

# PORQUE NO HAY QUE USAR ECHO EN UNA API

El principal problema de hacer un echo en una API REST es que rompes la respuesta HTTP y puede provocar errores muy difíciles de encontrar.

## ❌ 1. Ensucia la salida JSON

Tu API debe devolver solo JSON (o XML), pero si haces un echo extra, por ejemplo:

php

```
echo "Hola";  
echo json_encode($data);
```

El cliente recibirá:

código

```
Hola{"id":1,"nombre":"Juan"}
```

👉 Eso no es JSON válido, y el cliente no podrá parsearlo.

## ❌ 2. Rompe los encabezados HTTP

Si haces un echo antes de enviar header(), por ejemplo:

php

```
echo "debug";  
header('Content-Type: application/json');
```

PHP mostrará:

código

```
Warning: Cannot modify header information – headers already sent
```

Y tu API no tendrá los headers correctos → fallo seguro.

### ❌ 3. Los logs de depuración via echo son peligrosísimos

Ejemplo:

```
php

echo "Conectado";
echo json_encode($empleados);
```

El cliente recibe texto mezclado con la respuesta → rompe el protocolo.

### ❌ 4. APIs modernas requieren “output limpio”

El estándar REST implica respuestas limpias con:

- códigos HTTP correctos
- headers correctos
- body válido (JSON, XML, etc.)

Un echo rompe ese contrato.

### ✓ ¿Cuál es la forma correcta?

Nunca uses echo en tu API.

Para depurar, usa:

- `error_log("mensaje");` → va al log, NO al cliente
- `var_dump()` controlado solo en entorno de desarrollo
- `xdebug`, `logs`, etc.

Usa:

```
php

header('Content-Type: application/json');
http_response_code(200);
echo json_encode($data);
exit;
```