**App.js**

import './App.css';

import ReducerDataFetching from './Reducer/ReducerDataFetching';

function App() {

  return (

    <div className="App">

      <ReducerDataFetching/>

    </div>

  );

}

export default App;

**ReducerDataFetching.js**

import React, { useState, useEffect } from "react";

import axios from "axios";

function ReducerDataFetching() {

  const [loading, setLoading] = useState(true);

  const [post, setPost] = useState({});

  const [error, setError] = useState("");

  useEffect(() => {

    axios

      .get(`https://jsonplaceholder.typicode.com/posts/1`)

      .then((response) => {

        console.log(response);

        setLoading(false);

        setPost(response.data);

        setError("");

      })

      .catch((error) => {

        console.log(error);

        setLoading(false);

        setPost({});

        setError("Something went wrong");

      });

  }, []);

  return (

    <div>

      {loading ? "Loading" : post.title}

      {error ? error : null}

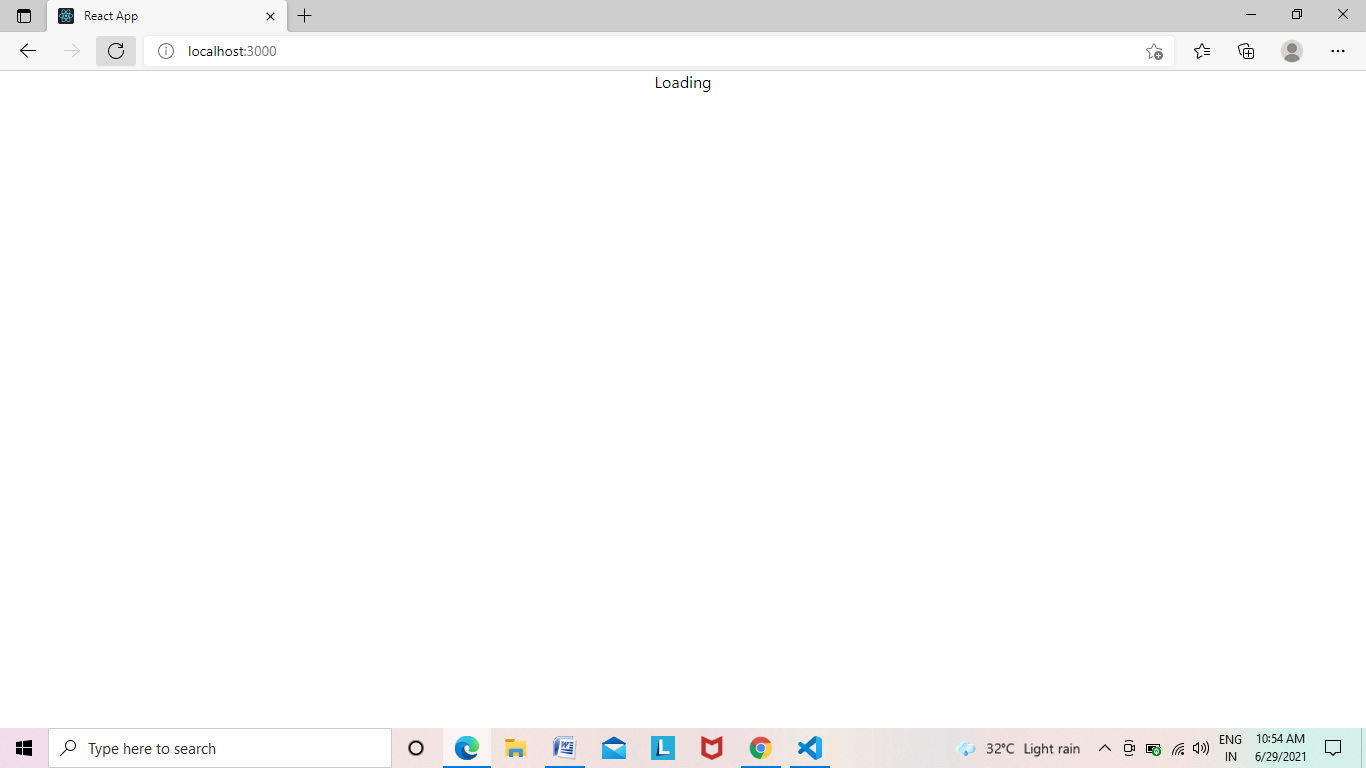
    </div>

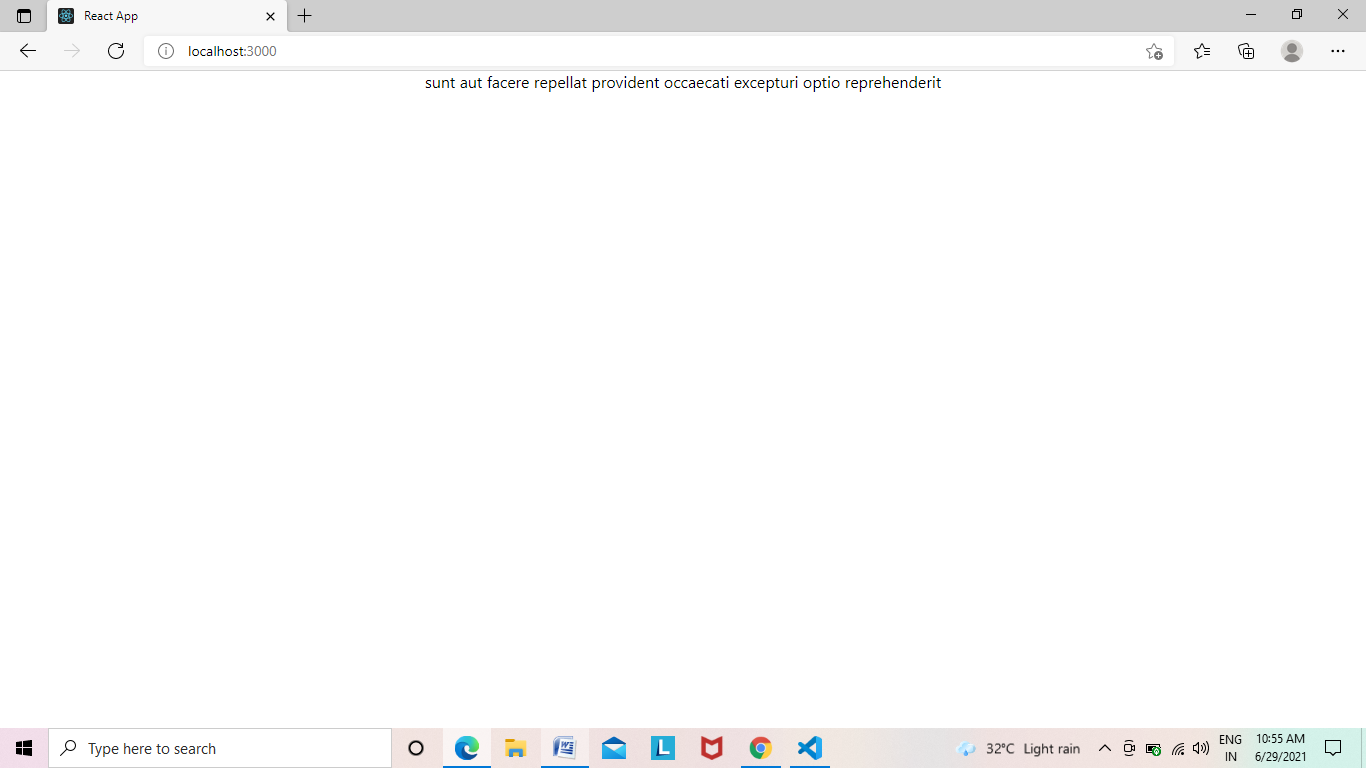
  );

