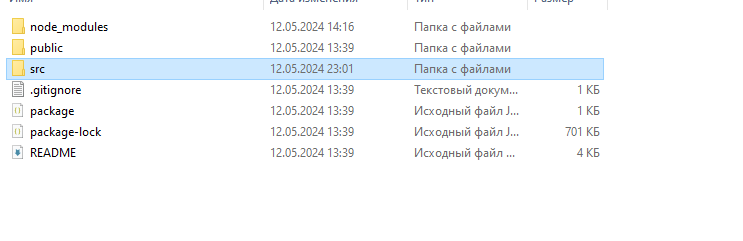
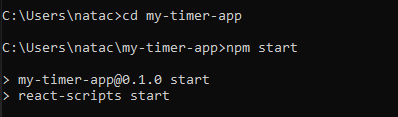
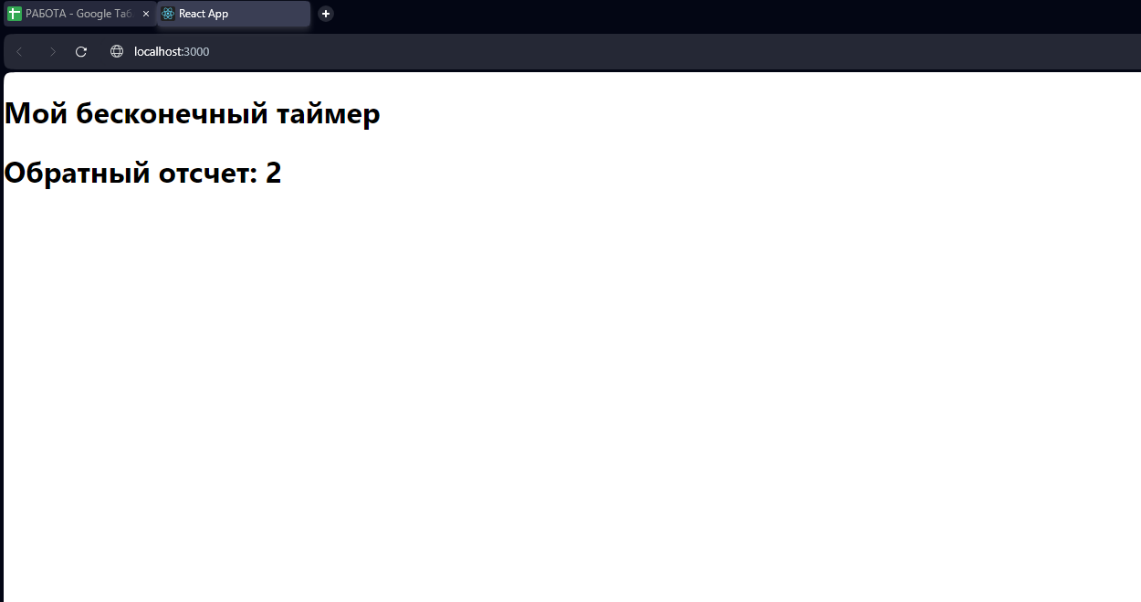
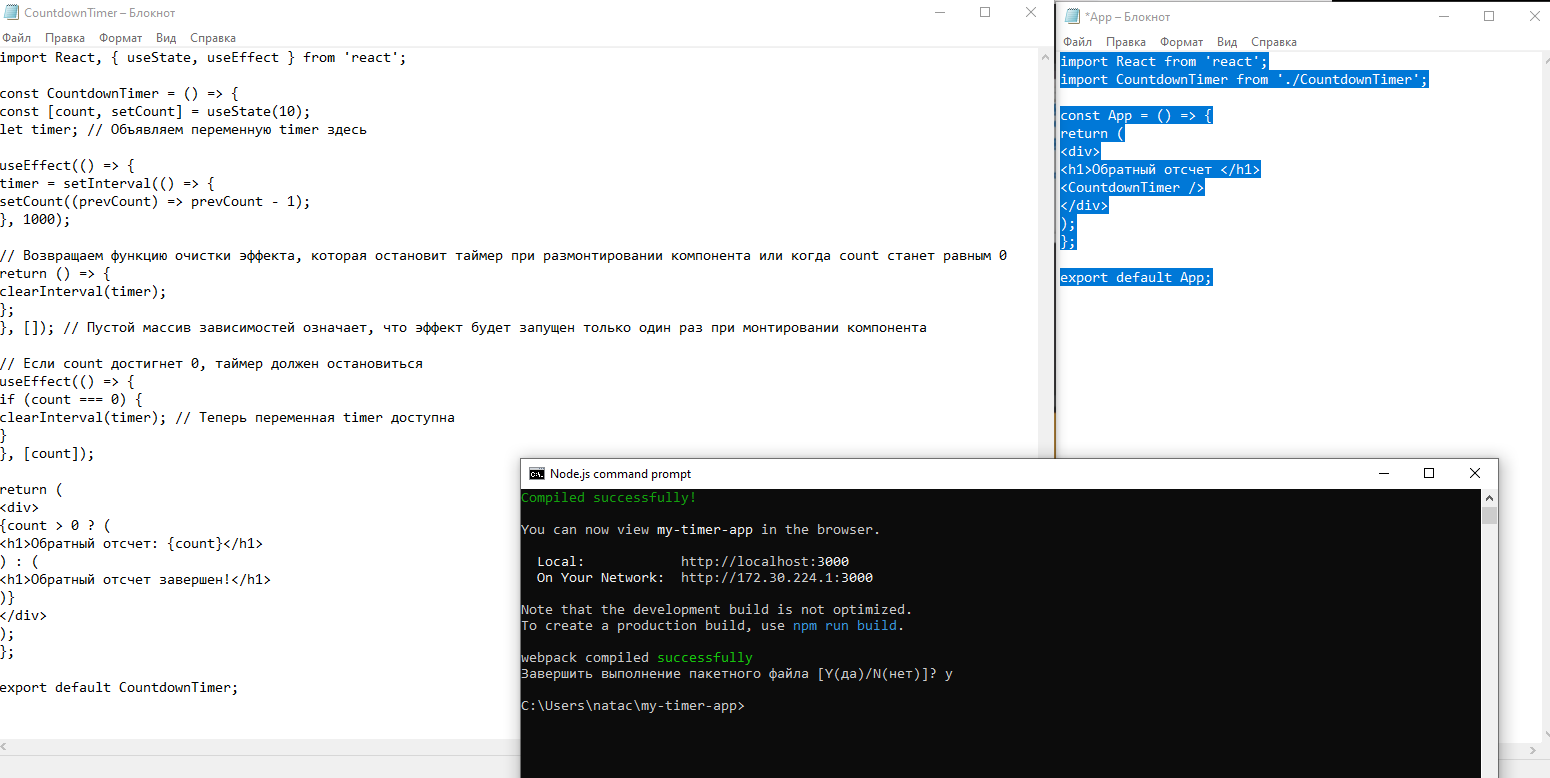
**ЛАБОРАТОРНАЯ РАБОТА №8**

