**1.       React Hooks là gì? Tại sao phải cần dùng hooks**

- React Hooks là một tính năng được giới thiệu trong React 16.8, cho phép bạn sử dụng state và các tính năng khác của React mà không cần phải viết một class. Các Hook phổ biến nhất là useState, useEffect, và useContext.

**Tại sao cần dùng Hooks:**

1. Code dễ hiểu hơn và ngắn gọn hơn
2. Tái sử dụng logic một cách hiệu quả
3. Tránh phải xử lý this
4. Quản lý state và side effects dễ dàng hơn
5. Tích hợp tính năng mới mà không phải thay đổi nhiều code

Trước đây (Class Component) :



Với Hooks (Function Component) :



1. **useState hook khi nào dùng? Cách triển khai useState hook**

- useState là một hook cơ bản trong React được sử dụng để quản lý state trong các function components. Nó cho phép bạn thêm state vào các component mà trước đây chỉ có sẵn trong các class component. Dưới đây là các tình huống và cách triển khai useState:

**Khi nào sử dụng useState**

1. **Khi cần lưu trữ một giá trị thay đổi theo thời gian:** Ví dụ như bộ đếm số lần nhấp, đầu vào của người dùng, trạng thái mở hoặc đóng của một modal, v.v.
2. **Khi một giá trị cụ thể cần gây ra việc re-render lại component:** useState giúp đảm bảo component re-render mỗi khi state thay đổi.
3. **Khi bạn muốn triển khai logic state đơn giản:** useState phù hợp với các state đơn giản và có thể mở rộng cho các state phức tạp hơn nhưng chỉ tới một mức độ nhất định (với state phức tạp hơn, bạn có thể cần useReducer).

**Cách triển khai useState**



* *state*: Giá trị hiện tại của state.
* *setState*: Hàm để cập nhật state tương ứng.
* *initialState*: Giá trị ban đầu của state. Nó có thể là một giá trị bất kì và chỉ được sử dụng trong lần render đầu tiên.

1. **useEffect hook khi nào dùng ? Cách triển khai useEffect hook**

- useEffect là một hook trong React được sử dụng để xử lý các side effects trong function components. Side effects có thể bao gồm việc thực hiện gọi API, đăng ký hoặc hủy đăng ký event listeners, hoặc thao tác DOM trực tiếp. Hook này cho phép bạn thực hiện một số hoạt động khi component được render và cơ hội để dọn dẹp (cleanup) để tránh rò rỉ bộ nhớ hoặc lỗi logic.

**Khi nào sử dụng useEffect**

* **Gọi API hoặc thực hiện fetch dữ liệu**: Khi bạn cần thực hiện các yêu cầu dữ liệu qua mạng.
* **Tương tác trực tiếp với DOM:** Ví dụ như cài đặt trình nghe sự kiện không hỗ trợ bởi React trực tiếp, hoặc thao tác DOM không đồng bộ.
* **Thiết lập bộ định thời gian (setTimeout, setInterval):** Khi bạn cần thực hiện các hành động sau một khoảng thời gian nhất định và dọn dẹp chúng khi component unmount.
* **Dọn dẹp mã hoặc nghe sự kiện:** Ví dụ như hủy đăng ký các sự kiện hoặc làm sạch bộ nhớ để tránh rò rỉ.
* **Đồng bộ hóa với một số giá trị ngoại vi:** Ví dụ như làm việc với LocalStorage, cập nhật title của trang, hoặc đồng bộ hóa khi value nào đó thay đổi.

**Cách triển khai useEffect**

- useEffect nhận hai đối số:

**Effect function :** Hàm sẽ chứa logic cần thực hiện khi component render hoặc khi một trong các dependencies thay đổi.

**Dependency array (tùy chọn) :** Một mảng các giá trị mà useEffect phụ thuộc vào. Effect sẽ chỉ chạy lại khi một trong những giá trị này thay đổi. Nếu không cung cấp, effect sẽ chạy sau mỗi lần render.

