Weather App in React

51. App Overview and Planning

52. Component Setup

53. Controlled Components and Binding Context

54. Form Elements in React

55. Working with API's

56. Introduction to Middleware

57. Ajax Requests with Axios

58. Redux-Promise in Practice

59. Redux-Promise Continued

60. Avoiding State Mutations in Reducers

61. Building a List Container

62. Mapping Props to a Render Helper

63. Quick Note

64. Adding Sparkline Charts

65. Making a Reusable Chart Component

66. Labeling of Units

67. Google Maps Integration

68. Google Maps Integration Continued

69. Project Review

App Wireframe

**Component Setup**

Make a container component for search bar and weather list because both component will access the redux state so first we will add a directory named containers and search bar and weather list component will be created in containers directory.

Search Bar Component (containers/search\_bar.js)

Search Bar Component(containers/search\_bar.js)

import React,{Component} from 'react';

class SearchBar extends Component{

render(){

return(

<div>

<h6> This is Search bar component </h6>

</div>

)

}

}

export default SearchBar;

Weather List Component(containers/weather\_list.js)

import React,{Component} from 'react';

class WeatherList extends Component{

render(){

return(

<div>

<h6> This is Weather List component </h6>

</div>

)

}

}

export default WeatherList;

**Controlled Components and Binding Context**

A Controlled Component is one that takes its current value through props and notifies changes through callbacks like onChange. A parent component "controls" it by handling the callback and managing its own state and passing the new values as props to the controlled component. You could also call this a "dumb component".

import React,{Component} from 'react';

class SearchBar extends Component{

constructor(props){

super(props);

this.state={

term:''

}

this.SerchChange=this.SerchChange.bind(this);

}

SerchChange(event){

this.setState({

term:event.target.value

})

}

render(){

return(

<div>

<form className="input-group">

<input onChange={this.SerchChange} className="form-control" />

<span className='input-group-btn'>

<button type="submit" className="btn btn-primary">Search</button>

</span>

</form>

<p>{this.state.term}</p>

</div>

)

}

}

export default SearchBar;