**Создадим** 



**Задание 1**



**CountdownTimer.js**

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

const CountdownTimer = () => {

const [count, setCount] = useState(10);

let timer; // Объявляем переменную timer здесь

useEffect(() => {

timer = setInterval(() => {

setCount((prevCount) => prevCount - 1);

}, 1000);

// Возвращаем функцию очистки эффекта, которая остановит таймер при размонтировании компонента или когда count станет равным 0

return () => {

clearInterval(timer);

};

}, []); // Пустой массив зависимостей означает, что эффект будет запущен только один раз при монтировании компонента

// Если count достигнет 0, таймер должен остановиться

useEffect(() => {

if (count === 0) {

clearInterval(timer); // Теперь переменная timer доступна

}

}, [count]);

return (

<div>

{count > 0 ? (

<h1>Обратный отсчет: {count}</h1>

) : (

<h1>Обратный отсчет завершен!</h1>

)}

</div>

);};

export default CountdownTimer;

**APP.js**

import React from 'react';

import CountdownTimer from './CountdownTimer';

const App = () => {

return (

<div>

<h1>Обратный отсчет </h1>

<CountdownTimer />

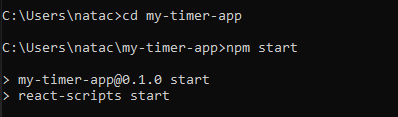
</div>

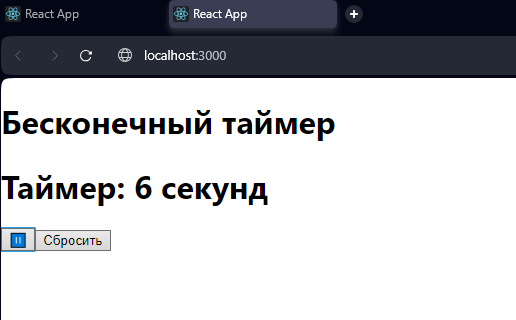
);

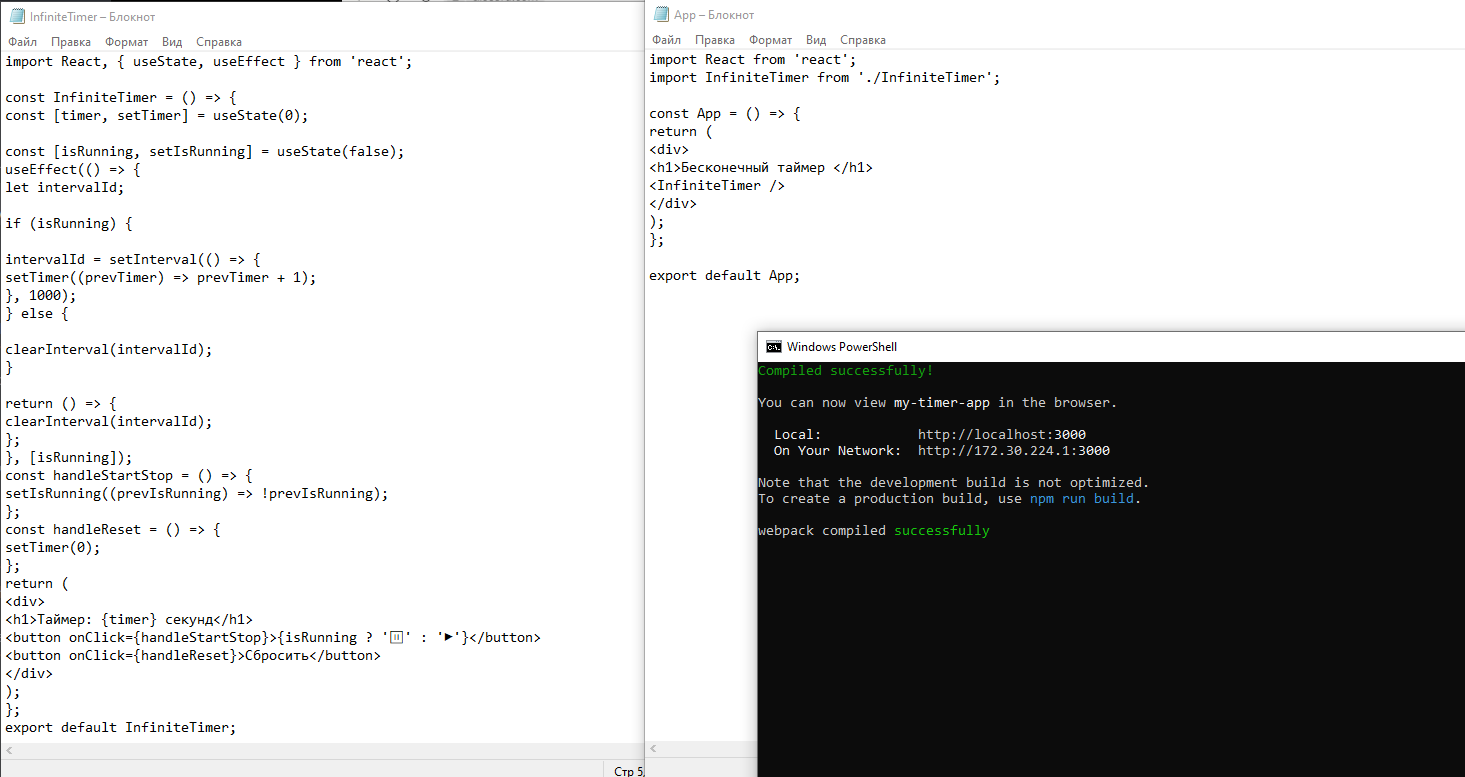
};

