

# Week 9 Portfolio

Name: Deepak Kumar

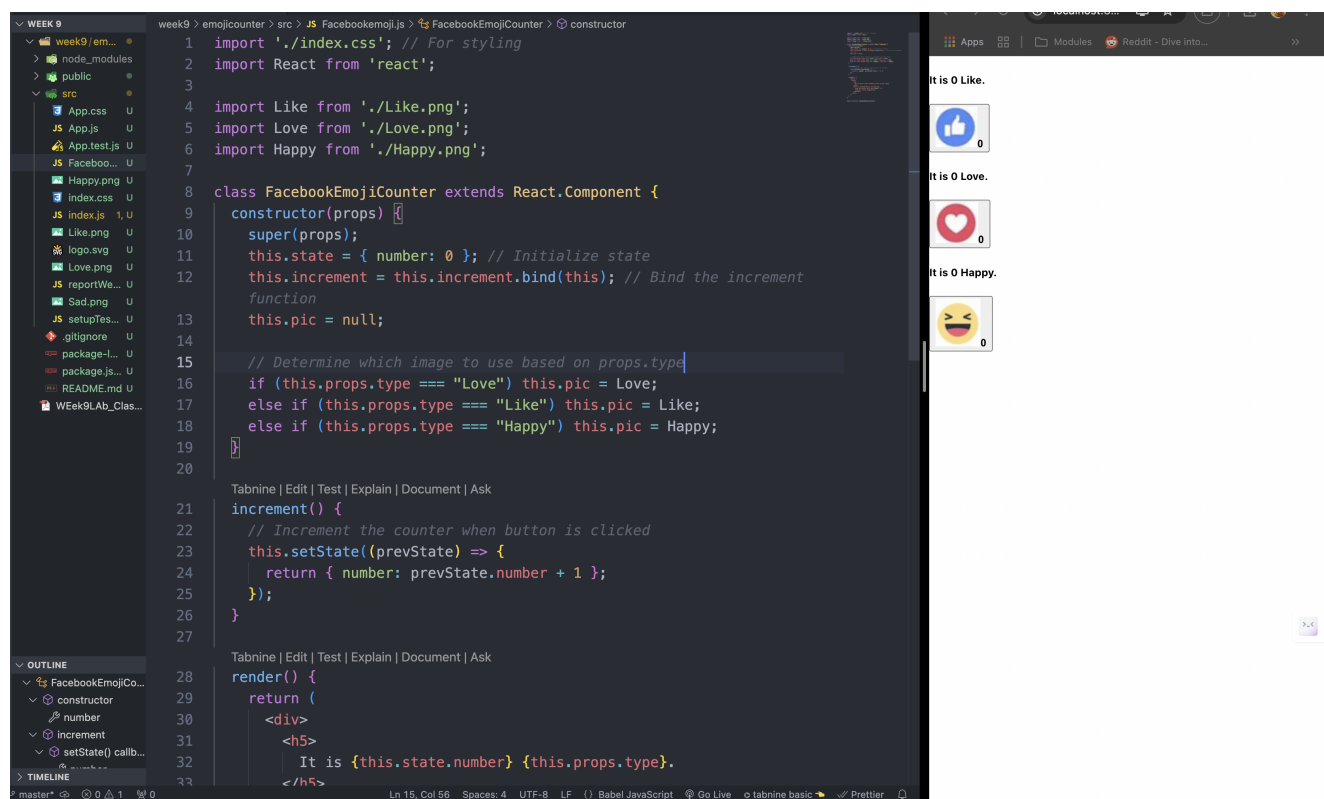
Student ID: 2284279

## React Class Components

### 1. Screenshots

#### Task 1: Facebook Emojis Class Component

In the task one, I started by creating a `Facebookemoji.js` file in my `src` folder and wrote a class component to count clicks for different emojis. I imported images and used props to display specific emojis like "Like," "Love," and "happy." Then, I made sure each button increased the counter when clicked.



```
1 import './index.css'; // For styling
2 import React from 'react';
3
4 import Like from './Like.png';
5 import Love from './Love.png';
6 import Happy from './Happy.png';
7
8 class FacebookEmojiCounter extends React.Component {
9   constructor(props) {
10     super(props);
11     this.state = { number: 0 }; // Initialize state
12     this.increment = this.increment.bind(this); // Bind the increment
13     function
14     this.pic = null;
15
16     // Determine which image to use based on props.type
17     if (this.props.type === "Love") this.pic = Love;
18     else if (this.props.type === "Like") this.pic = Like;
19     else if (this.props.type === "Happy") this.pic = Happy;
20
21   }
22
23   increment() {
24     // Increment the counter when button is clicked
25     this.setState((prevState) => {
26       return { number: prevState.number + 1 };
27     });
28   }
29
30   render() {
31     return (
32       <div>
33         <h5>
34           It is {this.state.number} {this.props.type}.
35         </h5>
36       </div>
37     );
38   }
39 }
```