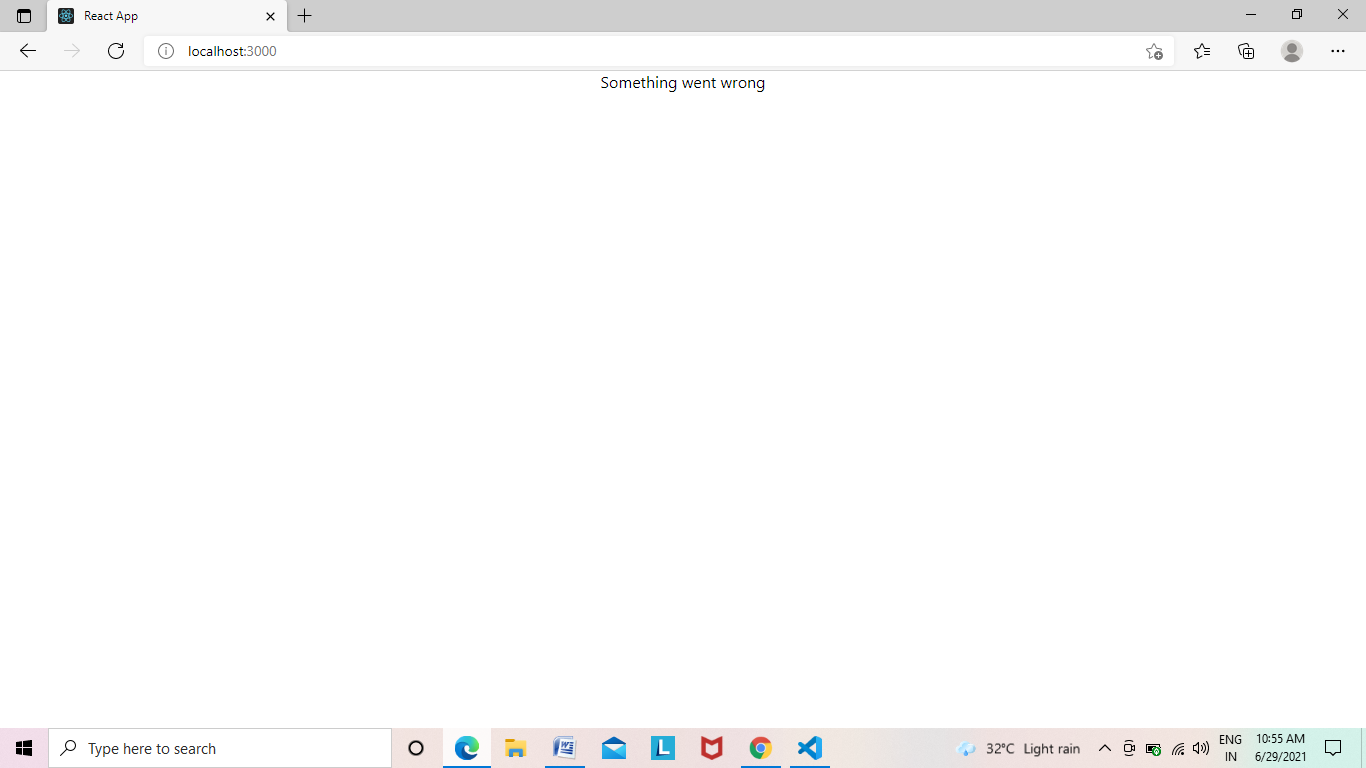
}

export default ReducerDataFetching;

**Output:**

****

****

****

**App.js**

import './App.css';

import ReducerDataFetching2 from './Reducer/ReducerDataFetching2';

function App() {

  return (

    <div className="App">

      <ReducerDataFetching2/>

    </div>

  );

}

export default App;

**ReducerDataFetching2.js**

import React, { useReducer, useEffect } from "react";

import axios from "axios";

const initialState = {

  loading: true,

  post: {},

  error: "",

};

const reducer = (state, action) => {

  switch (action.type) {

    case "FETCH\_SUCCESS":

      return {

        loading: false,

        post: action.payload,

        error: "",

      };

    case "FETCH\_ERROR":

      return {

        loading: false,

        post: {},

        error: "Something went wrong",

      };

    default:

      return state;

  }

};

function ReducerDataFetching2() {

  const [state, dispatch] = useReducer(reducer, initialState);

  useEffect(() => {

    axios

      .get(`https://jsonplaceholder.typicode.com/posts/1`)

      .then((response) => {

        console.log(response);

        dispatch({ type: "FETCH\_SUCCESS", payload: response.data });

      })

