Contents

[FIXING THE REDIREFCT TO THE FRONT PAGE 1](#_Toc78295227)

[USING updateObject() IN THE ENTIRE APP 2](#_Toc78295228)

[SHARING THE VALIDATION METHOD 2](#_Toc78295229)

[USING ENVIRONMENT VARIABLES 3](#_Toc78295230)

[REMOVING CONSOLE.LOG() 3](#_Toc78295231)

[ADDING LAZY LOADING 3](#_Toc78295232)

# FIXING THE REDIREFCT TO THE FRONT PAGE

When we order something without login, it will redirect to login page. After logged in, it should redirect to checkout page but it will redirect to home page. ### we need to fix this error####

Auth.js [container folder]

//redirect to

        let authRedirect = null;

        if(this.props.isAuthenticated){

            authRedirect =<Redirect to={this.props.authRedirectPath} />

        }

App.js

if(props.isAuthenticated){

    routes=(

      <Switch>

       <Route path="/checkout" component={Checkout} />

       <Route path="/orders" component={Orders} />

       <Route path="/logout" component={Logout} />

       <Route path="/auth" component={Auth} />

       <Route path="/" exact component={PizzaBuilder} />

      <Redirect to="/" />

      </Switch>

    );

  }

When we click side draw link, it wont close. So we fix this

SideDrawer.js

import React from 'react';

import './SideDrawer.css'

import Logo from '../../Logo/Logo';

import NavigationItems from '../NavigationItems/NavigationItems';

import Backdrop from '../../UI/Backdrop/Backdrop';

import Aux from '../../../hoc/Auxilliary/Auxilliary';

const SideDrawer = (props) => {

    let attachedClasses = ['SideDrawer','Close'];

    if(props.open) {

        attachedClasses = ['SideDrawer','Open'];

    }

    return (

        <Aux>

        <Backdrop show={props.open} clicked={props.closed} />

        <div className={attachedClasses.join(' ')} onClick={props.closed}>

            <Logo />

             <nav>

                <NavigationItems isAuthenticated={props.isAuth}/>

            </nav>

        </div>

        </Aux>

        );

}

export default SideDrawer;

# USING updateObject() IN THE ENTIRE APP

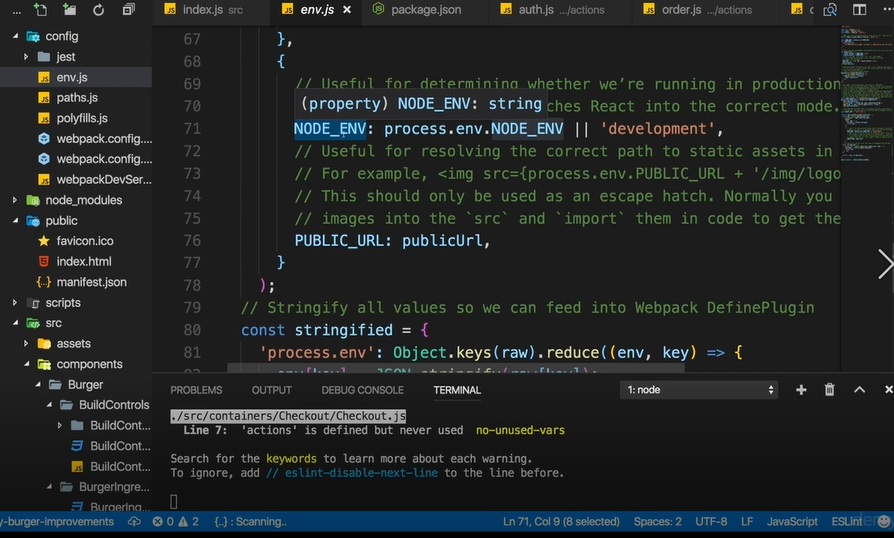
Use updateObject() in all actions files i.e. contactData.

# SHARING THE VALIDATION METHOD

We have used checkValidity() in both auth and contactData. So we can make new file to keep validity function to access by any functions.

# USING ENVIRONMENT VARIABLES

Make harder for others to look at your react states.