export default App;

**Задание 2**







**InfiniteTimer.js**

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

const InfiniteTimer = () => {

const [timer, setTimer] = useState(0);

const [isRunning, setIsRunning] = useState(false);

useEffect(() => {

let intervalId;

if (isRunning) {

intervalId = setInterval(() => {

setTimer((prevTimer) => prevTimer + 1);

}, 1000);

} else {

clearInterval(intervalId);

}

return () => {

clearInterval(intervalId);

};

}, [isRunning]);

const handleStartStop = () => {

setIsRunning((prevIsRunning) => !prevIsRunning);

};

const handleReset = () => {

setTimer(0);

};

return (

<div>

<h1>Таймер: {timer} секунд</h1>

<button onClick={handleStartStop}>{isRunning ? '⏸' : '▶'}</button>

<button onClick={handleReset}>Сбросить</button>

</div>

);

};

export default InfiniteTimer;

**APP.js**

import React from 'react';

import InfiniteTimer from './InfiniteTimer';

const App = () => {

return (

<div>

<h1>Бесконечный таймер </h1>

<InfiniteTimer />

</div>

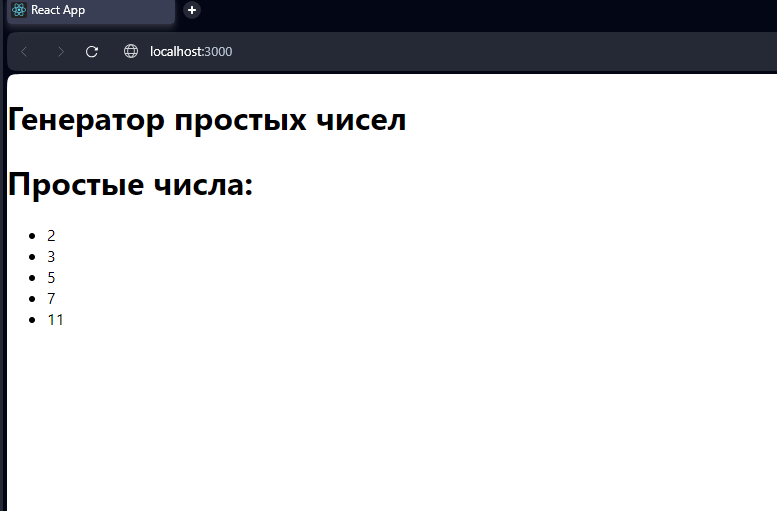
);

