

Week 7

- What is Name of the Component you have created in EmojiCounters.js

EmojiCounter

- Identify the line of code that uses the EmojiCounter in index.js

```
import EmojiCounter from './EmojiCounters';
```

- Declares the states of each of the html elements defined in the EmojiCounters.js (identify these lines and explain only those lines)

- Lines of codes that are used to associate the event handler used.

```
const ClickHandle = () => {  
  setCount(count + 1);  
};
```

```
<button onClick={ClickHandle}>  
  {count}  
  <img src={pic} alt="" />  
</button>
```

Explain the line : <EmojiCounter pic = 'love'/>, what is pic = 'love' means in this line

Pic='love' is a prop passed to the EmojeeCounter component. It means that props.pic will be equal to 'Love'

The prop (pic) is used in EmojeeCounter to determine which image to display. The useEffect hook checks props.pic and sets the pic state accordingly to display the correct image

- What is useEffect and why you think we have used it in the Component.

useEffect is a react hook for handling side effects in functional components. In this component, useEffect watches for changes in props.pic and updates the pic state with the appropriate image (love.like.sad).

Explain these line of the codes in functional component EmojeeCounter.js:

```
1. return (  
2. <div className="App">  
3. <p>{props.pic}<span></span>  
4. <button onClick={ClickHandle}>{count}  
5. <img src={pic} alt=""/>  
6. </button>  
7. </p>  
8. </div>  
9. )  
10. }
```

2. Wraps the components content and applies a CSS class named App
3. Displays the pic prop label ("love" or "sad") inside a paragraph
4. Creates a button that when clicked calls 'ClickHandle' to increase 'count'
{Count} displays the current count of button clicks

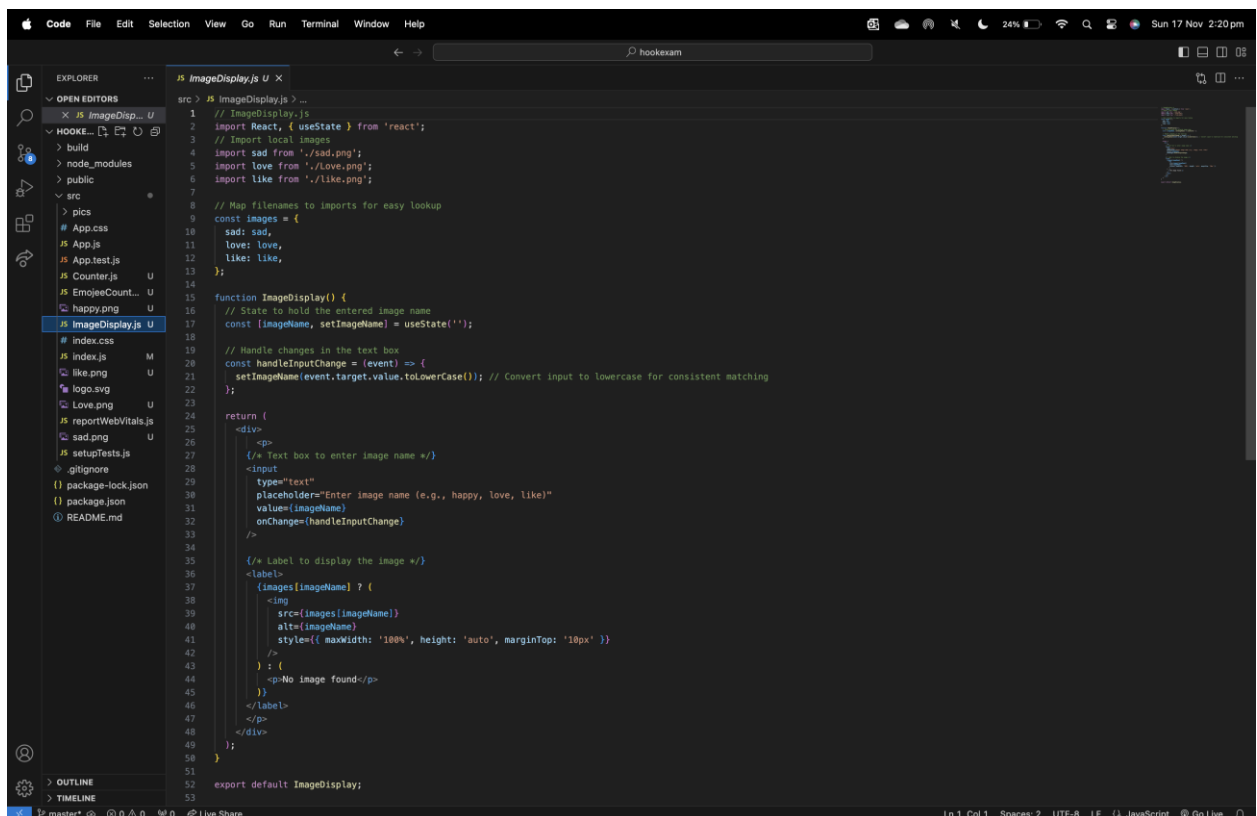
Create a code for a Component that takes two HTML one text box and one label. Label will be used to display the images.

