**Department of Computing**

**CS-213: Advanced Programming**

**Class: BSCS 7AB**

**Lab 08: React JS**

**Date: 07 November, 2019**

**Time: 10:00-01:00pm & 02:00-05:00pm**

**Muzamil Jillani**

**BSCS 7B**

**229071**

**Instructor: Dr. Sidra Sultana**

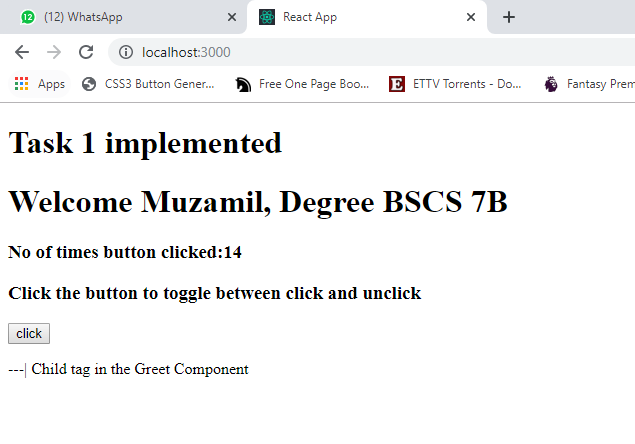
**Lab Engineer: Ms. Ayesha Asif**

**Lab Tasks**

1. You have to practice the states codes covered in class lecture

|  |
| --- |
| **App.js** |
| import React from 'react';  import ReactDOM from 'react-dom';  import './App.css';  import Greet from './Greet'  ReactDOM.render(<div>  <h1>Task 1 implemented</h1>  <Greet name="Muzamil" degree="BSCS 7B"><p>---| Child tag in the Greet Component</p></Greet>  </div>,document.getElementById('root')); |

|  |
| --- |
| **Greet.js** |
| import React, {Component} from 'react';  import './App.css';  class Greet extends Component{  constructor(props){  super(props);  this.state = {message:'click',  count : 0  }  }  render(){  return (  <div>  <h1>Welcome {this.props.name}, Degree {this.props.degree} </h1>  <h3> No of times button clicked:{this.state.count} </h3>  <h3>Click the button to toggle between click and unclick</h3>  <button onClick={()=>this.changeMessage()}>{this.state.message}</button>  <br />  {this.props.children}  </div>  )  }  increment()  {  this.setState(  {  count: this.state.count +1  }, ()=>{console.log("call back value", this.state.count)})  console.log(this.state.count)  }  changeMessage(){  this.increment();  if (this.state.message === 'click')  {  this.setState({message:'unclick'  })  console.log("clicked to unclicked")  }  else  {  this.setState({message:'click'  })  }  }  }  export default Greet; |



1. Create a timer application which asks the user for minutes. The user then click Start button and starts the timer count down. Preview @ <https://7zyz2y4p5j.csb.app/>

|  |
| --- |
| **index.js** |
| import React from 'react';  import ReactDOM from 'react-dom';  import App from './App';  import './App.css';  ReactDOM.render(  <App></App> , document.getElementById('root')); |
| **App.js** |
| import React, {Component} from 'react';  import './App.css';  class Timer extends Component{  render(){  return (  <div>  <h1>{this.props.minutes}:{this.props.seconds} </h1>  </div>  )  }  }  class App extends Component{  constructor(props){  super(props);  this.state = {  seconds:'00',  minutes:'00'  }  this.time=0;  this.intervalHander=0;  this.handleChange = this.handleChange.bind(this);  this.startCountDown = this.startCountDown.bind(this);  this.changeTime = this.changeTime.bind(this);  }  handleChange(event)  {  this.setState({  minutes: event.target.value  }, ()=>{console.log(this.state.minutes)})  }  changeTime()  {  if (this.time === 0)  {  console.log("hii")  document.getElementById("input").style.display = "block";  document.getElementById("button").style.display = "block";  document.getElementById("input").style.marginLeft = "600px";  document.getElementById("button").style.marginLeft = "650px";  clearInterval(this.intervalHandler);  }    console.log("time" +this.time);  var min = Math.floor(this.time/60); // convert to minutes  console.log("minutes: "+min)  var sec = this.time%60;  console.log("start count down")  if (min<10 && sec < 10)  {  this.setState({  minutes: "0"+min,  seconds:"0"+sec  })  }  else if (sec <10)  {  this.setState({  minutes: min,  seconds:"0"+sec  })  }  else if (min <10)  {  this.setState({  minutes: "0"+min,  seconds:sec  })  }  else  {  this.setState({  minutes: min,  seconds:sec  })  }    this.time--;    }  startCountDown(){  this.time= (this.state.minutes)\*60;  this.intervalHandler = setInterval(this.changeTime, 1000);  document.getElementById("input").style.display = "none";  document.getElementById("button").style.display = "none";    console.log("start count down")  }  render(){  return(  <div className="App">  <h1> Timer </h1>  <input id="input" type="number" onChange={this.handleChange} required />  <Timer minutes={this.state.minutes} seconds={this.state.seconds} />  <button id="button" onClick={this.startCountDown}>Start</button>    </div>  )  }  }  export default App; |

**\**