

                        My Account

                    </h3>

                    {this.props.comparisons.length === 0 ? (

                        <div className="empty">There are no comparisons added yet.</div>

                    ) : (<div className="account-table">

                        <div className="table-header">

                            <div className="index">S.No</div>

                            <div className="product-header">Product A</div>

                            <div className="product-header">Product B</div>

                            <div className="empty"></div>

                        </div>

                        <div className="table-content">

                            {this.props.comparisons.map(comp => {

                                return (

                                    <div className="content-row">

                                        <div className="index">{comp.\_id}</div>

                                        <div className="product-name">

                                            {comp.products[0].name.substr(0, 40)}

                                        </div>

                                        <div className="product-name">

                                            {comp.products[1].name.substr(0, 40)}

                                        </div>

                                        <div className="actions">

                                            <a href="#" className="btn btn-primary" onClick={(event) => this.openDetails(event, comp)}>Get Details</a>

                                        </div>

                                    </div>

                                )

                            })}

                        </div>

                    </div>)

                    }

                    {this.state.showComparisonModal ? (<ComparisonModal show={this.state.showComparisonModal} closeModal={this.closeModal} productsToCompare={[...this.state.selectedComp.products]} />) : null}

                </div >

            </>

        )

    }

}

const mapStateToProps = (state) => ({

    comparisons: state.accountReducer.comparisons,

    user: state.authReducer.user,

})

const mapDispatchToProps = (dispatch) => {

    return {

        getComparisons: (userId) => dispatch(getComparisons(userId))

    }

}

export default connect(mapStateToProps, mapDispatchToProps)(withRouter(MyAccount));

1. **Components/home.js**

import React from 'react';

import { connect } from 'react-redux';

import './Home.css';

import { getProducts, saveComparison } from './../store/actions/productActions';

import Brands from './Brands';

import SideMenu from './SideMenu';

import SearchInput from './SearchInput';

import Products from './Products';

import Spinner from './Spinner';

import ComparisonModal from './ComparisonModal';

import { ToastContainer, toast } from 'react-toastify';

import 'react-toastify/dist/ReactToastify.css';

import { Slide } from 'react-toastify';

class Home extends React.Component {

    constructor(props) {

        super(props);

        this.state = {

            showProducts: false,

            message: '',

            productsAddedForComparison: [],

            selectedProductIds: [],

            showToast: false,

            showComparisonModal: false

        }

    }

    onSearchHandler = async (searchTerm) => {

        if (searchTerm) {

            await this.props.getProducts(searchTerm);

        } else {

            this.setState({

                errorMsg: "Please enter a product name to get the results"

            })

        }

    }

    showToast = (message, flag) => {

        const position = { position: toast.POSITION.TOP\_RIGHT };

        switch (flag) {

            case "success":

                toast.success(message, position);

                break;

            case "error":

                toast.error(message, position);

                break;

            default:

                toast.info(message, position);

                break;

        }

    };

    comparisonHandler = (product) => {

        const index = this.state.selectedProductIds.indexOf(product.id);

        if (index >= 0) {

            const selectedProds = [...this.state.selectedProductIds];

            selectedProds.splice(index, 1);

            const updatedProducts = this.state.productsAddedForComparison.filter(pr => pr.id != product.id);

            this.setState({

                selectedProductIds: selectedProds,

                productsAddedForComparison: updatedProducts

            })

        } else {

            if (this.state.productsAddedForComparison.length < 2) {

                const productsAddedForComparison = [...this.state.productsAddedForComparison];

                productsAddedForComparison.push(product);

                const selectedProductIds = productsAddedForComparison.map(p => p.id)

                this.setState({

                    productsAddedForComparison: productsAddedForComparison,

                    showToast: false,

                    selectedProductIds: selectedProductIds

                }, () => console.log(this.state))

            } else {

                this.setState({

                    showToast: true

                })

                this.showToast("Maximum of two product can be selected for comparison", "error");

            }

        }

    }

    openComparisonModal = () => {

        console.log("in Method")

        this.setState({

            showComparisonModal: !this.state.showComparisonModal

        })

    }

    saveComparisonHandler = async () => {

        const dataObj = {

            userId: this.props.user.id,

            productsArray: [...this.state.productsAddedForComparison]

        }

        this.props.saveComparison(dataObj).then(resp => {

            this.setState({

                showComparisonModal: false,

                productsAddedForComparison: [],

                selectedProductIds: [],

                showToast: true

            })

            this.showToast(resp['data'].message, "success");

        });

