**Lesson 03 Demo 01**

**React HOC Component**

**Objective:** To demonstrate the react hoc component example and use.

**Tools required:** Node JS and React JS

**Prerequisites:** HTML, CSS, JavaScript ES5/ES6, Basic React Concept

**Note** : All react js project already created with version 18.x with Sample App.js file

**Steps to be followed:**

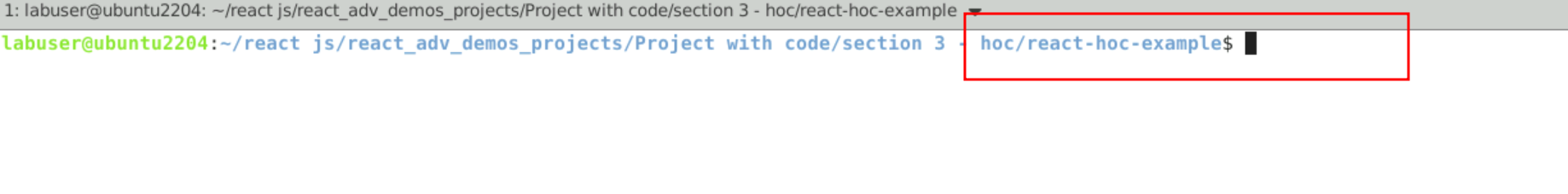
1. Set up for react js project

Create one user defined component which take props as parameter and return JSX code. This component demonstrates without HOC concept.

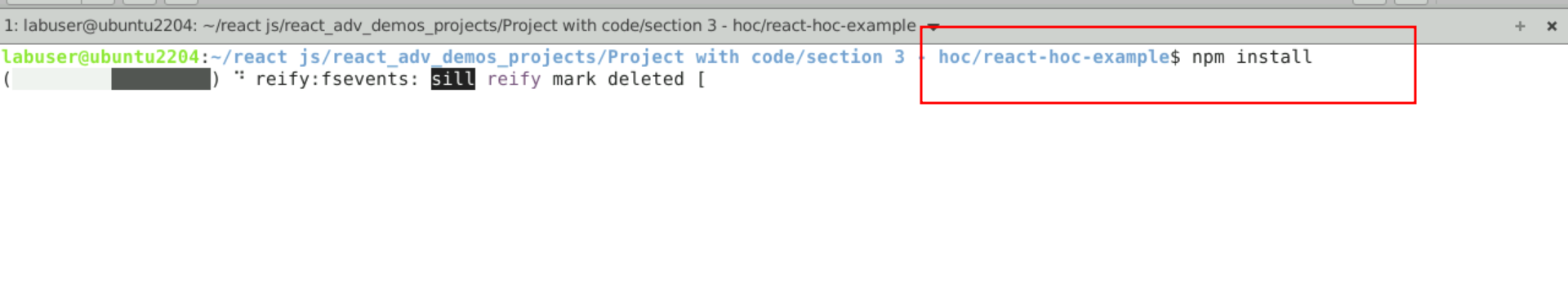
1. Now we will create HOC components and more than one user defined component which pass parameter to HOC components.
2. Now we need to import these all component in App.js file
3. Now test the application

**Step 1: Set up for react js project**

1. Open a terminal window inside a React JS pre-created project ie **react-hoc-example**

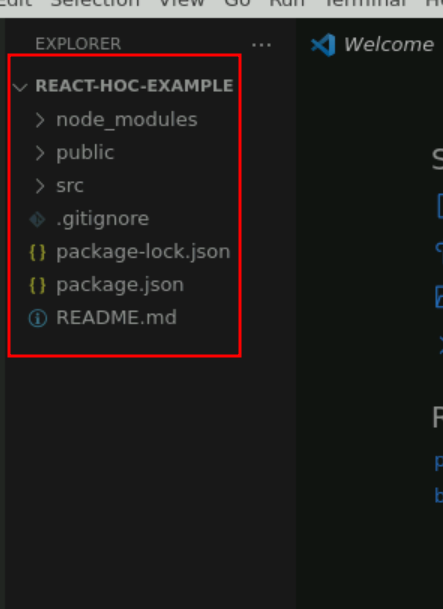


1. Now you need to run the command as **npm install.** This command helps us to installed all required dependencies mention in package.json file in local machine in the form of node\_module folder.