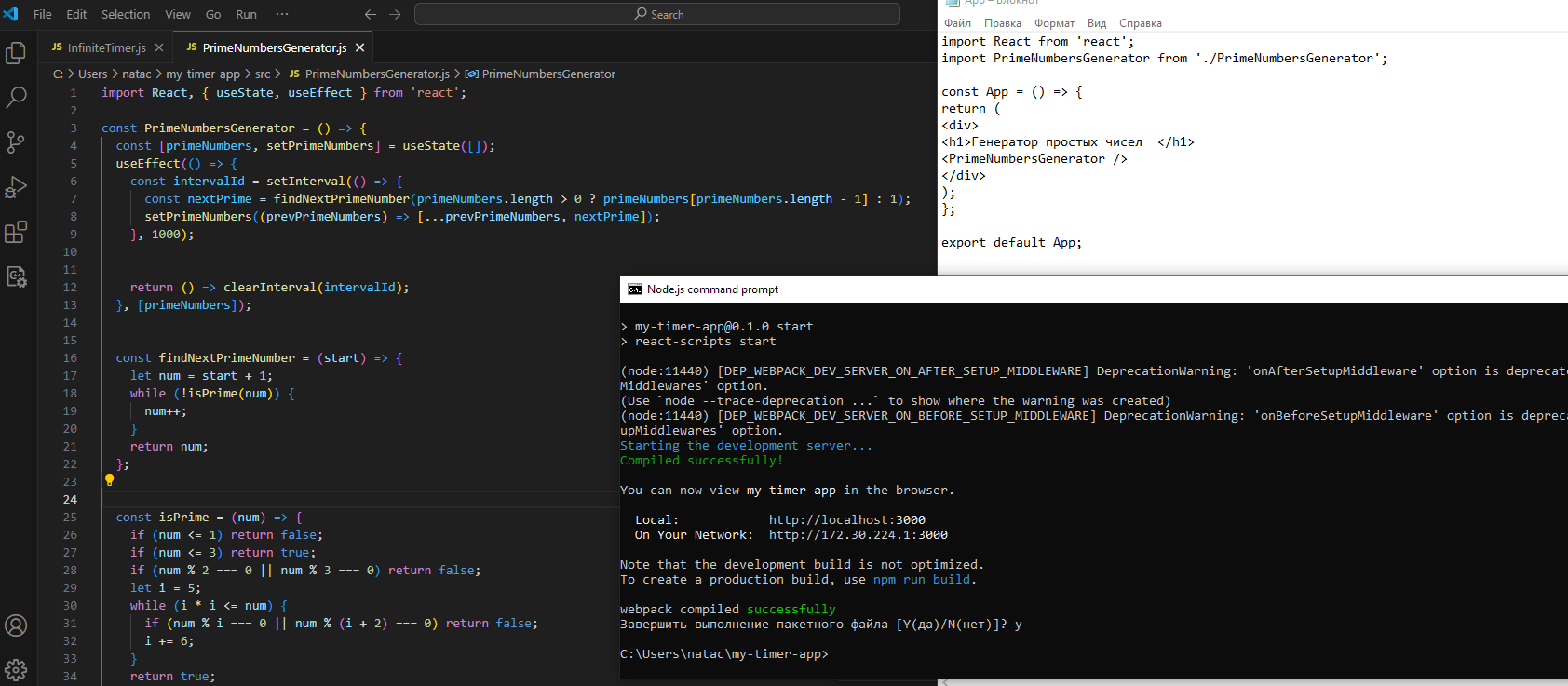
};

export default App;

**ЗАДАНИЕ 3**







**PrimeNumbersGenerator.js**

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

const PrimeNumbersGenerator = () => {

  const [primeNumbers, setPrimeNumbers] = useState([]);

  useEffect(() => {

    const intervalId = setInterval(() => {

      const nextPrime = findNextPrimeNumber(primeNumbers.length > 0 ? primeNumbers[primeNumbers.length - 1] : 1);

      setPrimeNumbers((prevPrimeNumbers) => [...prevPrimeNumbers, nextPrime]);

    }, 1000);

    return () => clearInterval(intervalId);