    }

    render() {

        return (

            <div className='container-page'>

                <SideMenu />

                {this.state.showToast && <ToastContainer transition={Slide} />}

                <div className='grid-item2'>

                    {this.state.errorMsg && (<div className="error">

                        <p className="error-msg">{this.state.errorMsg}</p>

                    </div>)}

                    <SearchInput onSearchHandler={this.onSearchHandler} />

                    {this.props.showLoader ? <Spinner /> : this.props.brands.length > 0 ? <Products brands={this.props.brands} addForComparison={this.comparisonHandler} selectedProductIds={this.state.selectedProductIds} />

                        : <Brands />

                    }

                    {this.state.productsAddedForComparison.length === 2 ? (<a className="btn btn-primary compare-btn" onClick={this.openComparisonModal}>Show Comparison</a>) : null}

                    {this.state.showComparisonModal ? (<ComparisonModal show={this.state.showComparisonModal} saveComparison={this.saveComparisonHandler} closeModal={this.openComparisonModal} productsToCompare={this.state.productsAddedForComparison} isAuthenticated={this.props.isAuthenticated} />) : null}

                </div>

            </div>

        );

    }

}

const mapStateToProps = (state) => ({

    brands: state.productReducer.brands,

    showLoader: state.productReducer.showLoader,

    isAuthenticated: state.authReducer.isAuthenticated,

    user: state.authReducer.user,

})

const mapDispatchToProps = (dispatch) => {

    return {

        getProducts: (searchTerm) => dispatch(getProducts(searchTerm)),

        saveComparison: (postData) => dispatch(saveComparison(postData)),

    }

}

export default connect(mapStateToProps, mapDispatchToProps)(Home);

1. **Components/ComparisonModal.js**

import React, { useRef } from 'react';

const ComparisonModal = ({ show, closeModal, productsToCompare, isAuthenticated, saveComparison }) => {

    console.log(productsToCompare[0]);

    const modalRef = useRef();

    window.onclick = function (event) {

        if (event.target == modalRef.current) {

            closeModal()

        }

    }

    return (

        <div id="myModal" className={show ? "modal show" : "modal"} ref={modalRef}>

            <div className="modal-content">

                <span className="close" onClick={closeModal}>&times;</span>

                <div className="comparison-content">

                    <div className="prod-info-row">

                        <div className="index info-label">

                        </div>

                        <div className="prod-image info-value">

                            <img src={productsToCompare[0].image} alt={productsToCompare[0].id} />

                        </div>

                        <div className="prod-image info-value">

                            <img src={productsToCompare[1].image} alt={productsToCompare[1].id} />

                        </div>

                    </div>

                    <div className="prod-info-row">

                        <div className="info-label">

                            Brand

                        </div>

                        <div className="info-value">

                            {productsToCompare[0].brand}

                        </div>

                        <div className="info-value">

                            {productsToCompare[1].brand}

                        </div>

                    </div>

                    <div className="prod-info-row">

                        <div className="info-label">

                            Title

                        </div>

                        <div className="info-value">

                            {productsToCompare[0].name}

                        </div>

                        <div className="info-value">

                            {productsToCompare[1].name}

                        </div>

                    </div>

                    <div className="prod-info-row">

                        <div className="info-label">

                            Price

                        </div>

                        <div className="info-value">

                            {productsToCompare[0].price}

                        </div>

                        <div className="info-value">

                            {productsToCompare[1].price}

                        </div>

                    </div>

                    <div className="prod-info-row">

                        <div className="info-label">

                            Rating

                        </div>

                        <div className="info-value">

                            {productsToCompare[0].rating ? productsToCompare[0].rating : "Not Available"}

                        </div>

                        <div className="info-value">

                            {productsToCompare[1].rating ? productsToCompare[1].rating : "Not Available"}

                        </div>

                    </div>

                    <div className="prod-info-row">

                        <div className="info-label">

                            Offers

                        </div>

                        <div className="info-value">

                            {productsToCompare[0].offers ? productsToCompare[0].offers : "No Offers"}

                        </div>

                        <div className="info-value">

                            {productsToCompare[1].offers ? productsToCompare[1].offers : "No Offers"}

                        </div>

                    </div>

                    <div className="prod-info-row">

                        <div className="info-label">

                        </div>

                        <div className="info-value">

                            <a href={productsToCompare[0].link} class="btn btn-primary">Buy now</a>

                        </div>

                        <div className="info-value">

                            <a href={productsToCompare[1].link} class="btn btn-primary">Buy now</a>

                        </div>

                    </div>

                    {isAuthenticated ? (<div className="save-btn-row">

                        <div className="save-btn">

                            <a href="#" class="btn btn-primary" onClick={saveComparison}>Save Comparison</a>

                        </div>

                    </div>) : null}

                </div>

            </div>

        </div>

    )

}

export default ComparisonModal;