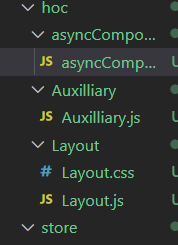
Index.js

const composeEnhancers = process.env.NODE\_ENV === 'development' ? window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_COMPOSE\_\_ : null || compose;

# REMOVING CONSOLE.LOG()

# ADDING LAZY LOADING [Downloading bundles after only redirecting]

Create file named: AysncComponent.js



aysncComponent.js

import React, { Component } from 'react';

const asyncComponent = (importComponent) => {

    return class extends Component {

        state= {

            component: null

        }

        componentDidMount () {

            importComponent()

            .then(cmp => {

                this.setState({component: cmp.default});

            });

        }

        render () {

            const C = this.state.component;

            return C ? <C {...this.props} /> : null;

        }

    }

}

export default asyncComponent;

App.js

import React,{useEffect} from 'react';

import Layout from '../hoc/Layout/Layout';

import PizzaBuilder  from './PizzaBuilder/PizzaBuilder';

// import Checkout from './Checkout/Checkout';

// import Orders from './Orders/Orders';

// import Auth from './Auth/Auth';

import Logout from './Auth/logout/logout';

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

import {connect } from 'react-redux';

import \* as actions from '../store/actions/index'

import asyncComponent from '../hoc/asyncComponent/asyncComponent';

const aysncCheckout =  asyncComponent( () => {

  return import('./Checkout/Checkout');

});

const aysncOrders =  asyncComponent( () => {

  return import('./Orders/Orders');

});

const aysncAuth =  asyncComponent( () => {

  return import('./Auth/Auth');

});

const App = (props) => {

  useEffect(() => {

    props.onTryAutoSignup();

  })

  let routes =(

    //JSX

    <Switch>

    <Route path="/auth" component={aysncAuth} />

    <Route path="/" exact component={PizzaBuilder} />

    <Redirect to="/" />

    </Switch>

  );

  if(props.isAuthenticated){

    routes=(

      <Switch>

       <Route path="/checkout" component={aysncCheckout} />

       <Route path="/orders" component={aysncOrders} />

       <Route path="/logout" component={Logout} />

       <Route path="/auth" component={aysncAuth} />

       <Route path="/" exact component={PizzaBuilder} />

      <Redirect to="/" />

      </Switch>

    );

  }

  return(

    <div className="App">

      <Layout>

          {routes}

      </Layout>

    </div>

  )

}

//APP.JS is root component which will loaded everytime. not matter which route we vist

//this makes great app to check authentication

const mapStateToProps = state =>{

  return{

    isAuthenticated: state.auth.token !=null

  };

};

