

Revisión de mariscal.ai: formulario de contacto y diagnóstico exprés

Publicado el 17-10-2026. Versión 1.

Sistema revisado. Web estática en Cloudflare Pages con tres funciones de servidor.

`/api/contacto` me manda el formulario por correo (Resend). `/api/diagnostico` firma las respuestas de un cuestionario de 7 pasos en un token y manda un correo de confirmación. `/api/confirmar` comprueba el token (doble opt-in) y manda el resultado.

Fechas. Nueve rondas el 2026-10-11: r1-r6 sobre la web y el formulario, r7-r9 sobre el diagnóstico. Publicación el 2026-10-12. Comprobaciones en producción el 2026-10-12 y el 2026-10-16.

Alcance. Entró el código de las tres funciones, los scripts de despliegue y de secretos, el HTML y el CSS generados y los textos públicos. No entró la configuración remota de Cloudflare y Resend, la medición en navegador ni el aviso de cookies, que tuvo revisión propia.

Método. Dos revisores independientes por ronda.

- Revisor A: un modelo de otro proveedor, distinto del que escribió el código, en sesión limpia y de solo lectura. Dos variantes: A1 itera, A2 cierra. Lee el diff y cada fichero con su hash; no ejecuta. Verifiqué el modelo de cada ronda salvo r9.
- Revisor B: un agente revisor sin el contexto de la construcción. Ejecuta los tests y reproduce con un `fetch` simulado.

Cierre: dos rondas seguidas sin hallazgo alto. El 2026-10-16 pasé cuatro tests contra la versión anterior al arreglo: los cuatro fallan (F1, F4, F5 y F8).

Resumen

Recuento bruto, con duplicados. A, 9 rondas: 3 altos, 33 medios, 6 bajos. B, rondas 1-3 y 7-9: 7 altos, 32 medios, 50 bajos.

Ficha	Hallazgo	Gravedad	Estado
F1	El plazo de lectura cancelaba antes de rechazar	Alta	Arreglado, <code>9e0bd5e</code> , 2026-10-11
F2	Aviso de error vacío	Alta	Arreglado, <code>acaa11e</code> , 2026-10-11
F3	Relay de correo desde mi dominio	Alta	Arreglado, <code>8cf77a8</code> , 2026-10-11
F4	Token con cuatro codificaciones válidas	Media	Arreglado, <code>c06af75</code> , 2026-10-11
F5	Cuerpo sin tope real de 32 KB	Media	Arreglado, <code>2966d4e</code> , 2026-10-11
F6	Llamada a Resend sin plazo	Media	Arreglado, <code>2966d4e</code> , 2026-10-11
F7	Dos lectores distintos del <code>.env</code>	Media	Arreglado, <code>33b0606</code> y <code>a640b7f</code> , 2026-10-11
F8	KV que tiraba el handler	Media	Arreglado, <code>c06af75</code> , 2026-10-11
F9	<code>bash -x</code> imprimiría las claves	Media	Arreglado, <code>636145d</code> , 2026-10-11
r7	<code>[hidden]</code> pisado por el CSS; repetición de <code>/api/confirmar</code> ; aviso de contacto nuevo que fallaba en silencio	Alta	Arreglado, <code>8cf77a8</code> , 2026-10-11
r9	R9-01, R9-02, R9-03	Media	Arreglado, <code>073ac60</code> , <code>6efccc7</code> , <code>7a9544b</code> , 2026-10-16

Hallazgos

F1. El plazo de lectura cancelaba antes de rechazar. A2, r4, registro de la ronda 4, R4-01. Lo introdujo el arreglo de R3-01 (A1, r3). Al vencer los 10 s, cancelar el stream cerraba la lectura pendiente con `done=true`, que podía ganar la carrera. Escenario: formulario completo y conexión abierta; se mandaba lo recibido hasta ahí. Alta: el usuario ve «gracias» y el mensaje llega truncado. Arreglo: rechazar primero, cancelar después y cortar el bucle con una bandera. Prueba: formulario válido y stream que nunca cierra. Contra la versión anterior devuelve `/gracias/` en vez de `/contacto/?error=1#error`.

F2. Aviso de error vacío. B, r1, H1; A1 lo marcó medio (registro de la ronda 1, R01). El aviso leía un atributo `data-error` que nadie ponía: cada fallo de envío acababa en una caja roja sin texto. Alta: la regla del formulario es que el usuario vea el error. Arreglo: el texto va en el HTML y lo muestra `:target`. Prueba: el texto está en el HTML de producción (`curl`, 2026-10-16). No hay test automático.

F3. Relay de correo desde mi dominio. B, r7, H3. Un `curl` sin `Origin`, sin `Sec-Fetch-Site` y sin marca de tiempo mandaba un correo a cualquier dirección, con mi remitente y el nombre libre en la primera línea («Hola, Llama ya al 600000000...»). Sirve para phishing y para agotar la cuota. Arreglo: el correo de confirmación no lleva nada del formulario, todo POST exige origen propio y hay freno por IP. Prueba: tres tests del relay en verde con el arreglo (`8cf77a8`, 30 de 30). Contra la versión anterior (`00a2d56`) el fichero de tests no llega a cargar (importa `MAX_P0R_H0RA`, que no existía), así que el «rojo» de F3 no es un test fallando sino el `curl` original de la ronda 7, reproducido antes del arreglo (2026-10-16, notas del proyecto).

F4. Token con cuatro codificaciones válidas. A2, r8, registro de la ronda 8, R8-01; B, r8, high 1. El decodificador ignoraba los dos bits sobrantes del último carácter de la firma. Un token usado admitía tres variantes, cada una con su marca de uso: tres envíos más por enlace, dentro del freno de 10 por hora. Arreglo: solo vale la codificación canónica. Prueba: test con las tres variantes; falla contra el código anterior.