      .catch((error) => {

        console.log(error);

        dispatch({ type: "FETCH\_ERROR" });

      });

  }, []);

  return (

    <div>

      {state.loading ? "Loading" : state.post.title}

      {state.error ? state.error : null}

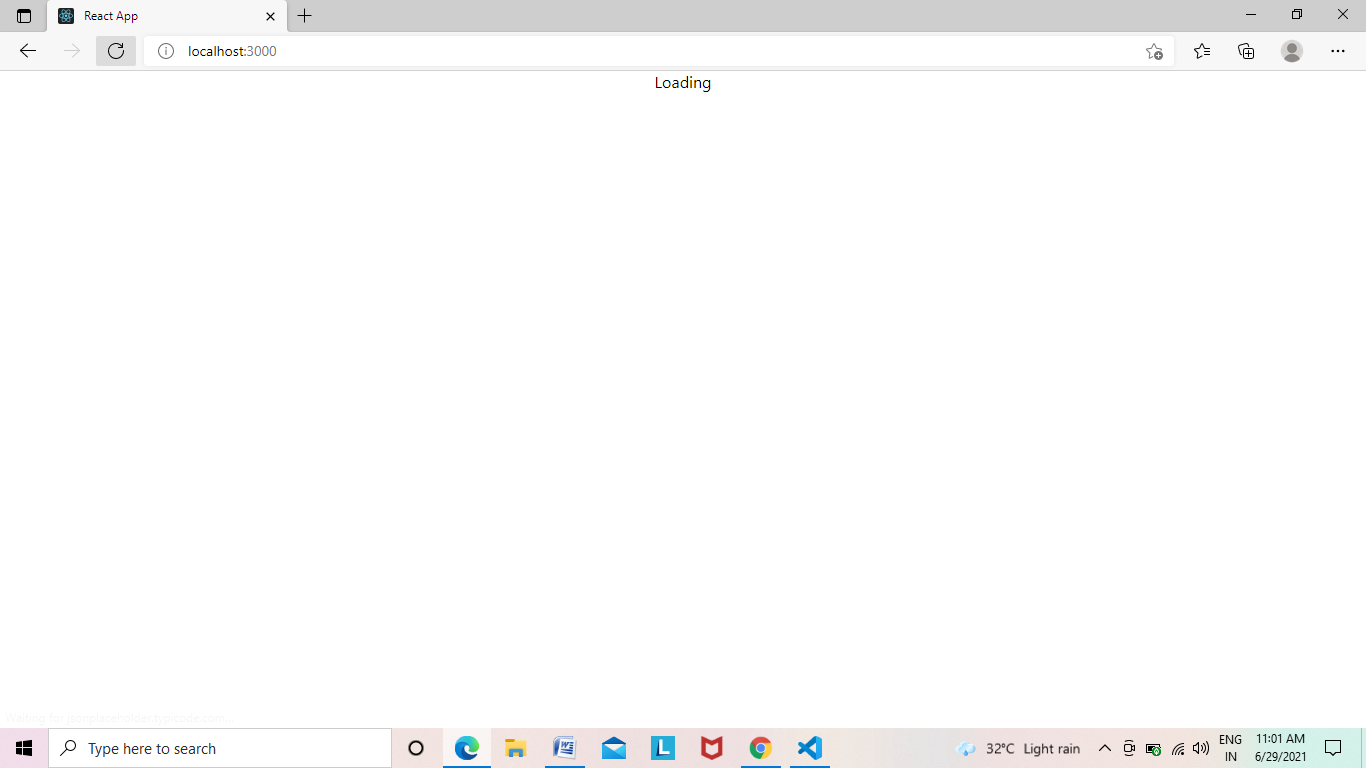
    </div>

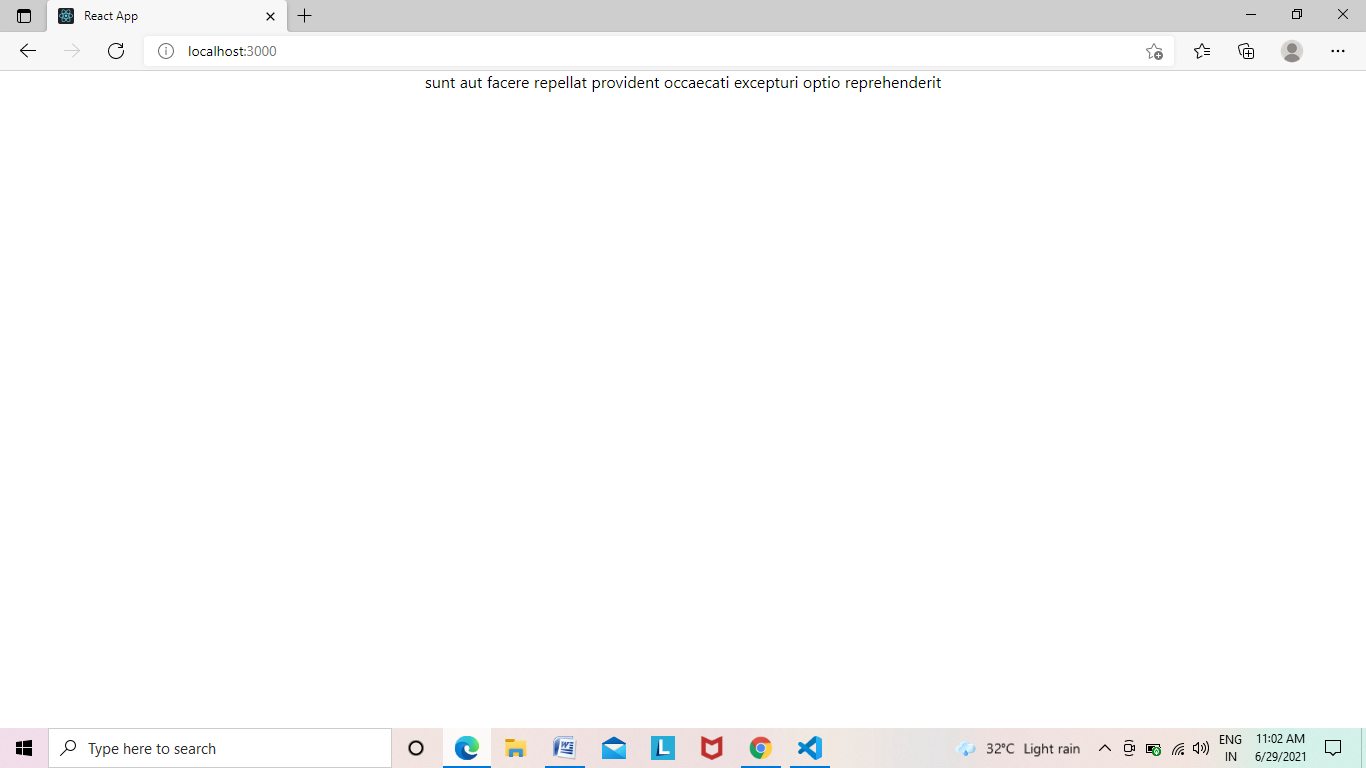
  );