const mapDispatchToProps =  dispatch => {

  return{

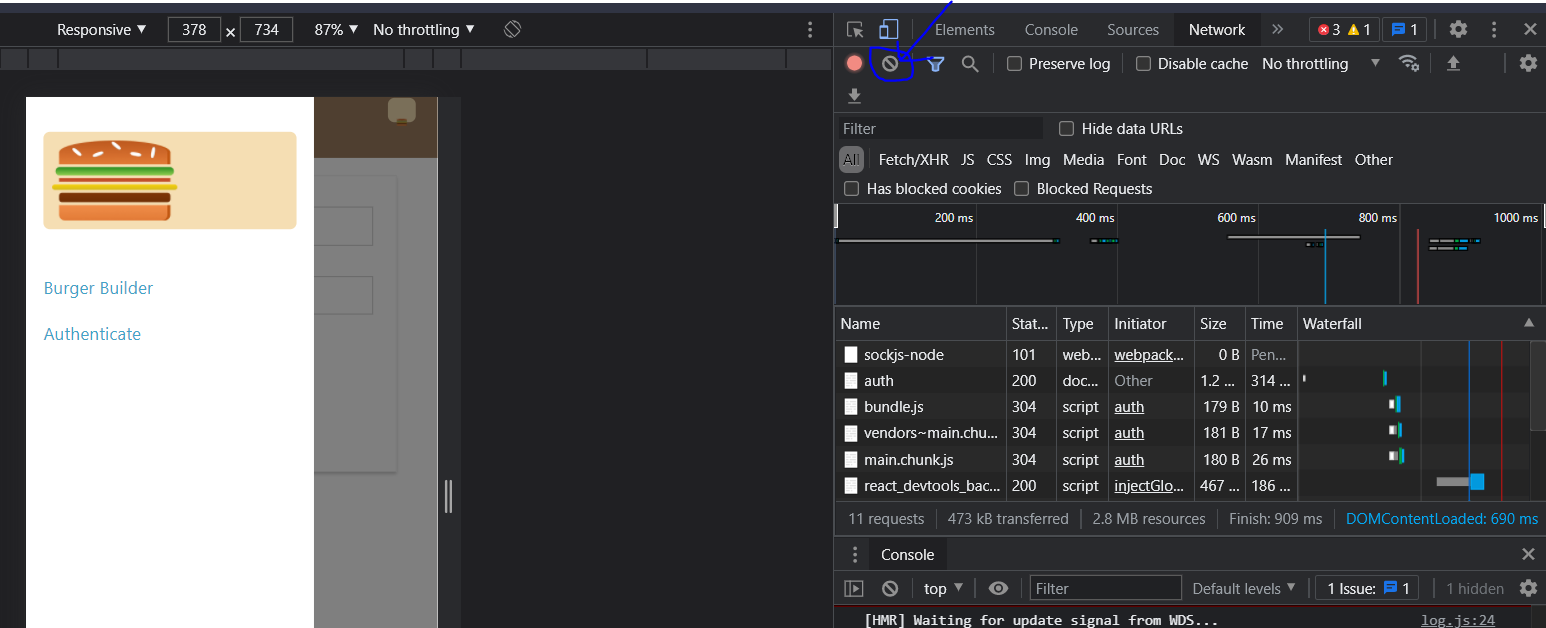
    onTryAutoSignup: () => dispatch(actions.authCheckState())

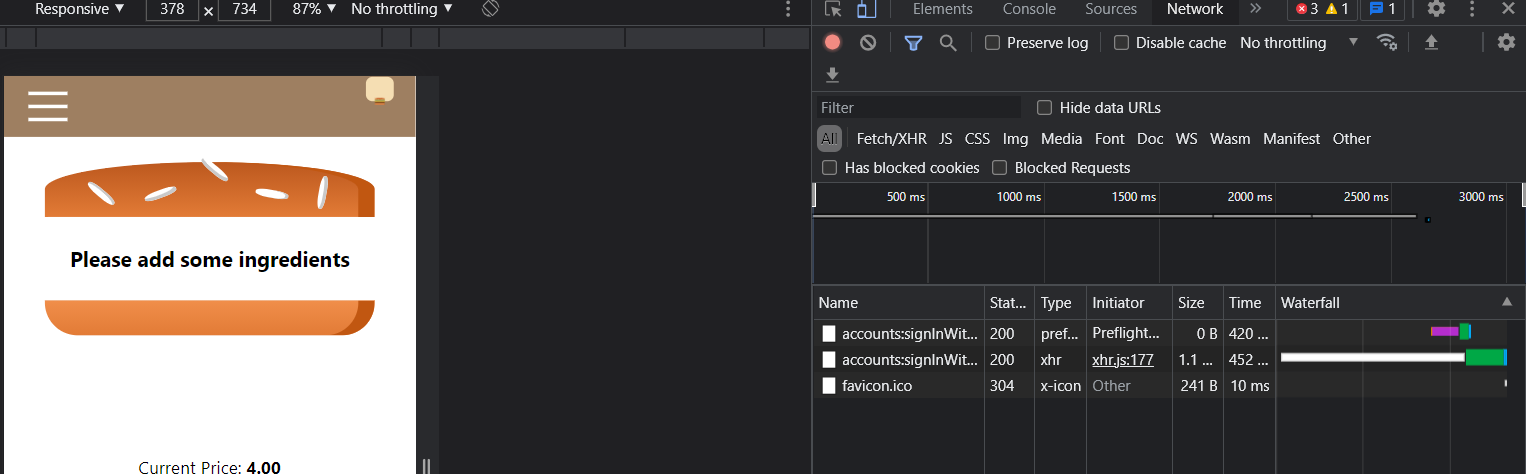
  };

};

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

CONSOLE:





After redirecting, it will download orders page bundle

