Week 7

Task 1:

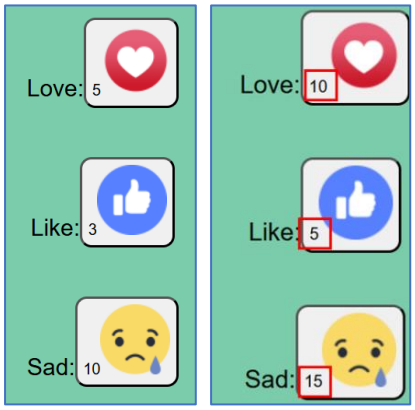
Output:

Explanation:

I used Hook API's useState method to keep track of the number of clicks. A button calls an event handler which executes the counter's function updater by an increment of one.

Task 2:

Output:



Explanation:

I used Hook API's useState and useEffect to keep track of and update both the counter's state and emoji's state. UseEffect activates whenever a change to the value of 'props.pic' happens. A conditional checks a string value and compares it to 'props.pic'. if a match is true, use the emoji's updater function to set the emoji's value. A button calls an event handler which executes the counter's function updater by an increment of one.

Question 1:

I learned that Hook API's enable state functionality within functional components. States are variables with values that are stored between function calls and are updated via updater functions. These functions are embedded within an event handler function so that they can be called when required. I worked with two methods of the Hook API: useState and useEffect. useState is used to declare and initialise a state and useEffect is used to perform a series of tasks when a particular change takes place. States are particularly important as their values either remain the same or are updated for each render/re-render.

Question 2:

What is Name of the Component you have created in EmojeeCounters.js?

Ans:

```
function EmojiCounter(props)
```

Identify the line of code that uses the EmojeeCounter in index.js:

Ans:

```
<EmojiCounter pic='Love'/>
<EmojiCounter pic='Like'/>
<EmojiCounter pic='Sad'/>
```

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

Ans:

```
const [pic, setPic] = useState(Love);
// Declare state variable and state upd
const [count, setCount] = useState(0);
```

First state has variable 'pic' and updater function 'setPic'. 'pic' is initialised as 'Love' emoji. Second state has variable 'count' and updater function 'setCount'. 'count' is initialised as 0.

Lines of codes that are used to associate the event handler used:

Ans:

Event handler function:

```
const clickHandle = () =>
{
    // Increment count by 1
    setCount(count + 1);
};
```

Called via button:

```
<button style={buttonStyle} onClick={clickHandle}
```

Explain the line : <EmojeeCounter pic='Love'/>, what does pic='Love' mean in this line?

Ans: 'pic' is a property of the component 'EmojeeCounter' and the argument 'Love' is passed as the property of said component. The Love emoji image is displayed.

What is useEffect and why do you think we have used it in a Component?

Ans: 'useEffect' is a method of the React Hook API which is used to perform various actions when states change, and adds additional functionality for the rendering of components.

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

// Returns JSX.

return (

// Set div class to App.

// Paragraph that displays the value of 'props.pic'.

<p>{props.pic}

// Button is associated with event handler 'ClickHandle'. When the button is clicked, the event handler performs the count updater function which increments the count by 1. Count is also displayed in the button tag.

<button onClick={ClickHandle}>{count}

// Image is displayed within the button tag. The source of the image is set to the value of 'pic', which has been imported into the file.


```
// Closing tags.
</button>
</p>
// End of return method.
);
```

Question 3:

Output:

Explanation:

I first imported useState, useEffect, and the three emojis into my JS file. I declared a functional component called 'TextBoxtEmoji' that is composed of state declarations, an event handler function, useEffect with an embedded switch statement, CSS object styling, and a return method. A text state is mapped to the value of the input field, and the emoji state's value is determined by the comparison between the text state and string values. The rendered text field calls the event handler function on change and thus sets in motion the chain of events explained above.