1. Now open **react-hoc-example** folder in VS Code Editor

Note: short cut to open write **code .**



**Step 2: Create one user defined component which take props as parameter and return JSX code. This component demonstrate without HOC concept**

2.1 Now create TextComponetWithoutHOC.js file. Which takes props as parameter and return some styling properties with JSX.

import { useState } from "react";

const TextComponentWithoutHOC = ({ text }) => {

const [isHovered, setHovered] = useState(false);

function handleMouseEnter() {

setHovered(true);

}

function handleMouseLeave() {

setHovered(false);

}

return (

<>

<p

style={{ backgroundColor: isHovered ? "orange" : "white" }}

onMouseEnter={handleMouseEnter}

onMouseLeave={handleMouseLeave}

>

{ text }

</p>

</>

);

};

export default TextComponentWithoutHOC;

2.2. Import this component in app.js file and test it.

import './App.css';

import TextComponentWithoutHOC from './TextComponentWithoutHOC';

function App() {

return (

<div className="App">

<TextComponentWithoutHOC text="This is simple text change style without HOC!"></TextComponentWithoutHOC>

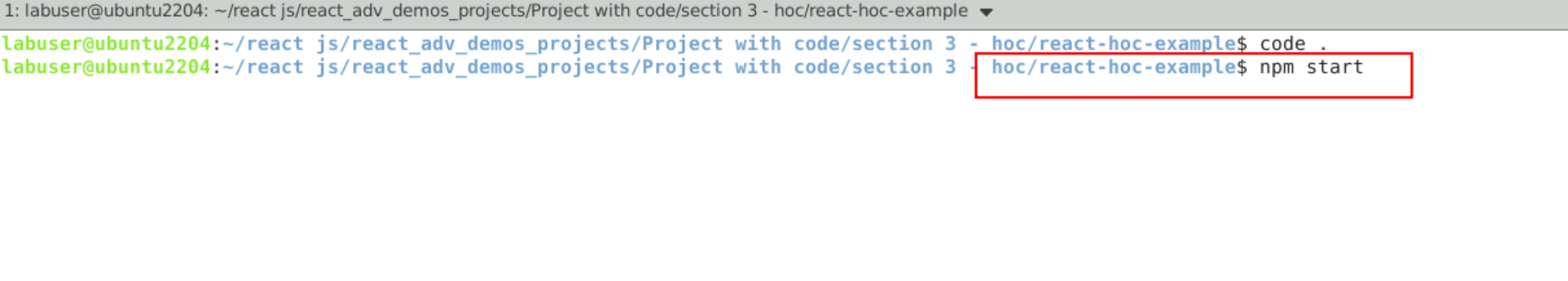
</div>

);

