React Assignment

1)Calculator

```
//
App.js
        import React, { Component } from "react";
        import "./App.css";
        class Calculator extends Component {
          constructor(props) {
            super(props);
            this.state = {
              value: 0,
              displayValue: "0",
              waitingForOperand: false,
              operator: null
            };
          }
          inputDigit(digit) {
            const { displayValue, waitingForOperand } = this.state;
            if (waitingForOperand) {
              this.setState({
                displayValue: String(digit),
                waitingForOperand: false
              });
            } else {
              this.setState({
                displayValue:
                  displayValue === "0" ? String(digit) : displayValue + digit
              });
            }
          }
          inputDot() {
            const { displayValue, waitingForOperand } = this.state;
            if (waitingForOperand) {
              this.setState({
                displayValue: ".",
                waitingForOperand: false
              });
            } else if (displayValue.indexOf(".") === -1) {
              this.setState({
                displayValue: displayValue + ".",
                waitingForOperand: false
              });
            }
          }
          clearDisplay() {
            this.setState({
```

```
    displayValue: "0"
  });
}
toggleSign() {
  const { displayValue } = this.state;
  this.setState({
    displayValue:
      displayValue.charAt(0) === "-"
        ? displayValue.substr(1)
        : "-" + displayValue
  });
}
inputPercent() {
  const { displayValue } = this.state;
  const value = parseFloat(displayValue);
  this.setState({
    displayValue: String(value / 100)
  });
}
performOperation(nextOperator) {
  const { value, displayValue, operator } = this.state;
  const nextValue = parseFloat(displayValue);
  const operations = {
    "/": (prevValue, nextValue) => prevValue / nextValue,
    "*": (prevValue, nextValue) => prevValue * nextValue,
    "+": (prevValue, nextValue) => prevValue + nextValue,
    "-": (prevValue, nextValue) => prevValue - nextValue,
    "=": (prevValue, nextValue) => nextValue
  };
  if (value == null) {
    this.setState({
      value: nextValue
    });
  } else if (operator) {
    const currentValue = value || 0;
    const computedValue = operations[operator](currentValue, nextValue);
    this.setState({
      value: computedValue,
      displayValue: String(computedValue)
    });
  }
  this.setState({
    waitingForOperand: true,
    operator: nextOperator
  });
}
```

```
  render() {
    const { displayValue } = this.state;
    return (
      <div className="calculator">
        <div className="calculator-display">{displayValue}</div>
        <div className="calculator-keypad">
          <div className="input-keys">
            <div className="function-keys">
              <button className="calculator-key key-clear" onClick={() =>
this.clearDisplay()}>
                AC
              </button>
              <button className="calculator-key key-sign" onClick={() => this.toggleSign()}>
                ±
              </button>
              <button className="calculator-key key-percent" onClick={() =>
this.inputPercent()}>
                %
              </button>
            </div>
            <div className="digit-keys">
              <button className="calculator-key key-0" onClick={() => this.inputDigit(0)}>
                0
              </button>
              <button className="calculator-key key-dot" onClick={() => this.inputDot()}>
                ●
              </button>
              <button className="calculator-key key-1" onClick={() => this.inputDigit(1)}>
                1
              </button>
              <button className="calculator-key key-2" onClick={() => this.inputDigit(2)}>
                2
              </button>
              <button className="calculator-key key-3" onClick={() => this.inputDigit(3)}>
                3
              </button>
              <button className="calculator-key key-4" onClick={() => this.inputDigit(4)}>
                4
              </button>
              <button className="calculator-key key-5" onClick={() => this.inputDigit(5)}>
                5
              </button>
              <button className="calculator-key key-6" onClick={() => this.inputDigit(6)}>
                6
              </button>
              <button className="calculator-key key-7" onClick={() => this.inputDigit(7)}>
```

```
                7
              </button>
              <button className="calculator-key key-8" onClick={() => this.inputDigit(8)}>
                8
              </button>
              <button className="calculator-key key-9" onClick={() => this.inputDigit(9)}>
                9
              </button>
            </div>
          </div>
          <div className="operator-keys">
            <button className="calculator-key key-divide" onClick={() =>
this.performOperation("/")}>
              ÷
            </button>
            <button className="calculator-key key-multiply" onClick={() =>
this.performOperation("*")}>
              ×
            </button>
            <button className="calculator-key key-subtract" onClick={() =>
this.performOperation("-")}>
              −
            </button>
            <button className="calculator-key key-add" onClick={() =>
this.performOperation("+")}>
              +
            </button>
            <button className="calculator-key key-equals" onClick={() =>
this.performOperation("=")}>
              =
            </button>
          </div>
        </div>
      </div>
    );
  }
}
export default Calculator;
// App.css
.calculator {
  font-family: sans-serif;
  width: 320px;
  background: #fcfcfc;
  padding: 20px;
  border-radius: 10px;
  box-shadow: 0px 5px 10px 0px rgba(0, 0, 0, 0.2);
  margin: auto;
}
.calculator-display {
  width: 100%;
  margin-bottom: 20px;
  font-size: 2.5em;
  text-align: right;
  color: #222;
  background: #e0e0e0;
  padding: 12px 20px;
  border-radius: 3px;
  box-shadow: 0px 5px 10px 0px rgba(0, 0, 0, 0.2);
}
.calculator-key {
  float: left;
  width: 25%;
  padding: 10px 0;
  cursor: pointer;
  font-size: 1.5em;
  background: #f0f0f0;
  border: none;
  outline: none;
  border-radius: 3px;
  box-shadow: 0px 2px 5px 0px rgba(0, 0, 0, 0.2);
  margin-bottom: 10px;
}
.calculator-key.key-clear,
.calculator-key.key-sign {
  background: #fa9595;
  color: #fff;
}
.calculator-key.key-percent {
  background: #a7bfe8;
  color: #fff;
}
.calculator-key.operator {
  background: #ffcd75;
  color: #fff;
}
.calculator-key.key-equals {
  background: #55efc4;
  color: #fff;
}
```

