React Router Redux demo app

This is a simple app to demonstrate the basics to develop a production ready React app using Router and Redux. This app has two functionalities, add a student to the student registration system and view all added students. the app UI needs a bit of styling, so I use [**material-ui**](https://material-ui.com/)**package**to achieve that.

**1. Environment set up**

*I assume you have installed node.js and ready to execute npm commands.*

Firstly, create react app folder structure and required files using the [***create-react-app***](https://github.com/facebook/create-react-app) command. Then we will install other dependencies ***Router, Redux and material-ui***

npm create-react-app react-router-redux-demo  
cd react-router-redux-demo  
npm install --save react-router-dom   
npm install --save redux  
npm install --save react-redux  
npm install @material-ui/core

What is react-router?

React Router is the standard routing library for React. It loads code/components based on the specified URL path. As you know, React app use only one HTML file to render components. React Router dynamically renders components on this HTML file according to the URL path. More importantly, this feature allows us to add multipage behaviour to a react app.

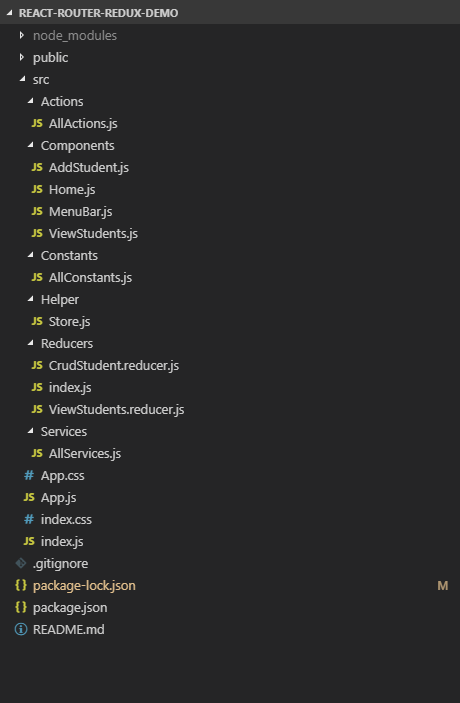
What is Redux

Redux makes it easy to manage the state of JavaScript applications. In other words, Redux helps us to manage the data model of our application and reflect these changes on the views.

**2. Folder Structure**

We will use the following folder structure for the app.

https://miro.medium.com/max/40/1*InbZMOSNMevMMOSW0RQnKA.png?q=20



Actions: Call service methods and update the state of our application

Components: React components and Views

Constants: Common Constants definitions to be used in the application

Helper: Supporting items

Reducers: Model definitions to manage the state of the application

Services: Execute operations such as accessing external resources

**3. Add Redux to the app**

We need to understand three basics of Redux before applying it to React.

* Store

Redux store holds the application state. The state of the application can be accessed by ***getState()***method. Also, React components can register their listeners to the Redux store using **connect()** which comes with **react-redux**package.

Here is our store for this application.

* Actions

Actions are payloads of information that send data from the application to the Redux store. Action payload contains action type and data. Following is an example of Redux action, ADD\_STUDENT\_REQUEST is the type of this action and it carries a student as its data.

{   
 type: ADD\_STUDENT\_REQUEST,  
 student: {  
 "firstName":"Rafael",  
 "lastName":"Nadal",  
 "country":"Spain",  
 "course":"React"  
 }  
}

An action is sent to the store using **dispatch(action)**action. AllAction.js defines different actions to update the application state.

* Reducers

Reducers specify how the application’s state changes in response to actions sent to the store. The reducer is a pure function that takes the previous state and action then returns the next state. Here is our reducer to add a new student. You can see crudStudent() returns different states for different action types.

**4. Services**

This is the layer that interacts with external resources. A service method may call an external API to fetch data, read [local storage](https://www.w3schools.com/html/html5_webstorage.asp) or execute simple client-side logic. In our case, the service methods read and write students to the [local storage](https://www.w3schools.com/html/html5_webstorage.asp).

**5. Components**

Firstly, We will create *MenuBar.js* component, which is used to navigate to different views(pages) of the application. This is a common component of this application. We will use <AppBar> component from **material-ui** to add a little bit of styling to this component.

Now, we can create *Home.js* component. As you can see, it uses *MenuBar.js*as its dependent component.

Similarly, we will create the other two components [*AddStudent*](https://github.com/coderSinol/React-router-redux-demo/blob/master/src/Components/AddStudent.js)*.js* and [*ViewStudents*](https://github.com/coderSinol/React-router-redux-demo/blob/master/src/Components/ViewStudents.js)*.js*

**6. React Route**

Our purpose is creating a multipage application using React. This can be achieved using React-Router. There is a couple of options to do this,

* BrowserRouter: [https://mydomain.com/h](https://application.com/dashboard)ome
* HashRouter: [https://mydomain.com/#h](https://application.com/dashboard)ome

We will use BrowserRouter for our application as it provides much-needed features in the production such as SEO friendly URLs and shareable links.

All route definitions are in the App.js component. This is the root component of the application and it renders other components based on the URL path.

**7. Test it**

Now app is ready to test. Run **npm start** to run the app