  }, [primeNumbers]);

  const findNextPrimeNumber = (start) => {

    let num = start + 1;

    while (!isPrime(num)) {

      num++;

    }

    return num;

  };

  const isPrime = (num) => {

    if (num <= 1) return false;

    if (num <= 3) return true;

    if (num % 2 === 0 || num % 3 === 0) return false;

    let i = 5;

    while (i \* i <= num) {

      if (num % i === 0 || num % (i + 2) === 0) return false;

      i += 6;

    }

    return true;

  };

  return (

    <div>

      <h1>Простые числа:</h1>

      <ul>

        {primeNumbers.map((primeNumber, index) => (

          <li key={index}>{primeNumber}</li>

        ))}

      </ul>

    </div>

  );

};

export default PrimeNumbersGenerator;

**APP.js**

import React from 'react';

import PrimeNumbersGenerator from './PrimeNumbersGenerator';

const App = () => {

return (

<div>

<h1>Генератор простых чисел </h1>

<PrimeNumbersGenerator />

</div>

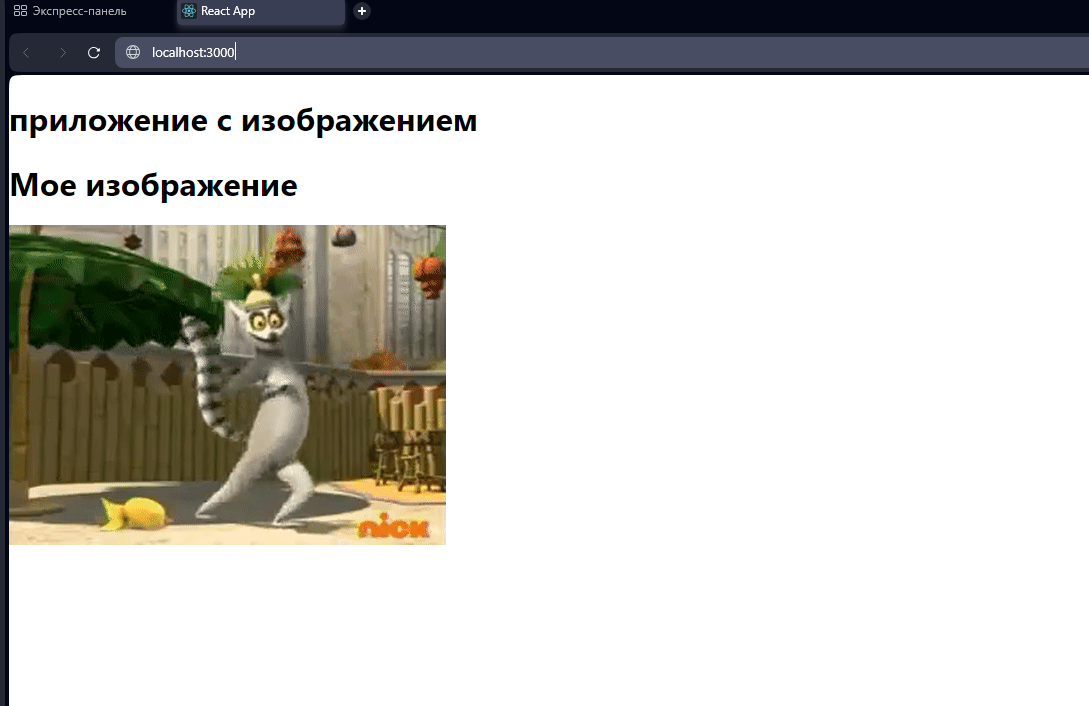
);

