127-Setting up React-Meteor-Data

- install a react addon, go to the terminal-this add on works behind the scenes, and here we are using the same version as the meteor version

```
joses-MacBook-Pro:notes mendoza$ meteor npm install --save react-addons-pure-render-mixin@15.4.1
```

- next install an atmosphere package - which we will be working with directly

```
joses-MacBook-Pro:notes mendoza$ meteor add react-meteor-data@=0.2.9
```

- **over to** PrivateHeader.js lets import the library we just added

```
import React from 'react';
import { Accounts } from 'meteor/accounts-base';
import { createContainer } from 'meteor/react-meteor-data';

const PrivateHeader = (props)=>{
    return (
        <div className="header">
            <div className="header__content">
                <h1 className="header__title">{props.title}</h1>
                <button className="button button--link-text " onClick={()=>
props.handleLogout()}>Logout</button>
            </div>
        </div>
    );
}

PrivateHeader.propTypes = {
    title: React.PropTypes.string.isRequired,
    handleLogout: React.PropTypes.func.isRequired
};
```

```
export default PrivateHeader;
```

- now lets createContainer down below, we are no longer going to export PrivateHeader as our default export, instead is going to be a named export

```
import React from 'react';

import { Accounts } from 'meteor/accounts-base';

import { createContainer } from 'meteor/react-meteor-data';


export const PrivateHeader = (props)=>{

    return (

        <div className="header">

            <div className="header__content">

                <h1 className="header__title">{props.title}</h1>

                <button className="button button--link-text " onClick={()=>
props.handleLogout()}>Logout</button>

            </div>

        </div>

    );

}


PrivateHeader.propTypes = {

    title: React.PropTypes.string.isRequired,

    handleLogout: React.PropTypes.func.isRequired

};


// export default PrivateHeader;
```

- **now lets createContainer() it takes two arguments, first is a function, this is like Tracker.autorun call, everything that PrivateHeader needs is going to be difined inside this function, which in this is case is handle.logout, the second argument is the actual component that you actually want to render to the screen**

```
PrivateHeader.propTypes = {

    title: React.PropTypes.string.isRequired,
    handleLogout: React.PropTypes.func.isRequired

};


createContainer(()=>{



}, PrivateHeader);

// export default PrivateHeader;
```

- now lets export createContainer so it can be used

```
export default createContainer(()=>{



}, PrivateHeader);
```

- next in our function we are going to return an object, we are going to return handleLogout

```
export default createContainer(()=>{

    return {

         handleLogout: () => Accounts.logout()

    }

}, PrivateHeader);
```

- now lets go to PrivateHeader.test.js, lets change the import PrivatedHeader

```
import { Meteor } from 'meteor/meteor';

import React from 'react';

import expect from 'expect';

import { mount } from 'enzyme';


import { PrivateHeader } from './PrivateHeader';


if(Meteor.isClient) {

    describe('PrivateHeader', function(){

        it('should set button text to logout', function(){

           const wrapper = mount(<PrivateHeader title="test title" handleLogout={()=>{}}/>);
```

```
          const buttonText = wrapper.find('button').text();


          expect(buttonText).toBe('Logout');
       });
```

- over in our terminal let quit out of test mode and run meteor to run our app
- login to an existing account and logout, it should take you back to the login screen