**Example 1:**

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

export default function UseEffect1() {  
  // useState  
  const [count, setCount] = useState(0);  
  console.log("component rendered");  
  const [name, setName] = useState("");  
  // useEffect  
  useEffect(() => {  
    /\* fetch of data from the backend \*/  
    console.log("useEffect called..");  
    document.title = `You clicked ${count} times`;  
  }, [count]);

  return (  
<div>  
<p>You clicked {count} times</p>  
<button onClick={() => setCount(count + 1)}>Click me</button>  
<input  
        type="text"  
        value={name}  
        onChange={(event) => {  
          setName(event.target.value);  
        }}  
      />  
</div>  
  );  
}

**Example 2:**

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

 function UseEffect2() {  
  const [x, setX] = useState(0);  
  const [y, setY] = useState(0);

  function logMouseHandler(e) {  
    console.log("logMouseHandler called..");  
    setX(e.clientX);  
    setY(e.clientY);  
  }

   // useEfect with Only once call behaviour  
  useEffect(() => {  
    console.log("useEffect called");  
    window.addEventListener("mousemove", logMouseHandler);  
  }, []);

  return (  
<div>  
<h1> X = {x} </h1>  
<h1> Y = {y} </h1>  
</div>  
  );  
}

export default UseEffect2;

**Example 3:**

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

const App = () => {  
  const [posts, setPosts] = useState([]);       // Store the fetched posts  
  const [loading, setLoading] = useState(true); // Show loading indicator  
  const [error, setError] = useState(null);     // Store any error message

  useEffect(() => {  
    // Fetching posts using .then() and Promises  
    fetch("[https://jsonplaceholder.typicode.com/posts")](https://jsonplaceholder.typicode.com/posts%22))  
      .then((response) => {  
        if (!response.ok) {  
          throw new Error(`HTTP error! Status: ${response.status}`);  
        }  
        return response.json();  
      })  
      .then((data) => {  
        setPosts(data);  
        setLoading(false);  
      })  
      .catch((err) => {  
        setError(err.message);  
        setLoading(false);  
      });  
  }, []); // Empty dependency array = run once on mount

  if (loading) {  
    return <p>Loading posts...</p>;  
  }

  if (error) {  
    return <p>Error: {error}</p>;  
  }

  return (  
<div style={{ padding: "20px" }}>  
<h1>Fetched Posts</h1>  
<ul>  
        {posts.map((post) => (  
<li key={post.id}>  
<strong>{post.title}</strong>  
<p>{post.body}</p>  
</li>  
        ))}  
</ul>  
</div>  
  );  
};

**Example 4:**

export default App;

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

const App = () => {  
  const [posts, setPosts] = useState([]);       // Stores fetched posts  
  const [loading, setLoading] = useState(true); // Tracks loading status  
  const [error, setError] = useState(null);     // Tracks errors

  useEffect(() => {  
    // Define async function inside useEffect  
    const fetchPosts = async () => {  
      try {  
        const response = await fetch("[https://jsonplaceholder.typicode.com/posts");](https://jsonplaceholder.typicode.com/posts%22);)  
        if (!response.ok) {  
          throw new Error(`HTTP error! Status: ${response.status}`);  
        }  
        const data = await response.json();  
        setPosts(data);  
        setLoading(false);  
      } catch (err) {  
        setError(err.message);  
        setLoading(false);  
      }  
    };

    fetchPosts(); // Call the async function  
  }, []); // Empty dependency array ensures this runs once after initial render

  if (loading) {  
    return <p>Loading posts...</p>;  
  }

  if (error) {  
    return <p>Error: {error}</p>;  
  }

  return (  
<div style={{ padding: "20px" }}>  
<h1>Fetched Posts</h1>  
<ul>  
        {posts.slice(0, 10).map(post => (  
<li key={post.id}>  
<strong>{post.title}</strong>  
<p>{post.body}</p>  
</li>  
        ))}  
</ul>  
</div>  
  );  
};