So it should be like this If I write “Happy” in the text box the label should show happy face (You can use any image)

If I write “Like” in the text box the label should show Like icon

If I write “sad “ the label should show sad emoji.

Run this component take the screen shot of your newly run component and write a paragraph how did you develop this component

The screenshot shows a VS Code editor with a file explorer on the left and a code editor in the center. The file explorer shows a project structure with folders like 'src' and 'public', and files like 'App.js', 'App.test.js', 'Counter.js', 'EmojiCount.js', 'happy.png', 'ImageDisplay.js', 'index.css', 'index.js', 'like.png', 'logo.svg', 'Love.png', 'reportWebVitals.js', 'sad.png', 'setupTests.js', '.gitignore', 'package-lock.json', 'package.json', and 'README.md'. The code editor shows the 'ImageDisplay.js' file with the following code:

```
1 // ImageDisplay.js
2 import React, { useState } from 'react';
3 // Import local images
4 import sad from './sad.png';
5 import love from './Love.png';
6 import like from './like.png';
7
8 // Map filenames to imports for easy lookup
9 const images = {
10   sad: sad,
11   love: love,
12   like: like,
13 };
14
15 function ImageDisplay() {
16   // State to hold the entered image name
17   const [imageName, setImageName] = useState('');
18
19   // Handle changes in the text box
20   const handleInputChange = (event) => {
21     setImageName(event.target.value.toLowerCase()); // Convert input to lowercase for consistent matching
22   };
23
24   return (
25     <div>
26       <p>
27         /* Text box to enter image name */
28         <input
29           type="text"
30           placeholder="Enter image name (e.g., happy, love, like)"
31           value={imageName}
32           onChange={handleInputChange}
33         />
34       <p>
35         /* Label to display the image */
36         <label>
37           {images[imageName] ? (
38             <img
39               src={images[imageName]}
40               alt={imageName}
41               style={{ maxWidth: '100%', height: 'auto', marginTop: '10px' }}
42             />
43           ) : (
44             <p>No image found</p>
45           )}
46         </label>
47       </p>
48     </div>
49   );
50 }
51
52 export default ImageDisplay;
```

The component imageDisplay imports a useState hook for managing the components state and images for the display functionality. These images are stored in the image object, where is key represents an image name and its value in the src.

The component uses an input field for the user to input the name of the image they want to view. The input is converted into lowercase for consistency. The handleInputChange function updates the imageName state whenever the input value changes. The return statement contains the JSX structure. Including the input field and a conditional

rendering block. If the entered image name matches a key in the images object, the image is displayed otherwise a fallback message (“no image found”) is shown.