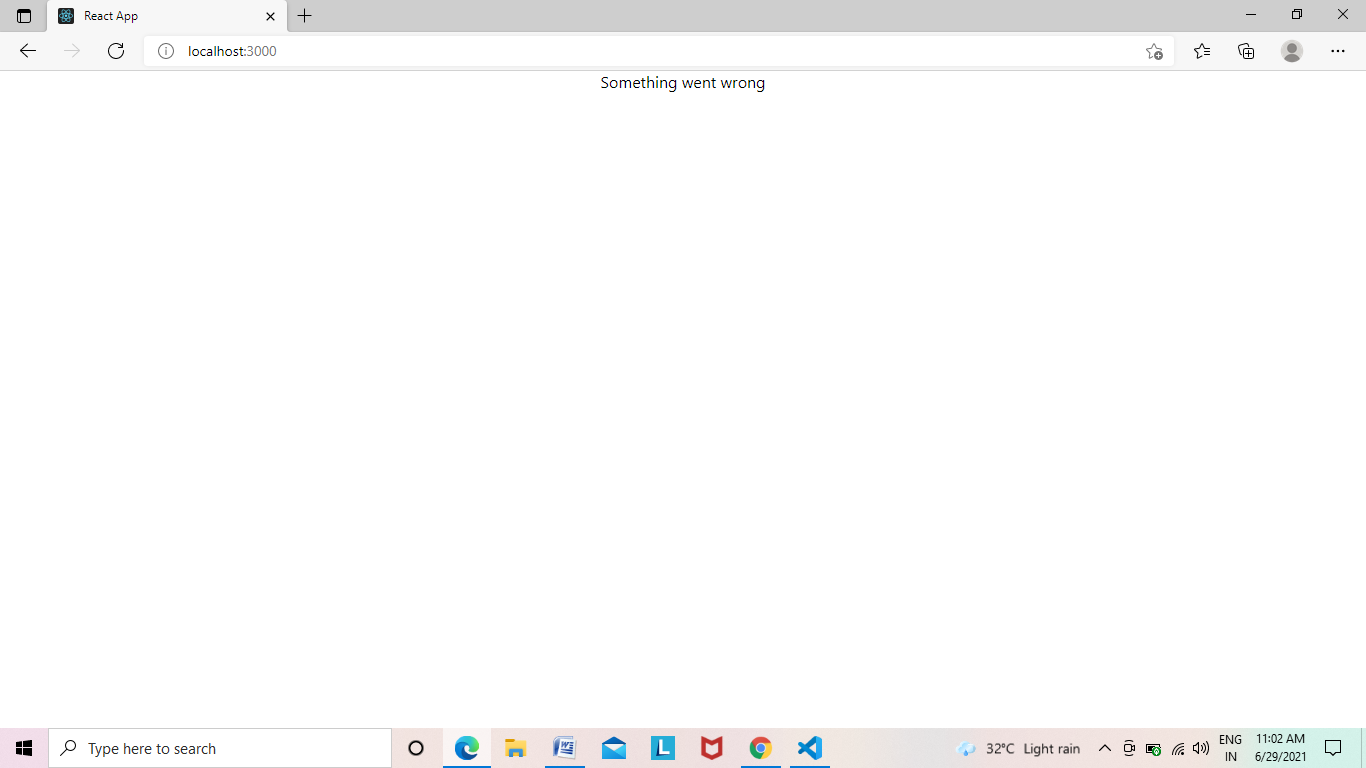
}

export default ReducerDataFetching2;

**Output:**

****

****

****

**App.js**

import './App.css';

import ParentComponent from './useCallback/ParentComponent';

function App() {

  return (

    <div className="App">

      <ParentComponent/>

    </div>

  );

}

export default App;

**Button.js**

import React from "react";

function Button({ handlerClick, children }) {

  console.log("Rendering Button - ", children);

  return <button onClick={handlerClick}>{children}</button>;

}

export default Button;

**Title.js**

import React from 'react'

function Title() {

    console.log('Rendering title')

    return (

        <div>

            <h2>Without using Usecallback hook</h2>

        </div>

    )

}

export default Title

**CountBack.js**

import React from 'react'

function CountBack({text,count}) {

    console.log(`Count Rendering ${text}`)

    return (

        <div>

            {text} - {count}

        </div>

    )

}

export default CountBack

**ParentComponent.js**

import React, { useState, useCallback } from "react";

import Button from "./Button";

import Title from "./Title";

import CountBack from "./CountBack";

function ParentComponent() {

  const [age, setAge] = useState(25);

  const [salary, setSalary] = useState(50000);

  const incrementAge = () => {

    console.log("inside incrementAge");

    setAge(age + 1);

  };

  const incrementSalary = () =>{

    console.log("inside incrementSalary");

    setSalary(salary + 1000);

