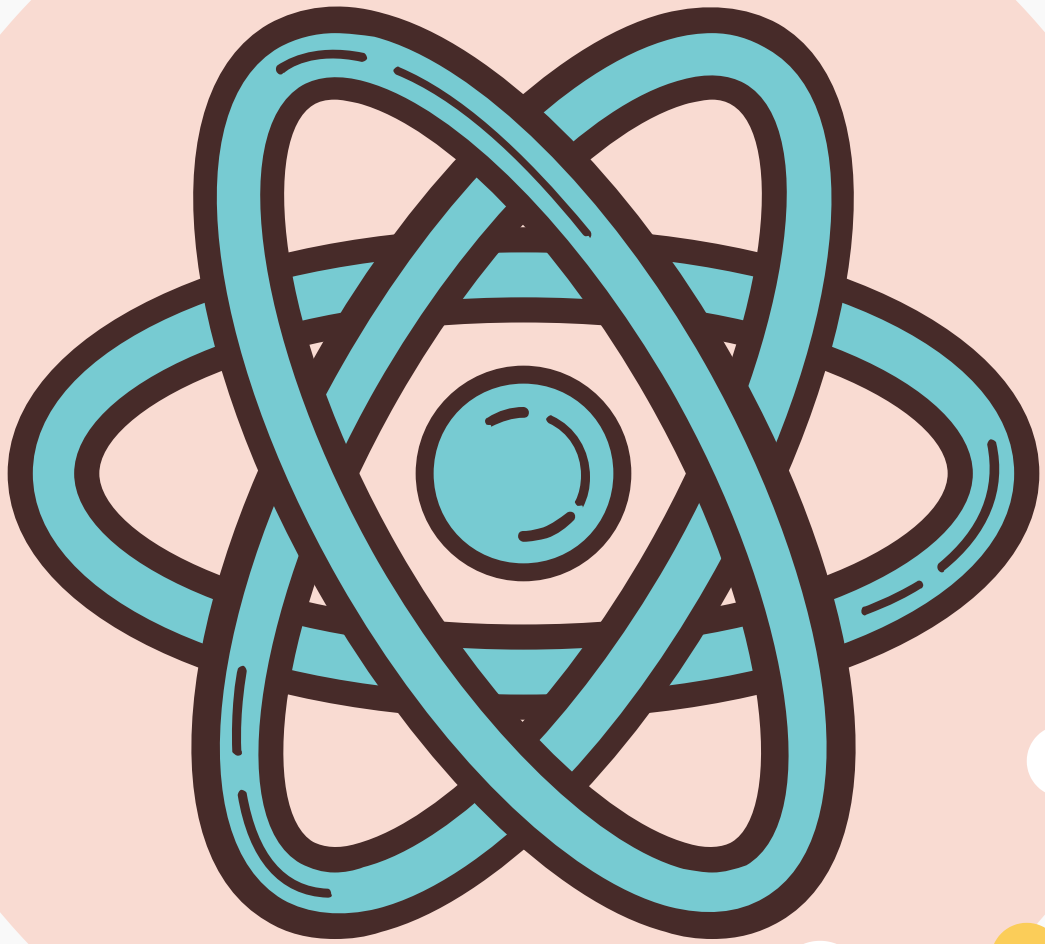
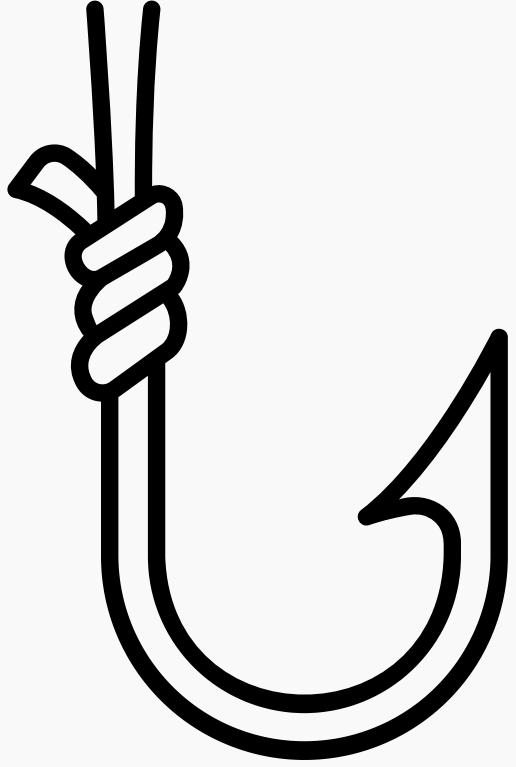
MUKUL
RAJPOOT



SOME
REACT.JS
CONCEPTS



THE useEffect HOOK

LIFECYCLE

The lifecycle of a component is just what it sounds like: it describes how a component evolves. Like us, components are born, do some things during their time here on earth, and then they die.

React's useEffect hook combines componentDidMount, componentDidUpdate and componentWillUnmount lifecycle methods.

COMPONENTDIDMOUNT EQUIVALENT

In order to have this hook run only once (when a component is mounted), we need to set an empty array as a hook dependency.

```
useEffect(() => {
  /* ComponentDidMount Logic */
}, [])
```

COMPONENTDIDUPDATE EQUIVALENT

In order to have this hook run when the component is updated (this includes mounting), we need to set at least one variable as hook's dependency (in this case, var1 and var2).

```
useEffect(() => {
  /* componentDidUpdate Logic */
}, [var1, var2]);
```

COMPONENTWILL UNMOUNT EQUIVALENT

In order to have this hook run when the component is unmounted, we need to return a function from the hook. If we want cleanup function to run only when component has unmounted, we need to set an empty array. If we set one or more variables in the dependency array, cleanup will run at every re-render.

```
useEffect(() => {
   return () => {
     /* componentWillUnmount Logic */
   }
}, []);
```

COMPLETE HOOK ALL THREE COMBINED

```
useEffect(() => {

  /* componentDidMount Logic +
componentDidUpdate Logic */

  return () => {
   /* componentWillUnmount Logic */
  }
}, [var1, var2]);
```

THANKYOU FOLLOW

MUKUL RAJPOOT