039 Redux Store Chrome Extension Setup

yarn install redux react-redux redux-thunk

ابتدا reducer/index,reducers/authReducer.js را میسازیم در authReducer

**const** initialState = {  
 isAuthenticated: **false**,  
 user: {}  
};  
**export default function** (state = initialState, action) {  
 **switch** (action.type) {  
 **default**:  
 **return** state;  
 }  
}

و در index.js

**import** {combineReducers} **from** 'redux'  
  
**import** authReducer **from** './authReducer'  
  
**export default** combineReducers({  
 auth: authReducer  
})

حال در root اصلی فایل store.js را میسازیم داریم :

**import** {createStore, applyMiddleware, compose} **from** 'redux'  
**import** thunk **from** 'redux-thunk'  
**import** rootReducer **from** './reducers/index'  
  
**const** middleware = [thunk];  
  
**const** initialState = {};  
  
**const** store = createStore(  
 rootReducer,  
 initialState,  
 compose(applyMiddleware(...middleware),  
 window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_ && window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_()//برای redux dev tool  
 ));  
  
**export default** store;

سپس در App.js

**import** {Provider} **from** "react-redux";  
**import** store **from** "./store";

<Provider store={store}>  
 <BrowserRouter>  
 ....   
 </BrowserRouter>  
</Provider>

040 Redux Action Reducer Workflow Example

ابتدا actions/authAction.js را میسازیم داخل آن داریم

**export const** TEST\_DISPATCH = 'TEST\_DISPATCH';  
  
**export const** registerUser = (userData) => {  
 **return** {  
 type: TEST\_DISPATCH,  
 payload: userData  
 }  
};

حال از این در authReducer.js استفاده می کنیم

**import** {TEST\_DISPATCH} **from** '../actions/authActions'

**switch** (action.type) {  
 **case** TEST\_DISPATCH:  
 **return** {  
 ...state,  
 user: action.payload  
 };

حال داده و اکشن را در component/register.js

**import** {registerUser} **from** '../../actions/authActions'  
**import** {connect} **from** 'react-redux'

**const** mapStateToProps = (state) => ({  
 auth: state.auth  
});  
  
**export default** connect(mapStateToProps, {registerUser})(Register);

حال با this.props به اطلاعات دسترسی داریم

onSubmit(e) {  
......   
  
 **this**.props.registerUser(newUser);  
}

یا

render() {  
 **const** {errors} = **this**.state;  
  
 **const** {user} = **this**.props.auth;

....

حال می توانیم از prop\_types برای نوع داده استفاده کنیم برای این کار در components/register.js

**import** PropTypes **from** 'prop-types'

حال قبل از mapStateToProps

Register.propTypes = {  
 registerUser: PropTypes.func.isRequired,  
 auth: PropTypes.object.isRequired  
};

041 Registration The Error Reducer

حال میخاهیم فرایند ثبت نام در actions/authAction.js انجام شود :

**export const** GET\_ERRORS = 'GET\_ERRORS';  
**export const** registerUser = (userData) => dispatch => {  
 axios.post('/api/users/register', userData)  
 .then(res => console.log(res.data))  
 .catch(err =>  
 dispatch({  
 type: GET\_ERRORS,  
 payload: err.response.data  
 }));  
};

حال فایل reducers/errorReducer.js را میسازیم

**import** {GET\_ERRORS} **from** '../actions/authActions'  
  
**const** initialState = {};  
**export default function** (state = initialState, action) {  
 **switch** (action.type) {  
 **case** GET\_ERRORS:  
 **return** action.payload;  
 **default**:  
 **return** state;  
 }  
}

حال در reducers/index.js

**import** errorReducer **from** "./errorReducer";  
  
**export default** combineReducers({  
 auth: authReducer,  
 errors: errorReducer  
})

حال در componets/Register.js در متد mapPropsToState

**const** mapStateToProps = (state) => ({  
 auth: state.auth,  
 errors: state.errors  
});

حال در متد componentReciveProps از این errors استفاده میکنیم

componentWillReceiveProps(nextProps) {  
 **if** (nextProps.errors) {  
 **this**.setState({  
 errors: nextProps.errors

})}}

در آخر میخاهیم وقتی کاربر درست ثبت نام کرد میخاهیم به صفحه login برود

ابتدا در authAction.js

**export const** registerUser = (userData, history) => dispatch => {  
 axios.post('/api/users/register', userData)  
 .then(res => history.push('/login'))  
 ….   
};

از history استفاده می کنیم. سپس در components/Register.js

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

**export default** connect(mapStoreToProps, {registerUser})(withRouter(Register));

onSubmit(e) {  
 e.preventDefault();  
  
 **const** newUser = {  
 ……..  
 **this**.props.registerUser(newUser, **this**.props.history);

اینجا هم از this.props.history استفاده می کنیم.

042 Redux Login Action Set Current User