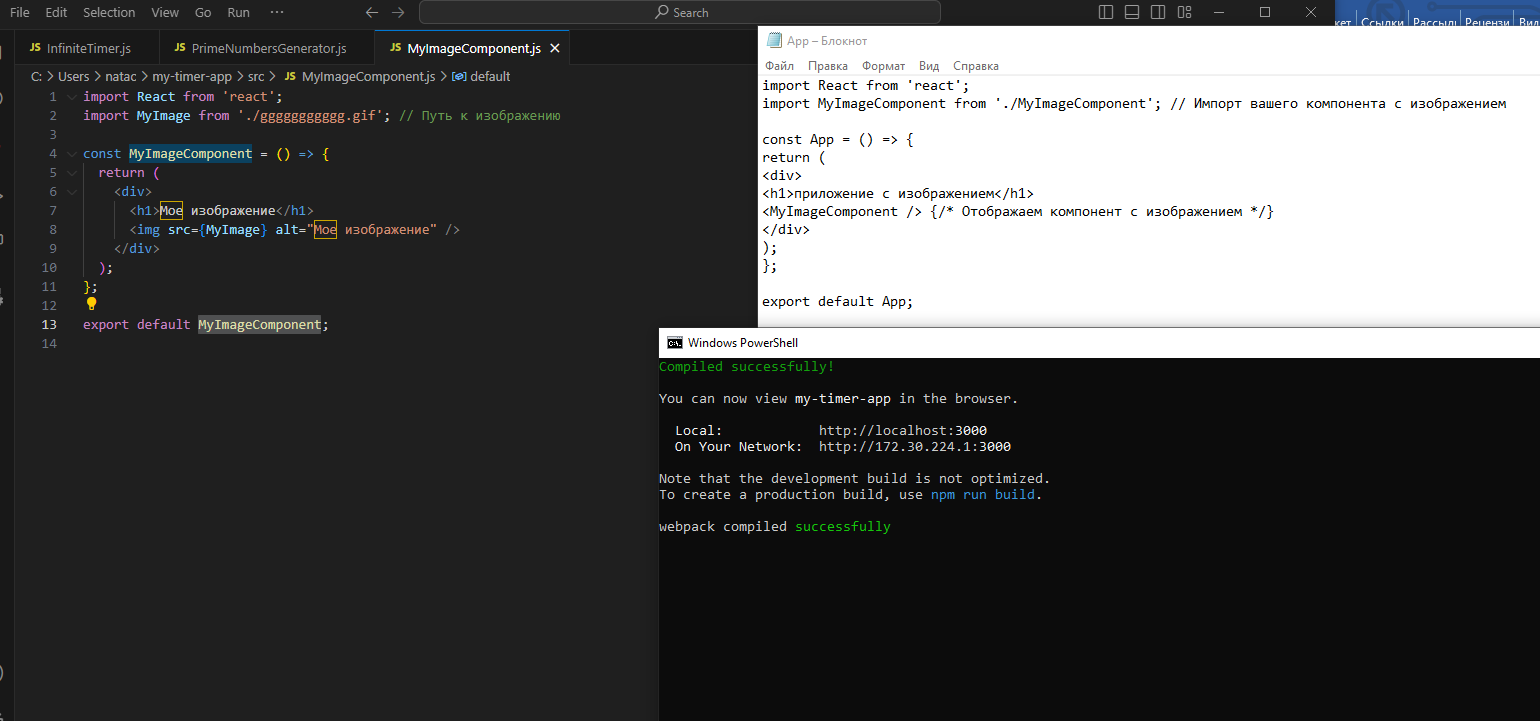
};

export default App;

**Задание 4**







**MyImageComponent.js**

import React from 'react';

import MyImage from './ggggggggggg.gif'; // Путь к изображению

const MyImageComponent = () => {

  return (

    <div>

      <h1>Мое изображение</h1>

      <img src={MyImage} alt="Мое изображение" />

    </div>

  );

};

export default MyImageComponent;