2)Voting System

App.js

```
import React, { Component } from 'react';
import './App.css';
class VoteCalculator extends Component {
  constructor(props) {
    super(props);
    this.state = {
      count: 0
    };
  }
  incrementCounter() {
    this.setState({
      count: this.state.count + 1
    });
  }
  render() {
    return (
      <div className="App">
        <header className="App-header">
          <h1>Vote Calculator</h1>
        </header>
        <p>You have voted {this.state.count} times.</p>
        <button onClick={() => this.incrementCounter()}>
          Vote Now
        </button>
      </div>
    );
  }
}
export default VoteCalculator;
```

App.css

```
.App {
  text-align: center;
}
.App-header {
  background-color: #282c34;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  font-size: calc(10px + 2vmin);
  color: white;
}
button {
  background-color: #282c34;
  font-size: 1rem;
  padding: 0.5rem 1.5rem;
  border: none;
  color: #fff;
  width: 100px;
  margin-top: 2rem;
  cursor: pointer;
  transition: all 0.3s ease;
  &:hover {
    background-color: #1f212b;
  }
}
```

3) Compass Clock

/
App.js

```
import React, { Component } from 'react';
import './App.css';
class App extends Component {
  constructor(props) {
    super(props);
    this.state = {
      date: new Date()
    };
  }
  componentDidMount() {
    this.timerID = setInterval(
      () => this.tick(),
      1000
    );
  }
  componentWillUnmount() {
    clearInterval(this.timerID);
  }
  tick() {
    this.setState({
      date: new Date()
    });
  }
  render() {
    return (
      <div className="App">
        <h1>Compass Clock</h1>
        <div className="clock">
          <div className="center-circle"></div>
          <div className="second-hand hand"
          style={{transform: `rotate(${this.state.date.getSeconds() * 6}deg)`}}></div>
          <div className="minute-hand hand"
          style={{transform: `rotate(${this.state.date.getMinutes() * 6}deg)`}}></div>
          <div className="hour-hand hand"
          style={{transform: `rotate(${this.state.date.getHours() * 30 +
this.state.date.getMinutes() * 0.5}deg)`}}></div>
        </div>
      </div>
    );
  }
}
export default App;
// App.css
.App {
  text-align: center;
  font-family: sans-serif;
  background-color: #e6e6e6;
}
.clock {
  position: relative;
  width: 500px;
  height: 500px;
  margin: auto;
  border-radius: 50%;
  border: 5px solid #000000;
  background-color: #ffffff;
}
.center-circle {
  position: absolute;
  width: 10px;
  height: 10px;
  border-radius: 50%;
  background-color: #000000;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
}
.hand {
  position: absolute;
  width: 50%;
  height: 5px;
  background-color: #000000;
  top: 50%;
  left: 50%;
  transform-origin: 100%;
  transition: all 0.05s ease-in-out;
}
.second-hand {
  transform: rotate(90deg);
  transform-origin: 0% 100%;
}
.minute-hand {
  transform: rotate(90deg);
  transform-origin: 0% 100%;
}
.hour-hand {
  transform: rotate(90deg);
  transform-origin: 0% 100%;
}
```