1. **Components/Products.js**

import React from 'react';

const Products = ({ brands, addForComparison, selectedProductIds }) => {

    return (

        brands.map(brand => (

            <div className="result-container">

                <h3 className="brand-title">{brand.brand}</h3>

                <div className="products" key={brand.id}>

                    {brand.products.map(product => {

                        const selectedClass = selectedProductIds.indexOf(product.id) >= 0 ? ' selected' : ''

                        return (<div className={`card${selectedClass}`} key={product.id} onClick={() => addForComparison(product)}>

                            <img className="card-img-top" src={product.image} alt="Card image cap" />

                            <div className="card-body">

                                <h5 className="card-title">{product.name.substr(0, 40)}</h5>

                                <p className="card-price"> {product.price}</p>

                                <a href={product.link} class="btn btn-primary">Buy now</a>

                            </div>

                        </div>)

                    })}

                </div>

            </div>

        ))

    )

}

export default Products;

1. **Components/Offers.js**

import React, { useEffect, useState } from 'react';

import { connect } from 'react-redux';

import { getOffers } from './../store/actions/offerActions';

class Offers extends React.Component {

    constructor(props) {

        super(props);

    }

    componentDidMount() {

        this.props.getOffers();

    }

    componentDidUpdate() {

        console.log(this.props.offers);

    }

    render() {

        const allBrandOffers = this.props.offers;

        return (

            <div className="offers" >

                <h1 className="section-title">Offers</h1>

                {allBrandOffers.map(brandOffer => {

                    const offers = brandOffer.offers;

                    return (

                        <>

                            <h3 className="brand-title">{brandOffer.brand}</h3>

                            <div className="brand-offers">

                                {offers.map(offer => {

                                    return (<div className="offer-card" key={offer.id}>

                                        <img className="offer-card-img-top" src={offer.imageLink} alt="Card image cap" />

                                        <div className="offer-card-body">

                                            <h5 className="offer-card-title">{offer.title.substr(0, 40)}</h5>

                                            <div className="offer-card-price">

                                                <p class="normalPrice">{offer.reducedPrice}</p>

                                                <p className="offer"><span className="strikethrough">{offer.originalPrice}</span>{offer.offerValue}</p>

                                            </div>

                                            <a href={offer.link} class="btn btn-deal btn-primary">Check Deal</a>

                                        </div>

                                    </div>)

                                }

                                )}

                            </div>

                        </>

                    )

                })}

            </div>

        )

    }

}

const mapStateToProps = (state) => ({

    offers: state.offerReducer.offers

})

const mapDispatchToProps = (dispatch) => {

    return {

        getOffers: () => dispatch(getOffers())

    }

}

export default connect(mapStateToProps, mapDispatchToProps)(Offers);

1. **Store**

import \* as ACTIONS from "./types";

import axios from "axios";

export const getComparisons = (userId) => {

    console.log("In Action Account")

    return dispatch => {

        axios.post("http://localhost:5000/api/user/comparisons", {

            userId: userId

        }).then(response => {

            console.log(response);

            dispatch({

                type: ACTIONS.GET\_COMPARISONS,

                payload: response['data']['data']

            })

            // return response;

        }).catch(err => {

            console.log(err);

        })

    }

}

import \* as ACTIONS from "./types";

import axios from "axios";

import setAuthToken from "./../../utils/setAuthToken";

import jwt\_decode from "jwt-decode";

// Register user action

export const registerUser = (data, history) => (dispatch) => {

  axios

    .post("http://localhost:5000/api/user/register", data)

    .then((response) => {

      console.log(response);

      history.push("/login");

    })

    .catch((err) => {

      console.log(err);

      dispatch({

        type: ACTIONS.SET\_ERRORS,

        payload: err.response.data,

      });

    });

};

// Login User action

export const loginUser = (data) => (dispatch) => {

  axios

    .post("http://localhost:5000/api/user/login", data)

    .then((res) => {

      const { token } = res.data;

      localStorage.setItem("jwtToken", token);

      // Set token to Auth header

      setAuthToken(token);

      // Decode token to get user data

      const decoded = jwt\_decode(token);

      // Set current user

      dispatch(setCurrentUser(decoded));

    })

    .catch((err) => {

      console.log(err.response.data, "Error in action");

      dispatch({

        type: ACTIONS.SET\_ERRORS,

        payload: err.response.data,

      });

    });

};

// Set logged in user

export const setCurrentUser = (decoded) => {

  return {

    type: ACTIONS.SET\_CURRENT\_USER,

    payload: decoded,

  };

};

export const resetUser = () => {

  return {

    type: ACTIONS.RESET\_USER,

  }

}

// User loading

export const setUserLoading = () => {

  return {

    type: ACTIONS.USER\_LOADING,

  };

};

