2.1 a Take a class-based Counter component. Convert it into a functional component using useState for managing state and useEffect to log a message every time the component renders.

App.js

import React from 'react';

import Counter from './Counter';

function App() {

  return (

    <div>

      <h1>React Counter App</h1>

      <Counter />

    </div>

  );

}

export default App;

Counter.js

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

function Counter() {

  const [count, setCount] = useState(0);

  // Run once when component mounts

  useEffect(() => {

    console.log('Component mounted');

  }, []);

  // Run every time 'count' changes

  useEffect(() => {

    console.log('Count updated to:', count);

  }, [count]);

  const increment = () => {

    setCount(count + 1);

  };

  return (

    <div>

      <h2>Count: {count}</h2>

      <button onClick={increment}>Increment</button>

    </div>

  );

}

export default Counter;

App.test.js

import { render, screen } from '@testing-library/react';

import App from './App';

test('renders learn react link', () => {

  render(<App />);

  const linkElement = screen.getByText(/learn react/i);

  expect(linkElement).toBeInTheDocument();

});

2.2 Starts a timer on mount using setInterval • Increments a seconds state every second • Cleans up the timer on unmount (componentWillUnmount)

App.js

import React, { useState } from 'react';

import Timer from './Timer';

function App() {

  const [showTimer, setShowTimer] = useState(true);

  const toggleTimer = () => {

    setShowTimer(!showTimer);

  };

  return (

    <div>

      <button onClick={toggleTimer}>

        {showTimer ? 'Unmount Timer' : 'Mount Timer'}

      </button>

      {/\* Conditionally render Timer component \*/}

      {showTimer && <Timer />}

    </div>

  );

}

export default App;

Timer.js

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

function Timer() {

  const [seconds, setSeconds] = useState(0);

  useEffect(() => {

    const interval = setInterval(() => {

      setSeconds(prev => prev + 1);

    }, 1000);

    // Cleanup runs on unmount

    return () => clearInterval(interval);

  }, []);

  return <h2>Seconds: {seconds}</h2>;

}

export default Timer;

App.test.js

import { render, screen } from '@testing-library/react';

import App from './App';

test('renders learn react link', () => {

  render(<App />);

  const linkElement = screen.getByText(/learn react/i);

  expect(linkElement).toBeInTheDocument();

});