4)Form Validation

App.js

```
import React, { Component } from 'react';
import './App.css';
//Variables to store data
var emailRegex = /^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/;
var phoneRegex = /^\(\d{3}\) \d{3}-\d{4}$/;
class App extends Component {
  state = {
    formData: {
      name: '',
      email: '',
      phone: '',
      date: ''
    },
    errors: {
      name: '',
      email: '',
      phone: '',
      date: ''
    }
  };
//Function to validate form
  validateForm = () => {
    let errors = {};
    let formIsValid = true;
//Validate name
    if (!this.state.formData.name) {
      formIsValid = false;
      errors['name'] = 'Name field cannot be empty';
    }
//Validate email
    if (!this.state.formData.email) {
      formIsValid = false;
      errors['email'] = 'Email field cannot be empty';
    }
    if (typeof this.state.formData.email !== 'undefined') {
      if (!emailRegex.test(this.state.formData.email)) {
        formIsValid = false;
        errors['email'] = 'Please enter a valid email address';
      }
    }
//Validate phone
    if (!this.state.formData.phone) {
      formIsValid = false;
      errors['phone'] = 'Phone field cannot be empty';
    }
    if (typeof this.state.formData.phone !== 'undefined') {
      if (!phoneRegex.test(this.state.formData.phone)) {
        formIsValid = false;
        errors['phone'] = 'Please enter a valid phone number';
      }
    }
//Validate date
    if (!this.state.formData.date) {
      formIsValid = false;
      errors['date'] = 'Date field cannot be empty';
    }
//Update errors in state
    this.setState({
      errors: errors
    });
    return formIsValid;
  };
//Function to handle input change
  handleInputChange = e => {
    const { name, value } = e.target;
    let formData = this.state.formData;
    formData[name] = value;
    this.setState({
      formData: formData
    });
  };
//Function to handle form submission
  handleFormSubmit = e => {
    e.preventDefault();
    if (this.validateForm()) {
      console.log('Form is valid. Submitted successfully!');
    } else {
      console.log('Form is invalid');
    }
  };
//Render the form
  render() {
    return (
      <div className='App'>
        <form>
          <label>Name</label>
          <input
            name='name'
            type='text'
            value={this.state.formData.name}
            onChange={this.handleInputChange}
          />
          <span className='error'>{this.state.errors.name}</span>
          <label>Email</label>
          <input
            name='email'
            type='text'
            value={this.state.formData.email}
            onChange={this.handleInputChange}
          />
          <span className='error'>{this.state.errors.email}</span>
          <label>Phone</label>
          <input
            name='phone'
            type='text'
            value={this.state.formData.phone}
            onChange={this.handleInputChange}
          />
          <span className='error'>{this.state.errors.phone}</span>
          <label>Date</label>
          <input
            name='date'
            type='text'
            value={this.state.formData.date}
            onChange={this.handleInputChange}
          />
          <span className='error'>{this.state.errors.date}</span>
          <button onClick={this.handleFormSubmit}>Submit</button>
        </form>
      </div>
    );
  }
}
export default App;
App.css
.App {
  width: 700px;
  margin: 0 auto;
  padding: 20px;
}
label {
  display: block;
  margin-top: 10px;
  font-size: 20px;
}
input {
  margin-top: 5px;
  border: 1px solid #ccc;
  border-radius: 3px;
  padding: 5px;
  font-size: 16px;
}
button {
  background-color: #4CAF50;
  border: none;
  color: white;
  padding: 10px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  margin-top: 15px;
  margin-bottom: 15px;
  cursor: pointer;
  border-radius: 3px;
}
.error {
  color: red;
  font-size: 14px;
  margin-top: 5px;
}
```

