Miracle
Pg.No :
Date: / /20

Day-5

☆ Working with React hooks

→ Hooks are Javascript functions that manage the state behavior and side effects by isolating them from a component.

→ We can isolate all the stateful logic in hooks and use them into the components.

☆ useState hook

(From Scrimba notes)

☆ useEffect hook

⟹ What are side effects?

→ A side effect is something that makes a function impure.

→ Pure function	Impure function
• No side effects	• Has side effects
• They receive an input and produce a predictable output of JSX.	• Not predictable because they are actions which are performed with the "outside world."

- Always return the same output (given the same input)
- Predictable, reliable and easy to test.

- Side effects include:
  1) invoke console.log
  2) invoke fetch
  3) invoke geolocation,
  4) interact with browsers API
  5) Request to an API

etc.

→ In simple words,

Side effect is something external or outside of a function.

→ Why useEffect() ?

It provides a way to handle side effects.

useEffect is a tool that lets us interact with the outside world but not affect the rendering or performance of the component that it's in.

→ Basic syntax:

```
useEffect( () => {
    :
}, [])
```

Runs after comp. renders ↑ — ↑ on which side effect relies upon

Two arguments : a function and an array.

callback func. — dependencies array

Miracle
Pg.No :
Date: / /20

Agar array ni value change thai hoi, it will execute the useEffect function again.

→ If we do not provide the dependencies array, it will re-render everytime. This can lead to an infinite loop.

→ What is cleanup function in useEffect()?

Some side effects may need to clean up resources or memory that is not required anymore, avoiding any memory leaks that could slow down your applications.

e.g. Setting up subscription to an external data source. In that scenario it is imp. to perform a cleanup after the effect finishes its execution.

The cleanup function will be called when the component is unmounted.

going to new page / new route

```
useEffect(() => {
    let interval = setInterval(() => setTime(1), 1000)
    return () => {
        clearInterval(interval)
    }
}, [])
```