

L U M B R E



Leer logs sin pánico: el registrador de tu aplicación

Materia: Desarrollo de Software con IA

07/10/2026

Antes de empezar

Los logs son la caja negra de tu app: registran todo lo que pasó, en orden, con marca de tiempo. Cuando algo falla, el log te dice dónde buscar sin tener que entender cada línea de código. Aquí aprendes a leerlos —niveles, timestamps, trazas de error— y a pedirle a la IA que los haga legibles cuando no los hay.

QUÉ APRENDERÁS

- Interpretar niveles (info, warn, error), timestamps y trazas para ubicar la falla.

La caja negra que habla por ti

Un log es la memoria de la aplicación: qué ocurrió, cuándo y con qué resultado. Cuando el error está en producción y no puedes «verlo», el log es tu único testigo. No lo escribes a mano: lo pides. Y aprendes a **escanearlo** buscando lo que brilla en rojo.

Los niveles: qué mira cada uno

En un incidente, ve directo a los «error» y luego retrocede desde ahí en el tiempo.

Nivel	Qué significa	Cuándo mirarlo
info	Todo normal, un hito	para seguir el flujo habitual
warn	Algo raro pero no roto	cuando sospechas de un borde
error	Algo falló	el primer sitio en un incidente
trace / stack	La ruta exacta del fallo por dentro	para localizar la línea culpable

Un log de ejemplo

El rastro de un 500

```
[10:02:11] INFO GET /contactos -> 200 (34 registros)\n[10:02:15] INFO POST /contactos cuerpo={nombre:"Ana"}\n[10:02:15] WARN email vacio, se acepto igual\n[10:02:40] ERROR GET /contactos/87 -> fallo: no existe\n[10:02:40] ERROR stack: findById(87) devolvio null; se llamo .telefono sobre null\n[10:02:40] INFO respuesta al cliente: 500
```

Qué le pides a la IA con esto

Le copias las líneas relevantes (no el archivo entero) y aplicas el protocolo del OA anterior. Los logs son el paso 4; unen el error con su contexto. También puedes pedir que **instrumente** la app para generar logs útiles donde aún no los hay:

Del log a la corrección y a la observabilidad

```
El log de produccion muestra: ERROR stack: findById(87) devolvio null; se llamo .telefono sobre null.\nAñade un chequeo para que un id inexistente devuelva 404 con mensaje claro en vez de reventar.\nAdemas, instrumenta el endpoint GET /contactos/{id} con log INFO de la entrada y WARN si no encuentra registro.
```

Prueba

1. Tu app devolvió un error 500 en producción. ¿Qué línea del log te señala primero la causa?

- ☐ La línea INFO de un GET anterior correcto.
- ☐ La línea ERROR con su stack: «findById devolvió null; se llamó .telefono sobre null».
- ☐ El timestamp de la petición.
- ☐ La línea WARN de email vacío.

2. Un log útil debe registrarse solo cuando algo falla; nunca conviene tener INFO de las operaciones normales.

- ☐ Verdadero
- ☐ Falso

Detecta el log inútil

ACTIVIDAD
INTERACTIVA
DISPONIBLE EN
LUMBRE

3. Revisas cómo una IA dejó la aplicación instrumentada. Señala la decisión que te dejará ciego ante el próximo fallo:



4. Pide a la IA que instrumente una ruta de tu proyecto con INFO, WARN y ERROR útiles. Provoca un fallo y localiza su causa leyendo solo el log.

Si no puedes explicar el fallo desde el log, tu registro todavía no te sirve.

Referencia curricular

Materia:	Desarrollo de Software con IA
-----------------	-------------------------------

Versión interactiva completa (con actividades, videos y autoevaluación):



<https://empresa.lumbre.lat/object/fsi-m12-0a02>

Material educativo generado desde Lumbre · lumbre.lat