F5. Cuerpo sin tope real de 32 KB. A2, r2, registro de la ronda 2, R1; antes B, r1, M4. El tope miraba solo `Content-Length`. Sin esa cabecera, o con una falsa, el cuerpo se leía entero en memoria. Arreglo: leer a trozos y cortar al pasar el límite.

F6. Llamada a Resend sin plazo. A2, r2, R2. Si el proveedor no contestaba, la función esperaba sin fin y el usuario no veía el error. Arreglo: `AbortController` de 10 s sobre la petición y la lectura de la respuesta. Prueba de F5 y F6: un fichero de 6 tests; contra la versión anterior fallan 4.

F7. Dos lectores del `.env`. A2, r4, R4-02, y r5, R5-01. El despliegue cargaba las claves con `source` y el script del secreto usaba otro parser. Con comillas, comentarios o `$VAR` leían valores distintos y se publicaba una clave errónea sin aviso. Arreglo: un único cargador. Prueba: ejecución real; el fichero no tenía ningún `$` y el valor publicado era correcto. No hay test automático.

F8. KV que tiraba el handler. A2, r8, R8-02; B, r8, M2. KV admite una escritura por segundo y clave. Dos envíos desde una IP en el mismo segundo lanzaban una excepción: página 500, también con los correos ya enviados. Arreglo: capturar los fallos y redirigir con error. Prueba: test con un KV que falla; el código anterior lanza `KV GET failed: 500`.

F9. `bash -x` imprimiría las claves. A2, r6, registro de la ronda 6, R6-01. Con trazas activas, la carga del `.env` escribía cada clave en stderr. Arreglo: `set +xv` y salida silenciada en los dos scripts. Prueba: con `bash -x` ya no sale ninguna asignación. No hay test automático.

Lo que se buscó y no quedó abierto

- Redirección abierta: B, r1, con `lang=//evil.com`. Vuelve a `es`. No encontrada.
- Inyección: B encontró CR/LF en el asunto (r1, medio, `acaa11e`) y patrones `$` en plantillas (r7, medio, `8cf77a8`). En r8, A2 y B buscaron XSS e inyección de cabeceras: nada.
- Origen ajeno y honeypot: en r1 llevaban a «gracias» (A1 R02, B M2; `acaa11e`). En la preview, 2026-10-11: honeypot, origen ajeno y 40 KB troceados acaban en error.
- CSP: `script-src 'self'`, sin `unsafe-inline` (`curl` a producción, 2026-10-16). `style-src` sí lo mantiene.
- Caché: `/_astro/*` con `public, max-age=31536000, immutable` (2026-10-16).
- 404: una ruta inexistente da 404 con página propia (2026-10-16).
- Producción, 2026-10-12: un POST real llegó a `/gracias/` con el correo entregado; del 6.º POST seguido en adelante, 429.

Residual declarado

Literal de las notas del proyecto, 2026-10-11. Abierto tras r9 (notas del proyecto; ahí «F2 (low)» es un hallazgo de B en r9, no la ficha F2):

- R9-03 (medium): `EMAIL` rechaza el apóstrofo en la parte local (o'connor@...): esa persona no puede mandar ni el contacto ni el diagnóstico. Fix de un carácter en `web/src/lib/texto.ts` + test. El primero del lote.
- R9-02 (medium): el freno por IP renueva la ventana de una hora en cada envío (KV cuenta el TTL desde la última escritura): cinco envíos espaciados desde una IP compartida bloquean al sexto aunque no haya cinco en la última hora. Fix: clave con la hora (`d:<ip>:<hora>`) o guardar el inicio de la ventana.
- R9-01 (medium) y F2 (low): `kvLeer` devuelve null también cuando KV falla; entre las 24 h de la clave de Resend y los 7 días del token, un fallo selectivo de lectura podría repetir un correo. Fix: distinguir «no hay clave» de «KV falló» y dar error en el segundo caso.

Aceptado tras r7 (notas del proyecto):

`/revisa-tu-correo/` abierta a pelo enseña el texto de «revisa tu correo» (no afirma envío: es una página informativa sin índice); [...] un `#r=` fabricado en `/confirmado/` enseña «Hecho» (sin estado de servidor no se distingue; sin daño: solo confunde a quien fabrica el enlace).

Origen de R9-01 a R9-03: A2, r9, registro de la ronda 9. Los tres se arreglaron el 2026-10-16, cada uno con su test en rojo antes y en verde después: R9-03 en `073ac60` (el apóstrofo vale en la parte local, pero no como primer carácter: `'@x.com` sigue siendo inválido), R9-02 en `6efccc7` (la ventana guarda su inicio en el valor y ya no se renueva con cada envío; un contador antiguo cuenta como ventana abierta ahora, que frena de más y nunca de menos) y R9-01 en `7a9544b` (un fallo de lectura del almacén ya no se toma por «no hay clave»: el formulario enseña el error y no se manda ningún correo). Publicados en producción con esta versión de la web.

Qué recibe un cliente

1. Un informe como este, con lo buscado sin resultado.
2. Un commit por arreglo, con el hallazgo citado en el mensaje.
3. Un test por arreglo, pasado contra la versión anterior para ver que falla.
4. Una inspección nueva en sesión limpia hasta dos rondas seguidas sin alto.
5. El residual por escrito, con el motivo y quién lo decide.

Aitor Mariscal · mariscal.ai · Informe anonimizable: este es el sistema propio del autor.