// Log user out

export const logoutUser = (history) => (dispatch) => {

  // Remove token from local storage

  console.log("In Actions logout");

  localStorage.removeItem("jwtToken");

  // Remove auth header for future requests

  setAuthToken(false);

  // Set current user to empty object {} which will set isAuthenticated to false

  dispatch(resetUser());

  history.push("/login");

};

import \* as ACTIONS from "./types";

import axios from "axios";

export const getOffers = () => {

    console.log("In Actions");

    return (dispatch) => {

        dispatch({

            type: ACTIONS.SHOW\_LOADER

        })

        axios.get('http://localhost:5000/api/scrapper/offers').then(response => {

            const offersData = response.data['data'];

            console.log(offersData, "in Actions");

            dispatch({

                type: ACTIONS.GET\_OFFERS,

                payload: offersData

            })

            dispatch({

                type: ACTIONS.HIDE\_LOADER

            })

        }).catch(err => console.log(err))

    }

}

import \* as ACTIONS from "./types";

import axios from "axios";

export const getProducts = (searchTerm) => {

    console.log(searchTerm, "In Actions");

    return (dispatch) => {

        dispatch({

            type: ACTIONS.SHOW\_LOADER

        })

        axios.post('http://localhost:5000/api/scrapper',

            { query: searchTerm }).then(response => {

                const brandsData = response.data['data'];

                console.log(brandsData);

                dispatch({

                    type: ACTIONS.GET\_PRODUCTS,

                    payload: brandsData

                })

                dispatch({

                    type: ACTIONS.HIDE\_LOADER

                })

            }).catch(err => console.log(err))

    }

}

export const saveComparison = (postData) => {

    console.log(postData);

    return dispatch => {

        dispatch({

            type: ACTIONS.SHOW\_LOADER

        })

        return axios.post('http://localhost:5000/api/user/save',

            postData).then(response => {

                console.log(response, "This is in actions");

                dispatch({

                    type: ACTIONS.HIDE\_LOADER

                })

                return response;

            }).catch(err => console.log(err))

    }

}

import \* as ACTIONS from "./../actions/types";

const initialState = {

    comparisons: [],

};

const accountReducer = (state = initialState, action) => {

    switch (action.type) {

        case ACTIONS.GET\_COMPARISONS:

            return {

                ...state,

                comparisons: action.payload

            }

        default:

            return state;

    }

}

export default accountReducer;

import \* as ACTIONS from "./../actions/types";

const isEmpty = require("is-empty");

const initialState = {

  isAuthenticated: false,

  user: null,

};

const authReducer = (state = initialState, action) => {

  switch (action.type) {

    case ACTIONS.IS\_AUTHENTICATED:

      return {

        ...state,

        isAuthenticated: true,

      };

    case ACTIONS.SET\_CURRENT\_USER:

      return {

        ...state,

        isAuthenticated: !isEmpty(action.payload),

        user: action.payload,

      };

    case ACTIONS.RESET\_USER:

      return {

        ...state,

        isAuthenticated: false,

        user: null,

      };

    default:

      return state;

  }

};

export default authReducer;

import \* as ACTIONS from "./../actions/types";

const initialState = {

  errors: null,

};

const errorReducer = (state = initialState, action) => {

  switch (action.type) {

    case ACTIONS.SET\_ERRORS:

      console.log(action, "in reducer");

      return {

        ...state,

        errors: action.payload,

      };

    default:

      return state;

  }

};

export default errorReducer;

import \* as ACTIONS from "./../actions/types";

const isEmpty = require("is-empty");

const initialState = {

    offers: [],

    showLoader: false

};

const productReducer = (state = initialState, action) => {

    switch (action.type) {

        case ACTIONS.GET\_OFFERS:

            return {

                ...state,

                offers: action.payload

            }

        case ACTIONS.SHOW\_LOADER:

            return {

                ...state,

                showLoader: true

            }

        case ACTIONS.HIDE\_LOADER:

            return {

                ...state,

                showLoader: false

            }

        default:

            return state;

    }

}

export default productReducer;

import \* as ACTIONS from "./../actions/types";

const isEmpty = require("is-empty");

const initialState = {

    brands: [],

    showLoader: false,

    comparisons: []

};

const productReducer = (state = initialState, action) => {

    switch (action.type) {

        case ACTIONS.GET\_PRODUCTS:

            return {

                ...state,

                brands: action.payload

            }

        case ACTIONS.SAVE\_COMPARISON:

            return {

                ...state,

                comparisons: [action.payload, ...state.comparisons]

            }

        case ACTIONS.SHOW\_LOADER:

            return {

                ...state,

                showLoader: true

            }

        case ACTIONS.HIDE\_LOADER:

            return {

                ...state,

                showLoader: false

            }

        default:

            return state;

    }

}

export default productReducer;