**¿Por qué evitar los callbacks en las llamadas asíncronas?**

Para evitar una sobrecarga de código y una anidación alta, cuando se puede ejecutar de manera más directa.

**¿Qué es una promesa?**

Una Promesa es un objeto en JavaScript que representa un valor que puede estar disponible ahora,

en el futuro o nunca. Proporcionan una forma más estructurada y legible de trabajar con operaciones

asincrónicas. Una Promesa tiene tres estados posibles: pendiente (pending), cumplida (fulfilled) o rechazada

(rejected). Una vez que una Promesa se cumple o se rechaza, se ejecuta el código asociado a través de los

métodos .then() o .catch() respectivamente.

function obtenerDatos() {

return new Promise(function(resolve, reject) {

setTimeout(function() {

const datos = "Estos son los datos";

resolve(datos);

}, 2000);

});

}

function procesarDatos(datos) {

console.log("Procesando los datos: " + datos);

}

obtenerDatos()

.then(procesarDatos)

.catch(function(error) {

console.log("Error: " + error);

});

**¿Qué es el callback hell?**

El infierno de devolución de llamada en Node.js es la situación en la que tenemos devoluciones de llamada anidadas complejas. En esto, cada devolución de llamada toma argumentos que se han obtenido como resultado de devoluciones de llamada anteriores. De esta manera, la estructura del código parece una pirámide, lo que conduce a una menor legibilidad y dificultad en el mantenimiento. Además, si hay un error en una función, todas las demás funciones se ven afectadas.