  };

  return (

    <div>

      <Title />

      <CountBack text="Age" count={age} />

      <Button handlerClick={incrementAge}>incrementAge</Button>

      <CountBack text="Salary" count={salary} />

      <Button handlerClick={incrementSalary}>incrementSalary</Button>

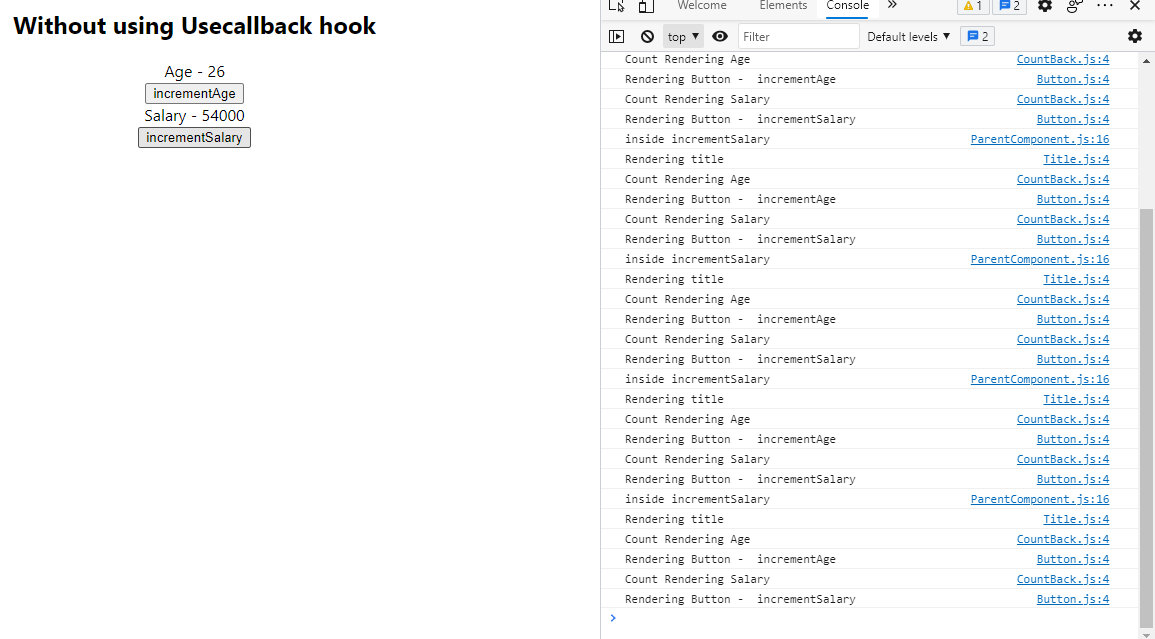
    </div>

  );

}

export default ParentComponent;

**Output**

****

**TO AVOID COMPONENT RERENDERING**

**Title.js**

import React from 'react'

function Title() {

    console.log('Rendering title')

    return (

        <div>

            <h2>React Memo Use</h2>

        </div>

    )

}

export default React.memo(Title)

**Button.js**

import React from "react";

function Button({ handlerClick, children }) {

  console.log("Rendering Button - ", children);

  return <button onClick={handlerClick}>{children}</button>;

}

export default React.memo(Button)

**CountBack.js**

import React from 'react'

function CountBack({text,count}) {

    console.log(`Count Rendering ${text}`)

    return (

        <div>

            {text} - {count}

        </div>

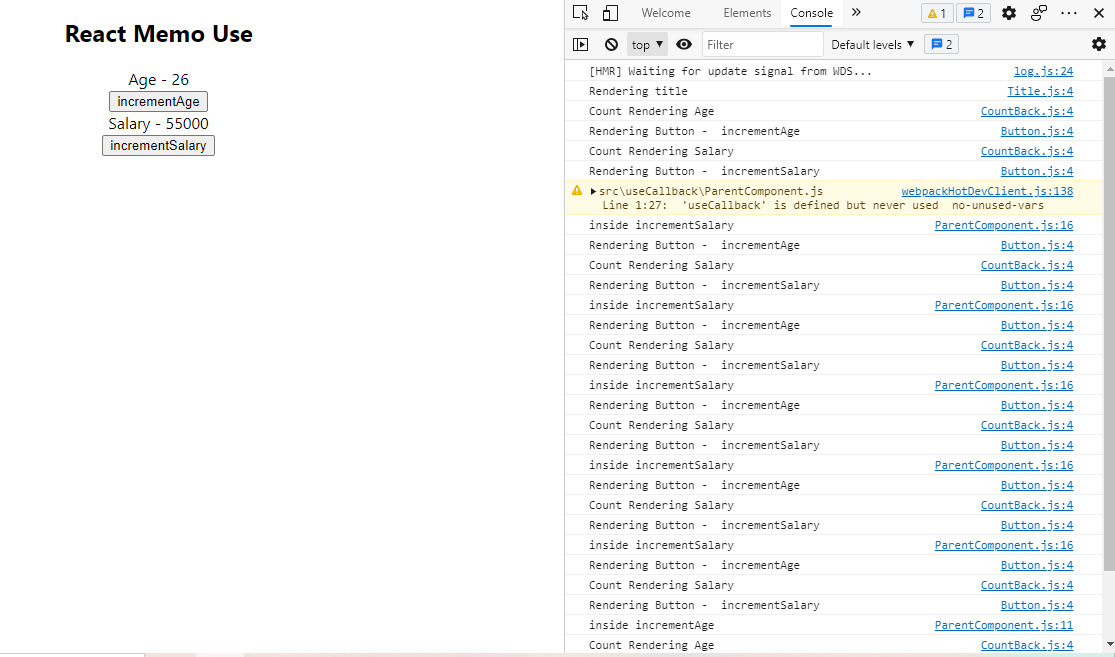
    )