The screenshot shows a code editor with the `FacebookEmojiCounter` class component. The component uses `React.Component` and manages state with `useState` (though the code uses `this.state` and `this.setState`). It imports three image files: `Like.png`, `Love.png`, and `Happy.png`. The `constructor` method initializes the state to `{ number: 0 }` and binds the `increment` method. The `render` method returns a `div` containing an `h5` element with the text "It is {this.state.number} {this.props.type}." The preview on the right shows three buttons: "Like", "Love", and "Happy", each with a corresponding emoji and a counter set to 0.

Rendering in the index.js

```

week9 > emojiCounter > src > JS index.js > ...
1  import React from 'react';
2  import ReactDOM from 'react-dom/client';
3  import './index.css';
4  import App from './App';
5  import reportWebVitals from './reportWebVitals';
6  import FacebookEmojiCounter from './Facebookemoji';
7
8  const root = ReactDOM.createRoot(document.getElementById
  ('root'));
9  root.render(
10     <React.StrictMode>
11       <FacebookEmojiCounter type="Like" />
12       <FacebookEmojiCounter type="Love" />
13       ⚠ <FacebookEmojiCounter type="Happy" />
14     </React.StrictMode>
15   );
16
17   // If you want to start measuring performance in your app,
18   // pass a function
19   // to log results (for example: reportWebVitals(console.log))
20   // or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
21   reportWebVitals();

```

## Task 2: Toggle Emoji

For this, I created a `ToggleModeComponent.js` file. I made a class that toggles between "happy" and "sad" emojis each time a button is clicked. I used the component's state to track which emoji to show and added the toggle logic inside a function.

```

1 import './index.css';
2 import React from 'react';
3
4 import Sad from './Sad.png';
5 import Happy from './Happy.png';
6
7 class ToggleMode extends React.Component {
8   constructor(props) {
9     super(props);
10    this.state = { pic: Happy }; // Initialize with happy emoji
11    this.Toggle_Mode = this.Toggle_Mode.bind(this); // Bind the toggle function
12  }
13
14  Toggle_Mode() {
15    // Toggle between sad and happy images
16    this.setState((prevState) => {
17      if (prevState.pic === Sad) {
18        return { pic: Happy };
19      } else {
20        return { pic: Sad };
21      }
22    });
23  }
24
25  render() {
26    return (
27      <div>
28        <h3>Toggle the Emoji!</h3>
29        <button onClick={this.Toggle_Mode}>
30          <img src={this.state.pic} alt="emoji" />
31        </button>
32      </div>
33    );
34  }
35 }
36
37 export default ToggleMode;
38

```

Then, I included this component in `index.js` and tested it to make sure it worked.

```
week9 > emojiCounter > src > JS ToggleModeComponent.js > ToggleMode > Toggle_Mode > setState() callback
1 import './index.css';
2 import React from 'react';
3
4 import Sad from './Sad.png';
5 import Happy from './Happy.png';
6
7 class ToggleMode extends React.Component {
8   constructor(props) {
9     super(props);
10    this.state = { pic: Happy }; // Initialize with
    happy emoji
11    this.Toggle_Mode = this.Toggle_Mode.bind
    (this); // Bind the toggle function
12  }
13
14  Toggle_Mode() {
15    // Toggle between sad and happy images
16    this.setState((prevState) => {
17      if (prevState.pic === Sad) {
18        return { pic: Happy };
19      } else {
20        return { pic: Sad };
21      }
22    });
23  }
24
25  render() {
26    return (
27      <div>
28        <h3>Toggle the Emoji!</h3>
29        <button onClick={this.Toggle_Mode}>
30          <img src={this.state.pic} alt="emoji" />
31        </button>
32      </div>
33    );
34  }
35}
```

Tabnine | Edit | Test | Explain | Document | Ask

It is 0 Like.

It is 0 Love.

It is 6 Happy.

Toggle the Emoji!

Ln 21, Col 8 Spaces: 4 UTF-8 LF Babel JavaScript Go Live tabnine basic Prettier

## 2. Reflection

This lab was super interesting because I learned how to use **class components** in React. They're different from the functional components we used before because they have a constructor and can hold their own state. This means the component can "remember" things, like how many times I clicked a button or which emoji to show. At first, it felt a bit tricky since I had to bind methods and use `this.setState`, but once I understood it, it made sense.

Writing the code for `increment` and `toggle` functions really helped me understand how buttons and events work better. Compared to last week's lab with functional components, which were easier since they used hooks like `useState`, class components felt more advanced but also more powerful. I feel like I'm learning how React works on a deeper level now!