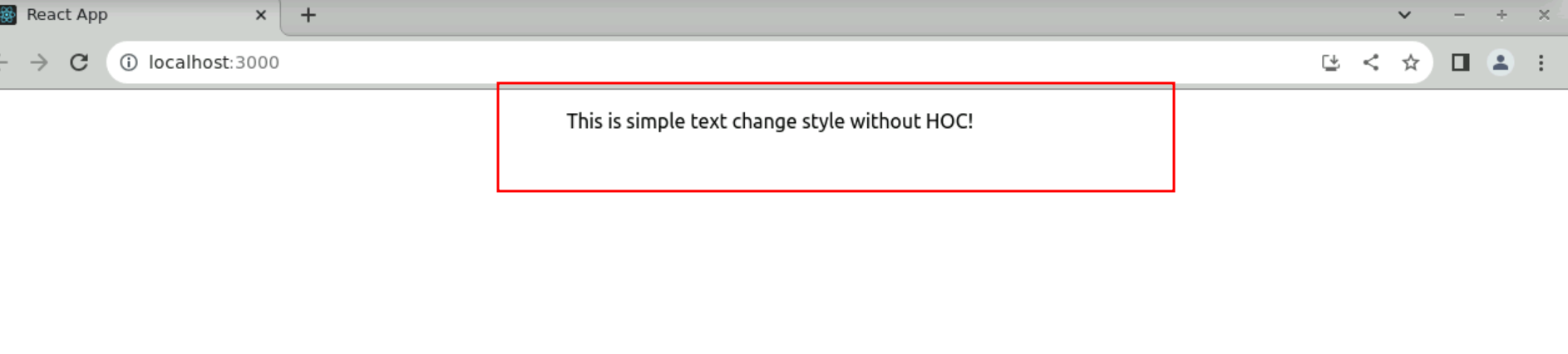
}

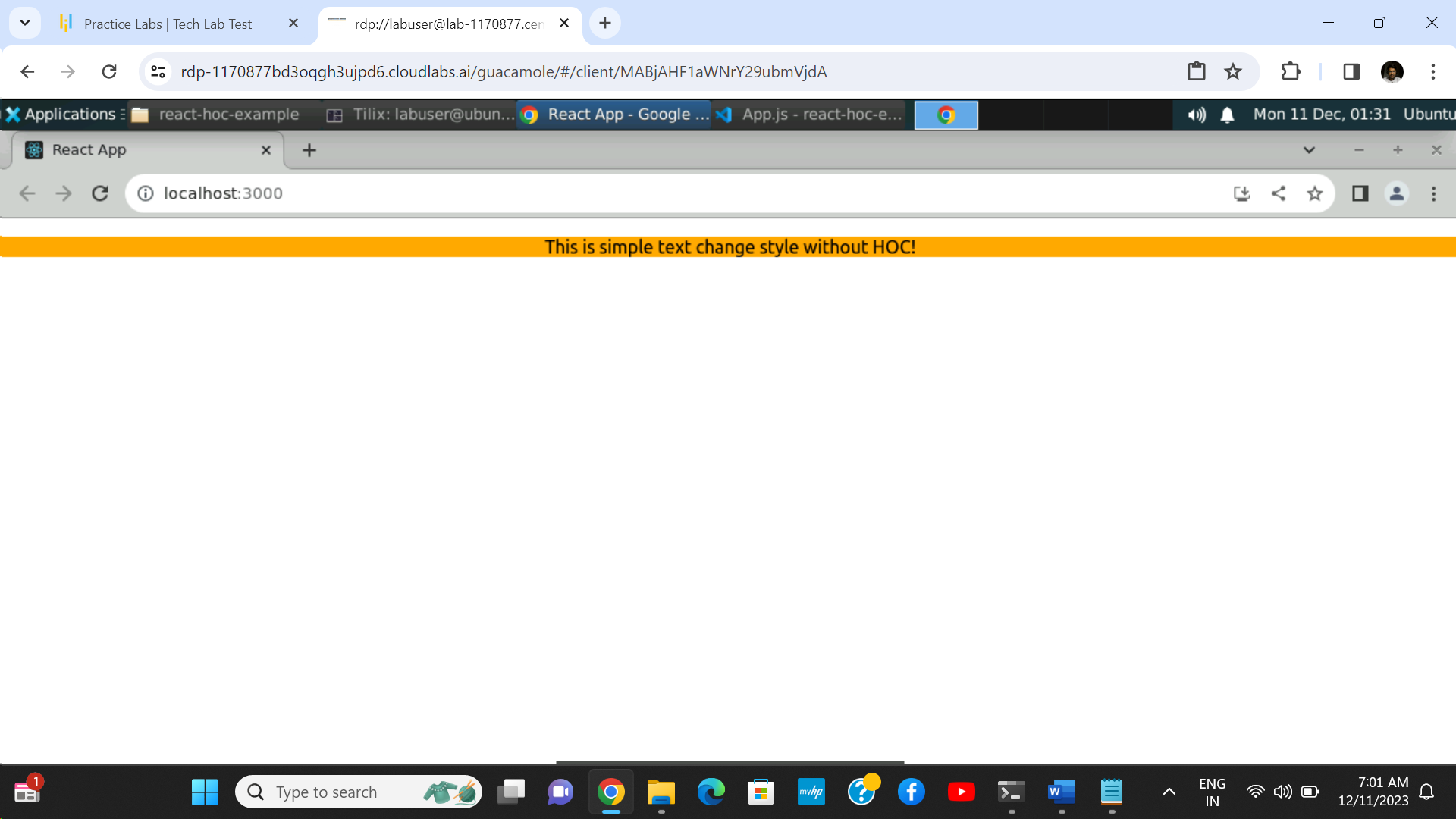
export default App;

2.3. Now run the application in terminal as **npm start.**



2.4 You can see the output. Move the cursor on text it will change the background colour and if move cursor back it change back to origin colour.





**Step 3: Now we will create HOC components and more than one user defined component which pass parameter to HOC components.**

3.1 In This file import all component as well as HOC component. For HOC component as user defined component it TextComponent and InputComponent as parameter and return new component as styling effect.