}

export default React.memo(CountBack)

**//ReactComponent.js code is same as previous code**

**Output:**

****

**ParentComponent.js**

import React, { useState, useCallback } from "react";

import Button from "./Button";

import Title from "./Title";

import CountBack from "./CountBack";

function ParentComponent() {

  const [age, setAge] = useState(25);

  const [salary, setSalary] = useState(50000);

  const incrementAge = useCallback(() => {

    console.log("inside incrementAge");

    setAge(age + 1);

  }, [age]);

  const incrementSalary = useCallback(() =>{

    console.log("inside incrementSalary");

    setSalary(salary + 1000);

  }, [salary]);

  return (

    <div>

      <Title />

      <CountBack text="Age" count={age} />

      <Button handlerClick={incrementAge}>incrementAge</Button>

      <CountBack text="Salary" count={salary} />

      <Button handlerClick={incrementSalary}>incrementSalary</Button>

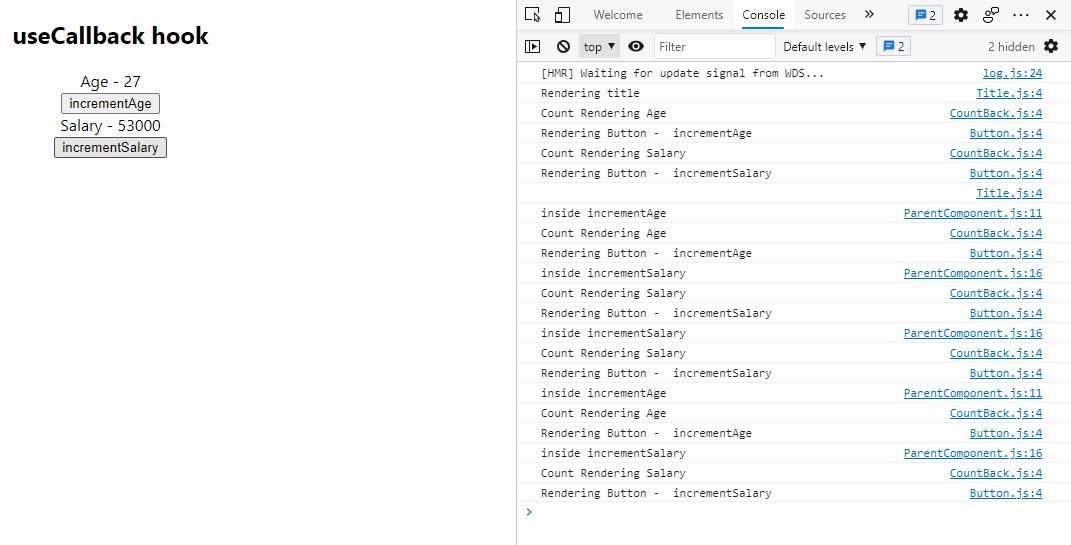
    </div>

  );

}

export default ParentComponent;

**Output:**

****