**Practical No 2**

**AIM: Using States in Class components**

**Q. 1 Updating message using state**

🡺stateclass.js

import React from "react";

import { Component } from "react";

class Message extends Component{

constructor()

{

super()

this.state = {

message: "Welcome Visitors"

}

}

render()

{

return(

<div>

<h1> {this.state.message}</h1>

<button onClick={()=> this.changeMessage()

}>Click here</button>

</div>

)

}

changeMessage()

{

this.setState(

{

message:"Thank You for visiting!"

}

)

}

}

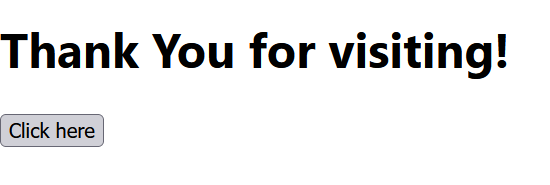
export default Message;

🡺 App.js

import Message from "./components/stateclass";

function App() {

return (

 <div className="App">

<Message/>

</div>

);

}

export default App;

O/p:

**Q2 Create a counter in Reactjs**

🡺counter.js

import React from "react";

import { Component } from "react";

class Counter extends Component{

constructor()

{

super()

{

this.state = {

count: 0

}

}

}

increment(){

//never modify the state directly, instead use this.setState()

/\* this.setState(

{

count: this.state.count + 1

},

()=> {pconsole.log("Callback Value: ",this.state.count)} //value 1

//setting the 2nd parameter call back function after the updation

) \*/

//console.log(this.state.count) //value 0

//update the state based on previous state, pass a function as an arg to setstate() instead of passing object

this.setState( (prevState) => ({

count: prevState.count + 1

}

)

)

console.log(this.state.count)

}

incrementFour()

{

this.increment()

this.increment()

this.increment()

this.increment()

//react may group multiple state calls into a single update for better performance

}

render()

{

return(

<div>

{ this.state.count} <br></br>

<button onClick={()=> this.incrementFour()}>Increment</button>

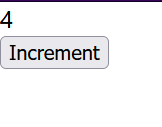
</div>

)

}

}

export default Counter;

🡺App.js

import Counter from "./components/counter";

function App() {

return (

<div className="App">

<Counter/>

</div>

);

}

export default App;

