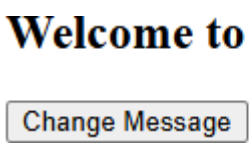
Activity 1 – MessageComponent.js

```
import React, { useState } from "react";

function MessageComponent() { const
    [message, setMessage]
= useState("Hello Student");

    return (
        <div>
            <h2>{message}</h2>
            <button        onClick={()        =>
setMessage("Welcome to React State")}>
                Change Message
            </button>
        </div>
    );
}
export default MessageComponent;
```

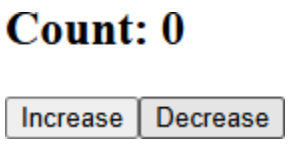
Output:



Activity 2 – Counter.js

```
import React, { useState } from "react";

function Counter() {
    const [count, setCount] = useState(0);

    return (
        <div>
            <h2>Count: {count}</h2>
            <button onClick={() =>
setCount(count + 1)}>Increase</button>
            <button onClick={() =>
setCount(count - 1)}>Decrease</button>
        </div>
    );
}
export default Counter;
```

Output:



Activity 3 – TextInput.js import React, { useState } from "react";

```
function TextInput() { const [text,
    setText] =
useState("");

    return (
        <div>
            <input
                type="text" value={text}
                onChange={(e) =>
setText(e.target.value)}
                placeholder="Enter your
name"
```

```
            />
            <p>You entered:
        {text}</p>
            </div>
        );
}
export default TextInput;
```

Output:

You entered: z4

Activity 4 – UserDetails.js import React, { useState } from "react";

```
function UserDetails() { const [name,
    setName] =
useState("");
    const [city, setCity] = useState("");

    return (
        <div>
            <input
                type="text"
                placeholder="Enter
Name"
                value={name}
                onChange={(e)=>
setName(e.target.value)}
            />

            <input
                type="text"
                placeholder="Enter
City"
                value={city} onChange={(e)
                =>
setCity(e.target.value)}
            />

            <p>Name: {name}</p>
            <p>City: {city}</p>
        </div>
    );
}
export default UserDetails;
```

Output:

Name: z4

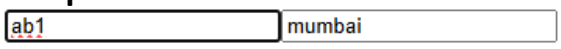
City: mumbai

Activity 5 – UserDetailsObject.js

```
import React, { useState } from "react";

function UserDetailsObject()
{
    const [user, setUser] = useState({
        name: "",
        city: ""
    });

    const handleChange = (e) =>
{
        const { name, value } =
e.target;

        setUser({
            ...user, [name]:value
        });

    };

    return (
        <div>
            <input
                type="text" name="name"
                placeholder="Enter
Name"
                value={user.name}

onChange={handleChange}
            />

            <input
                type="text"
```

```
                name="city"
                placeholder="Enter
City"
                value={user.city}

onChange={handleChange}
            />

            <p>Name:
{user.name}</p>
            <p>City:
{user.city}</p>
        </div>
    );
}
export default
UserDetailsObject;
```

Output:



Name: ab1

City: mumbai

Activity 6 – EmailValidation.js

```
import React, { useState } from
"react";

function EmailValidation() { const [email,
    setEmail] =
useState("");
    const [message, setMessage]
= useState("");

    const validateEmail = () =>
{
        if (email === "") { setMessage("Email
            field
cannot be empty.");
        } else if (!email.includes("@")){
            setMessage("Invalid email. Must
contain '@'.");
        } else { setMessage("Email is
    valid!");
        }
    };

    return (
        <div>
            <input
                type="text" placeholder="Enter

Email"        value={email} onChange={(e)=>


setEmail(e.target.value)}
            />
            <button
onClick={validateEmail}>Valid ate</button>
            <p>{message}</p>
        </div>

    );
}
export default EmailValidation;
```

Output:

Email is valid!

Invalid email. Must contain '@'.

1. What is state in React?

State in React is a built-in object that stores data or information about a component that can change over time.

When state changes, React automatically updates (re-renders) the component to reflect the new data in the UI.

In functional components, state is typically managed using the useState hook.

2. What is the difference between a normal variable and a state variable?

Normal Variable	State Variable
Does not trigger re-render when changed	Triggers re-render when updated
Value resets on every render	Value persists between renders
Cannot update UI automatically	Updates UI automatically
Declared normally (e.g., let count = 0)	Created using useState()

---

3. What is a controlled component?
A controlled component is a form element (like <input>, <textarea>, or <select>) whose value is controlled by React state.
Example concept:

- **The input value comes from state.**
- **When the user types, an onChange event updates the state.**
- **The state controls what is displayed in the input.**

So React becomes the single source of truth for the form data.

---

4. Why does React re-render when state changes?
React re-renders when state changes because:

- **React tracks state internally.**
- **When setState (or useState setter) is called, React knows the component's data changed.**
- **React compares the new UI with the previous one using its Virtual DOM.**
- **It updates only the parts of the real DOM that changed.**

This ensures the UI always reflects the latest state.

---

5. Why do we use the spread operator in object state?
The spread operator (...) is used to preserve existing properties when updating an object in state.
React state updates must be immutable (we should not directly modify existing state).
Example concept:

- **If you update one property without spreading the old object, you will lose the other properties.**
- **The spread operator copies the existing properties before updating specific ones.**

This prevents accidental data loss and keeps state updates predictable.