**Side effect**

**التعريف البسيط**

* **"Side effect" تأثير جانبي**هو أي شيء يعمله الكود **غير** إرجاع نتيجة من الدالة.
* يعني: الدالة مش بس تحسب وترجع قيمة، كمان بتغير حالة مش الها (state) أو تعمل فعل يُلاحظ من خارجها.

**🟢 أمثلة بدون Side Effects**

function add(a, b) {

return a + b;

}

* هنا الدالة **ترجع قيمة فقط**.
* ما غيّرت أي شيء في الخارج → ما في **side effect**.

**🔴 أمثلة مع Side Effects**

**1. تغيير متغير خارجي**

let count = 0;

function increment() {

count++; // ← هذا Side Effect لأنه غيّر state خارج الدالة

}

**2. طباعة على الشاشة**

function logMessage(message) {

console.log(message); // ← الطباعة نفسها Side Effect

}

**3. استدعاء API (غير متزامن)**

async function getData() {

const response = await fetch("https://api.example.com/data");

return response.json(); // ← استدعاء API يعتبر Side Effect

}

**📌 الخلاصة**

* **Pure Function(component) (دالة نقية)** → ترجع نفس النتيجة لنفس المدخلات، وما تغير أي شيء خارجها (بدون side effects).
* **Impure Function (impure compaonent)(دالة غير نقية)** → تعمل أشياء جانبية مثل: تغيير state، طباعة، تخزين بيانات، API calls.

✨ في React مثلًا:

* حساب قيمة **بدون Side Effects** يكون داخل render.
* أما **Side Effects** (زي جلب بيانات من API أو تعديل DOM أو التعامل مع localStorage) لازم تنحط داخل useEffect.

في **React**، الكومبوننت لازم بالأساس تكون **pure function**:

* يعني تاخذ **props/state** → وترجع **UI (JSX)**.
* وما تعمل أي شيء إضافي (زي الطباعة في console، أو تعديل DOM، أو استدعاء API).

لكن في الواقع، كتير أحيانًا بنحتاج نعمل وظائف "خارج إطار" مجرد العرض. وهون بييجي مفهوم **side effects**.

**✅ أمثلة على الـ Side Effects:**

1. **جلب بيانات (API Call)**
   * لما الكومبوننت ينعرض أول مرة، ممكن بدنا نجيب بيانات من السيرفر.
   * هذا تغيير خارجي (مش بس عرض بيانات).
2. **تغيير في الـ DOM بشكل مباشر**
   * مثلاً تغيير عنوان الصفحة document.title = "My App".
   * هذا خارج "الرسم العادي للـ UI".
3. **التخزين في Local Storage**
   * مثال: حفظ theme المستخدم (dark/light) في localStorage.
4. **Timers**
   * تشغيل setInterval أو setTimeout.

**🎯 باختصار**

* **Side effect** = أي عملية بتأثر على "العالم الخارجي" أو حالة غير مرتبطة مباشرة بالـ render العادي.
* **React** بيوفرنا useEffect عشان نكتب هاي الـ side effects بشكل منظم (زي جلب بيانات أو تغيير DOM أو حفظ بالـ localStorage).

**1️⃣ الكومبوننت تغيّر state تبعتها نفسها باستخدام useState**

مثال:

import { useState } from "react";

function Counter() {

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

return (

<div>

<p>Count: {count}</p>

<button onClick={() => setCount(count + 1)}>+1</button>

</div>

);

}

🔹 هون لما المستخدم يضغط على الزر، الكومبوننت تغيّر count (state داخلية).  
هذا **مش side effect** 🚫، لأنه مجرد تحديث لحالة داخليّة يتحكم فيها React.

**2️⃣ الكومبوننت تغيّر شيء خارجي عن نفسها**

مثال:

import { useState, useEffect } from "react";

function Counter() {

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

useEffect(() => {

document.title = `You clicked ${count} times`;

}, [count]);

return (

<div>

<p>Count: {count}</p>

<button onClick={() => setCount(count + 1)}>+1</button>

</div>

);

}

🔹 هون صحيح إحنا غيّرنا **state**، بس كمان عملنا **side effect**:  
غيّرنا document.title (شيء خارج الكومبوننت → DOM عالمي).

**✨ الخلاصة**

* تغيير **state داخل الكومبوننت نفسها** = طبيعي، مش side effect.
* تغيير **أي شيء خارجي** (DOM، API، localStorage...) = side effect.