5)Tic Tac Toe game

App.js

```
import React from 'react';
import { connect } from 'react-redux';
import { moveSquare, gameWon, gameDraw } from '../actions';
// Square component
const Square = (props) => {
  const { value, moveSquare, rowIndex, colIndex, gameWon, gameDraw } = props;
  const handleClick = () => {
    moveSquare(rowIndex, colIndex);
    gameWon();
    gameDraw();
  }
  return (
    <div className="square" onClick={handleClick}>
      {value}
    </div>
  );
};
// Board component
const Board = (props) => {
  const { squares, moveSquare, gameWon, gameDraw } = props;
  return (
    <div className="board">
      {
        squares.map((row, rowIndex) => {
          return (
            <div key={rowIndex} className="row">
              {
                row.map((value, colIndex) => {
                  return (
                    <Square
                      key={colIndex}
                      value={value}
                      moveSquare={moveSquare}
                      rowIndex={rowIndex}
                      colIndex={colIndex}
                      gameWon={gameWon}
                      gameDraw={gameDraw}
                    />
                  );
                })
              }
            </div>
          )
        })
      }
    </div>
  );
};
// App component
const App = (props) => {
  const { squares, moveSquare, gameWon, gameDraw } = props;
  return (
    <div>
      <h1>Tic Tac Toe</h1>
      <Board
        squares={squares}
        moveSquare={moveSquare}
        gameWon={gameWon}
        gameDraw={gameDraw}
      />
    </div>
  );
};
const mapStateToProps = (state) => {
  return {
    squares: state.squares
  };
};
const mapDispatchToProps = (dispatch) => {
  return {
    moveSquare: (row, col) => dispatch(moveSquare(row, col)),
    gameWon: () => dispatch(gameWon()),
    gameDraw: () => dispatch(gameDraw())
  };
};
export default connect(mapStateToProps, mapDispatchToProps)(App);
```

App.css

```
* {
  box-sizing: border-box;
}
body {
  margin: 0;
  padding: 0;
  font-family: sans-serif;
  background-color: #f1f1f1;
}
.container {
  margin: 0 auto;
  width: 600px;
  background-color: #fff;
  padding: 20px;
  text-align: center;
```

```
}
.board {
  margin-top: 20px;
  position: relative;
  width: 100%;
  height: 0;
  padding-top: 100%;
  background-color: #ccc;
  border-radius: 10px;
}
.board:after {
  content: '';
  position: absolute;
  top: 17.5%;
  left: 17.5%;
  width: 65%;
  height: 65%;
  background-color: #fff;
  border-radius: 10px;
  z-index: -1;
}
.board > div {
  position: absolute;
  width: 33.33%;
  height: 33.33%;
  background-color: #ccc;
  border-radius: 10px;
  display: flex;
  justify-content: center;
  align-items: center;
  font-size: 2rem;
  color: #fff;
  font-weight: bold;
  cursor: pointer;
  transition: all .3s;
}
.board > div:hover {
  background-color: #888;
}
.board > div.x {
  background-color: #0099ff;
}
.board > div.o {
  background-color: #ff9999;
}
.board > div.active {
  background-color: #888;
}
.turn {
  font-size: 1.2rem;
  font-weight: bold;
  color: #0099ff;
  margin-top: 20px;
}
.game-end {
  font-size: 1.2rem;
  font-weight: bold;
  color: #ff9999;
  margin-top: 20px;
}
```