3.2. TextComponent.js file. This component takes props as text and isHovered Boolean and and return some styling effect.

const TextComponent = ({ text, isHovered }) => {

return (

<>

<p style={{ backgroundColor: isHovered ? "orange" : "white" }}>{text}</p>

</>

);

};

export default TextComponent;

3.3. InputComponent.js file. This component takes props as type and isHovered Boolean and and return some styling effect.

const InputComponent = ({ type, isHovered }) => {

return (

<input

type={type}

style={{ backgroundColor: isHovered ? "orange" : "white" }}

/>

);

};

export default InputComponent;

**Step 4: Now we will import all user defined component as well as HOC component in app.js file**

4.1 Inside this file import all user defined component ie TextComponet and ButtonComponent and pass these component as parameter to HOC component is withHoverHOC component.

import './App.css';

import InputComponent from './InputComponent';

import TextComponent from './TextComponent';

import TextComponentWithoutHOC from './TextComponentWithoutHOC';

import withHoverHOC from './withHoverHOC';

// Creating components that contain hover logic using

// Higher Order Component.

const TextComponentWithHover = withHoverHOC(TextComponent);

const InputComponentWithHover = withHoverHOC(InputComponent);

function App() {

return (

<div className="App">

{/\* <TextComponentWithoutHOC text="This is simple text change style without HOC!"></TextComponentWithoutHOC> \*/}

<TextComponentWithHover

text="This is simple text change style with HOC!"

/>

<InputComponentWithHover text="input"></InputComponentWithHover>

</div>

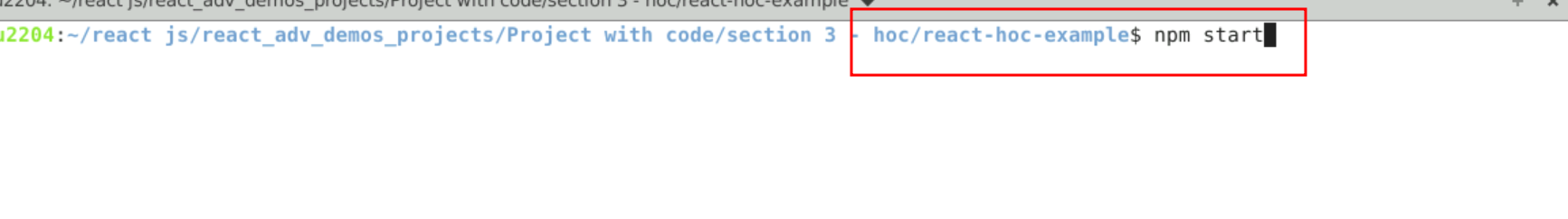
);

}

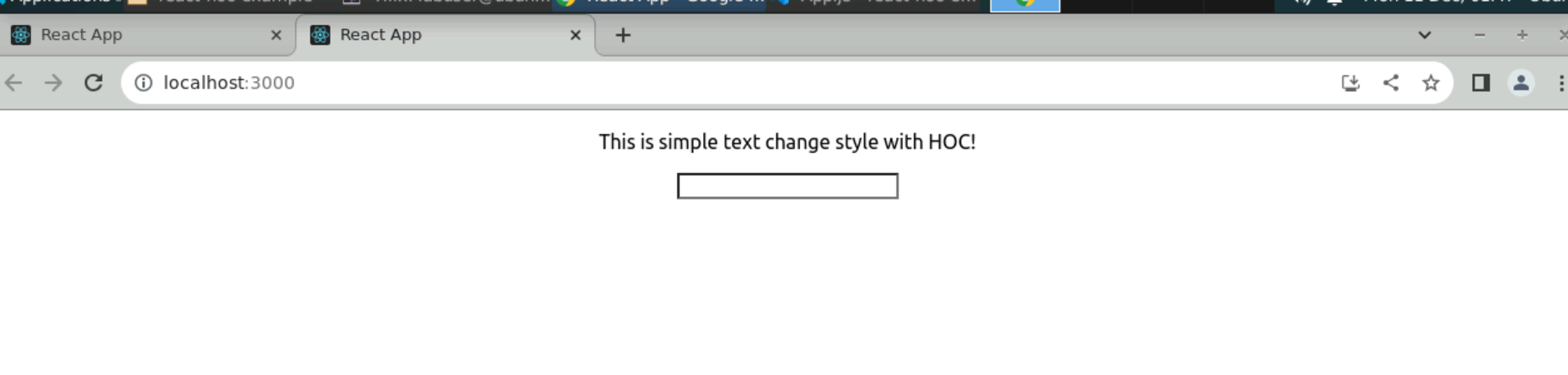
export default App;

**Step 5. Now test the application**

**5.1 npm start**



**8.2 See the output on browser**



**8.3 Now move the cursor on contents as well as text field and see the effect.**

