LAB ASSIGNMENT NO – 7b

Aim: Write a program to implement the concept of React Hooks.

Theory:

React hooks are a feature introduced in React 16.8 to allow functional components to have state and side effects, which were previously only possible in class components. They enable you to manage component state and lifecycle events without using class components.

There are several built-in hooks in React, but the two fundamental ones are:

1. useState:

   useState allows functional components to hold and update their own local state. It takes an initial value as an argument and returns an array with two elements: the current state value and a function to update it.

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

2. useEffect:

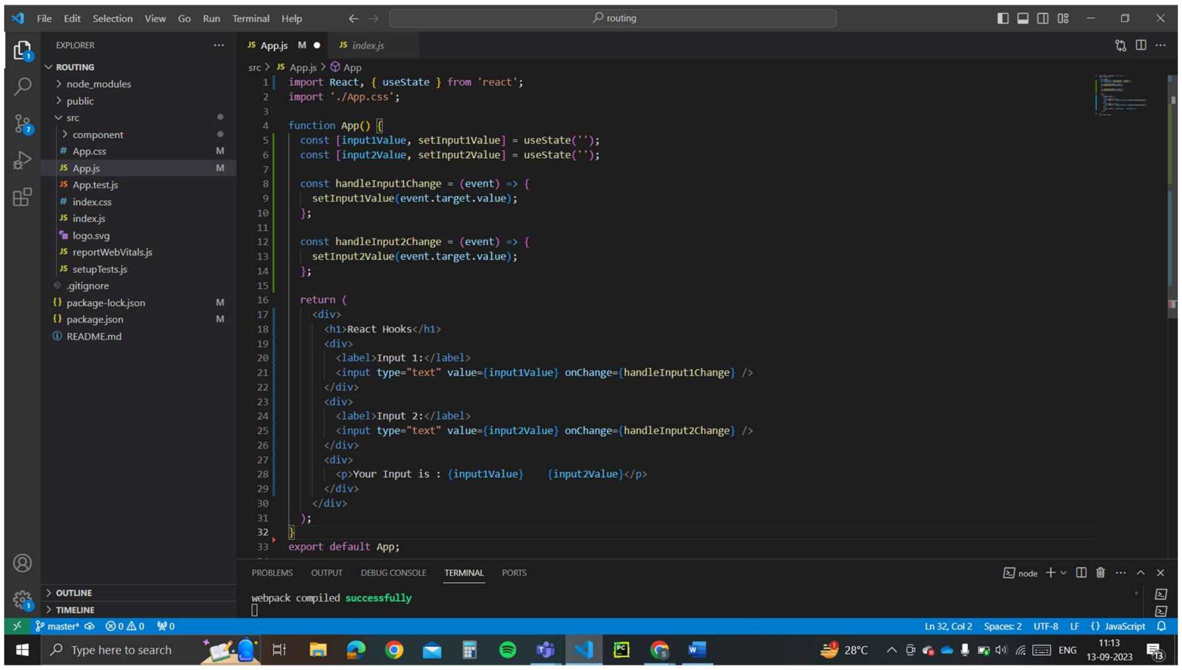
   useEffect is used to handle side effects in functional components, such as data fetching, subscriptions, or manually changing the DOM. It takes two arguments: a function to execute the side effect and an optional array of dependencies that trigger the effect when changed.

   ```
   useEffect(() => {
     // Side effect code here
     // This runs after each render
     // Cleanup code (optional) when the component is unmounted or
   dependencies change
     return () => {
       };
   }, [dependencies]);
   ```

Conclusion: This experiment demonstrates the effectiveness of React hooks in simplifying state management and side effect handling within functional components. Hooks, such as useState and useEffect, significantly enhance component logic and reusability, leading to more efficient and readable code. The ability to encapsulate state and lifecycle events in functional components marks a pivotal advancement, promoting a streamlined development process and improved maintainability.