6)BMI Calculator

App.js

```
import React, { useState } from "react";
const App = () => {
  const [height, setHeight] = useState(0);
  const [weight, setWeight] = useState(0);
  const [bmi, setBmi] = useState(0);
  const calculateBMI = e => {
    e.preventDefault();
    const bmiCalculation = (weight / (height * height)) * 10000;
    setBmi(bmiCalculation);
  };
  return (
    <div>
      <form onSubmit={calculateBMI}>
        <label>
          Height (cm):
          <input
            type="number"
            value={height}
            onChange={e => setHeight(parseInt(e.target.value))}
          />
        </label>
        <label>
          Weight (kg):
          <input
            type="number"
            value={weight}
            onChange={e => setWeight(parseInt(e.target.value))}
          />
        </label>
        <input type="submit" value="Calculate BMI" />
      </form>
      <p>Your BMI is: {bmi}</p>
    </div>
  );
};
export default App;
App.css
.container {
   width: 100%;
   max-width: 500px;
   margin: 0 auto;
   padding: 10px;
   text-align: center;
}
.title {
   font-size: 20px;
   font-weight: bold;
   margin-bottom: 10px;
}
.input-group {
   margin-top: 10px;
   margin-bottom: 10px;
   width: 100%;
   display: flex;
   flex-direction: row;
   justify-content: center;
   align-items: center;
}
.input-group > label {
   padding-right: 10px;
   width: 30%;
}
.input-group > input {
   width: 70%;
   padding: 5px 10px;
   border-radius: 5px;
   border: 1px solid #ccc;
}
.btn {
   background-color: #4CAF50;
   color: #fff;
   padding: 10px;
   text-align: center;
```

```
    border-radius: 10px;
    width: 100%;
    margin-top: 10px;
    margin-bottom: 10px;
    cursor: pointer;
  }
  .result {
    font-size: 18px;
    font-weight: bold;
    margin-top: 10px;
    margin-bottom: 10px;
  }
```

7) Counter app

```
import React from 'react';
import ReactDOM from 'react-dom';
import { createStore } from 'redux';
import { Provider } from 'react-redux';
import App from './App';
import reducer from './reducers';
const store = createStore(reducer);
ReactDOM.render(
  <Provider store={store}>
    <App />
  </Provider>,
  document.getElementById('root')
);
// app.css
.app {
  display: flex;
  flex-direction: column;
  width: 500px;
  margin: 0 auto;
  background-color: #f2f2f2;
  font-family: sans-serif;
}
.app-header {
  background-color: #24292e;
  color: white;
  padding: 10px;
  font-size: 20px;
}
.app-body {
  padding: 10px;
}
```

```
.app-item {
  display: flex;
  justify-content: space-between;
  padding: 10px 0;
  border-bottom: 1px solid #e2e2e2;
}
.app-item-name {
  font-weight: bold;
}
.app-item-quantity {
  font-weight: bold;
  color: #24292e;
}
.app-footer {
  padding: 10px;
  display: flex;
  flex-direction: row;
  justify-content: space-between;
  background-color: #24292e;
  color: white;
}
.app-total-price {
  font-weight: bold;
}
```

8)Temperature control app

```
import { connect } from 'react-redux';
import { incrementTemp, decrementTemp } from './actions';
const App = ({ temperature, incrementTemp, decrementTemp }) => {
  const bgColor = temperature > 0 ? 'red' : 'blue';
  return (
    <div className="App" style={{ backgroundColor: bgColor }}>
      <h1>Temperature: {temperature}</h1>
      <button onClick={incrementTemp}>+</button>
      <button onClick={decrementTemp}>-</button>
    </div>
  );
}
const mapStateToProps = state => ({
  temperature: state.temperature
});
const mapDispatchToProps = dispatch => ({
  incrementTemp: () => dispatch(incrementTemp()),
  decrementTemp: () => dispatch(decrementTemp())
});
```

```
export default connect(
  mapStateToProps,
  mapDispatchToProps
)(App);
/* app.css */
.App {
  padding: 20px;
  text-align: center;
  font-family: sans-serif;
}
button {
  padding: 10px;
  margin-top: 10px;
  font-size: 20px;
  color: white;
  background-color: black;
}
```