**APP.js**

import React from 'react';

import MyImageComponent from './MyImageComponent';

const App = () => {

return (

<div>

<h1>приложение с изображением</h1>

<MyImageComponent /> {/\* Отображаем компонент с изображением \*/}

</div>

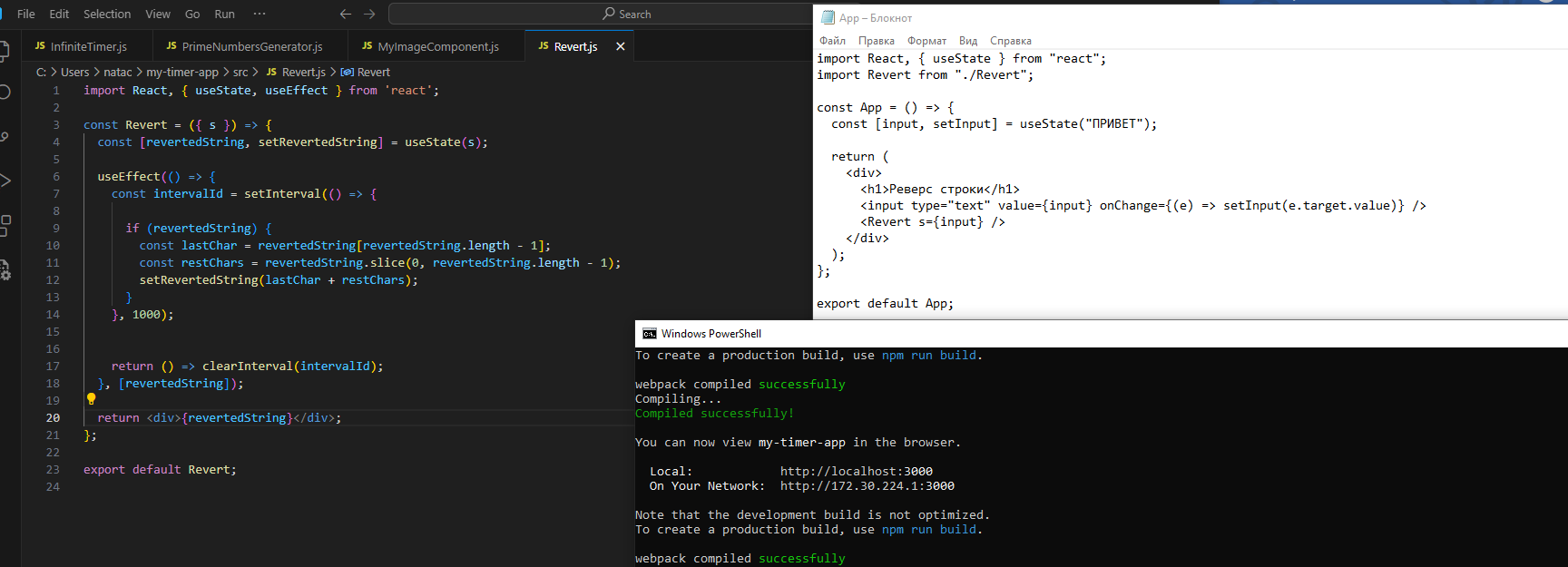
);

};

export default App;

**Задание 5**





**Revert.js**

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

const Revert = ({ s }) => {

  const [revertedString, setRevertedString] = useState(s);

  useEffect(() => {

    const intervalId = setInterval(() => {

      if (revertedString) {

        const lastChar = revertedString[revertedString.length - 1];

        const restChars = revertedString.slice(0, revertedString.length - 1);

        setRevertedString(lastChar + restChars);

      }

    }, 1000);

    return () => clearInterval(intervalId);

  }, [revertedString]);

  return <div>{revertedString}</div>;

};

export default Revert;

**APP.js**

import React, { useState } from "react";

import Revert from "./Revert";

const App = () => {

const [input, setInput] = useState("ПРИВЕТ");

return (

<div>

<h1>Реверс строки</h1>

<input type="text" value={input} onChange={(e) => setInput(e.target.value)} />

<Revert s={input} />

</div>

);

};

export default App;