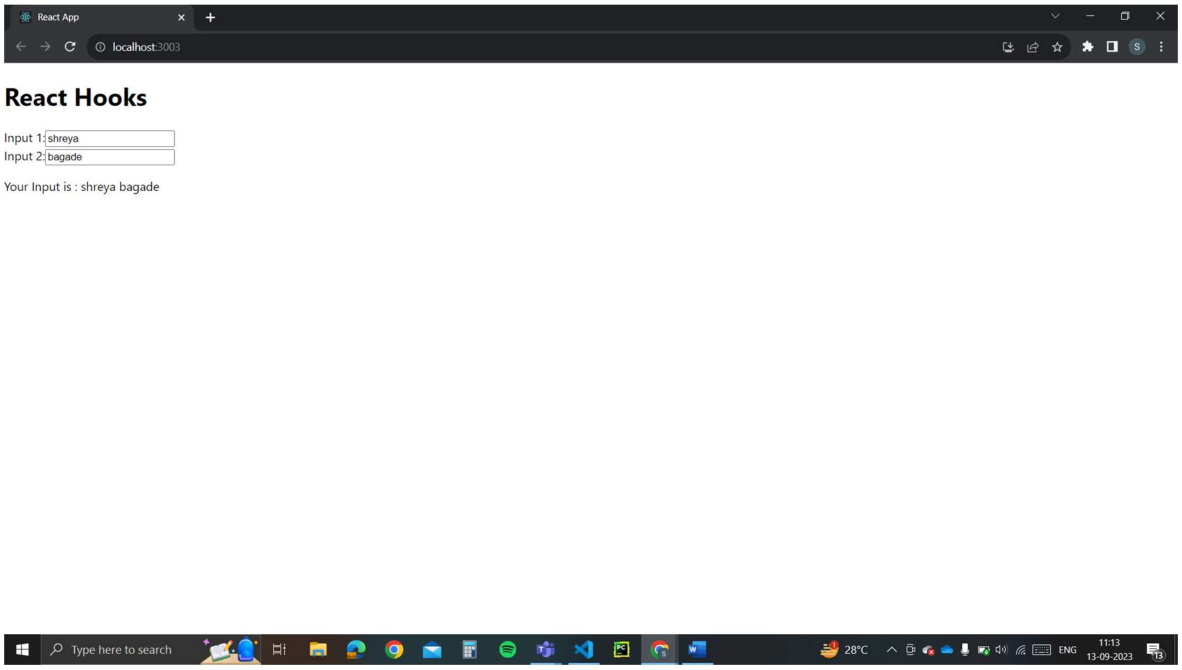
Shreya Bagade/ SEM V/ TE.IT/ T11/ Roll No-09/ Assignment no-7b

Code:



```
import React, { useState } from 'react';
import './App.css';

function App() {
  const [input1Value, setInput1Value] = useState('');
  const [input2Value, setInput2Value] = useState('');

  const handleInput1Change = (event) => {
    setInput1Value(event.target.value);
  };

  const handleInput2Change = (event) => {
    setInput2Value(event.target.value);
  };

  return (
    <div>
      <h1>React Hooks</h1>
      <div>
        <label>Input 1:</label>
        <input type="text" value={input1Value} onChange={handleInput1Change} />
      </div>
      <div>
        <label>Input 2:</label>
        <input type="text" value={input2Value} onChange={handleInput2Change} />
      </div>
      <div>
        <p>Your Input is : {input1Value}    {input2Value}</p>
      </div>
    </div>
  );
}
export default App;
```

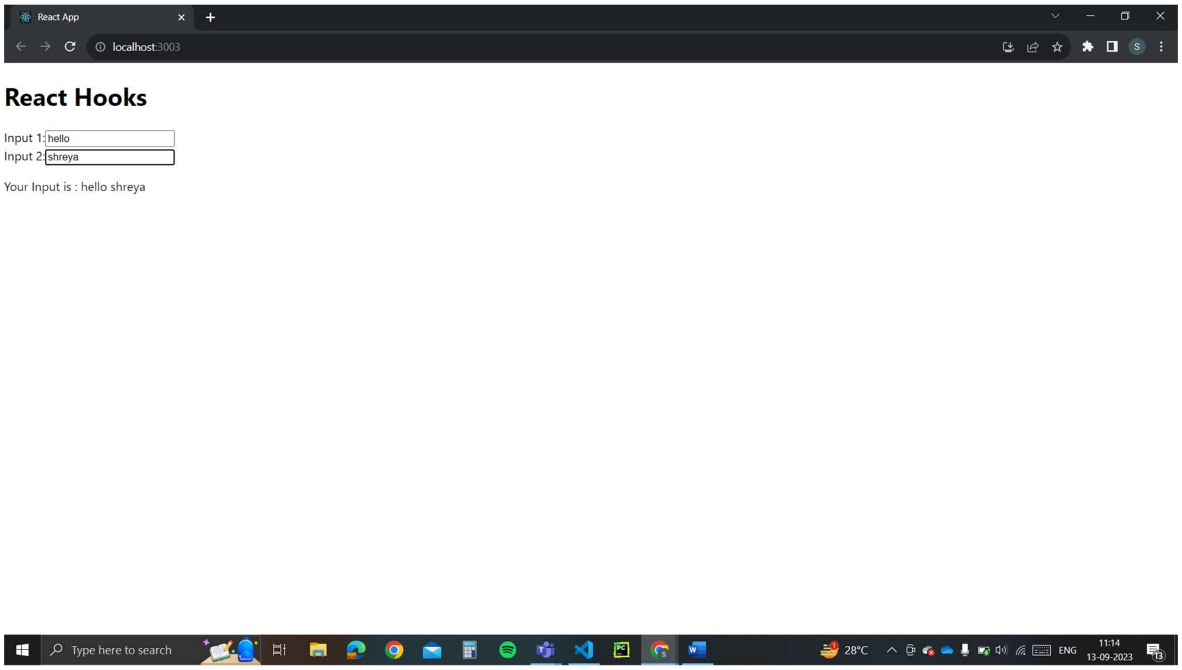
```
webpack compiled successfully
```



React Hooks

Input 1: shreya
Input 2: bagade

Your Input is : shreya bagade



React Hooks

Input 1: hello
Input 2: shreya

Your Input is : hello shreya

<u>Lab Outcome :</u> LO5- Construct front end applications using React.