POR BAIXO DO CAPÔ · OUT/2026

7 checagens antes de pôr uma API em produção

A API é a porta mais exposta do sistema. Escrita por você ou por um agente, ela passa por estas sete checagens **antes do merge**.



1



Chave só no servidor

O QUE FAZER

- ✓ A chave fica em variável de ambiente no servidor.
- ✓ O app e o site chamam o seu back-end, e o back-end chama a API de fora.
- ✓ Nunca no código, no app móvel, no front-end nem no repositório.

```
$ const chave = process.env.API_KEY // só no servidor
```

COMO CONFERIR · Abra o site com a aba Rede do navegador: nenhuma requisição leva a chave do fornecedor.

2

Escopo mínimo por integração

O QUE FAZER

- ✓ Uma chave por integração, nunca uma chave mestra para tudo.
- ✓ Só as permissões que aquela integração usa.
- ✓ Restrinja por IP ou domínio quando o fornecedor permitir.

```
$ chave_rotas → escopo: rotas:ler
```

COMO CONFERIR · Se essa chave vazar hoje, o que ela consegue fazer? A resposta tem que caber em uma linha.

3

Rotação de chave

O QUE FAZER

- ✓ Data de troca na agenda, não só quando der problema.
- ✓ Durante a troca, aceite a chave nova e a anterior; depois revogue a antiga.
- ✓ Vazou ou foi colada num chat: troque na hora.

```
$ API_KEY_ATUAL + API_KEY_ANTERIOR → revoga em 7 dias
```

COMO CONFERIR · Você sabe quando a chave foi criada e quem consegue revogá-la.



Timeout

O QUE FAZER

- ✓ Toda chamada para fora tem tempo máximo explícito.
- ✓ Novas tentativas com espera crescente e limite de vezes.
- ✓ Falha vira resposta clara para o usuário, não tela travada.

```
$ fetch(url, { signal: AbortSignal.timeout(5000) })
```

COMO CONFERIR · No teste, aponte para um endereço que não responde: a sua API devolve erro em segundos.



429 com Retry-After

O QUE FAZER

- ✓ A sua API limita pedidos por chave ou usuário e responde 429 com Retry-After.
- ✓ Ao chamar uma API de fora e receber 429, espere o Retry-After antes de tentar de novo.
- ✓ Sem limite, um cliente com bug derruba todo mundo.

```
$ if (res.status === 429) await esperar(retryAfter)
```

COMO CONFERIR · Em ambiente de teste, dispare 100 pedidos seguidos: o 429 aparece com Retry-After.

※ OWASP API4:2023 · consumo de recursos sem limite

6

Idempotência em reenvio

O QUE FAZER

- ✓ POST que cobra, cria pedido ou envia aceita um cabeçalho Idempotency-Key.
- ✓ Mesma chave, mesma resposta: nada é criado duas vezes.
- ✓ Guarde a chave por um prazo definido.

```
$ Idempotency-Key: 8e03978e… → 2° envio devolve o mesmo 201
```

COMO CONFERIR · Mande o mesmo pedido duas vezes seguidas: sai uma cobrança só.

※ Idempotency-Key é rascunho da IETF, não RFC: documente no contrato.

7

Log sem dado sensível

O QUE FAZER

- ✓ Registre método, rota, status, duração e um id de correlação.
- ✓ Nunca credencial, token, senha, cartão, documento nem o corpo inteiro.
- ✓ Mascare o que precisar aparecer.

```
$ log({ rota, status, ms, req_id })
```

COMO CONFERIR · Busque nos logs por "Bearer", "key" e "cpf": zero resultados.



Contrato primeiro. Merge só depois das sete.

FONTES · conferidas em 25/10/2026

RFC 9110 · HTTP Semantics (2022)
métodos e códigos 200, 201, 401, 403, 422

RFC 6585, seção 4 (2012)
429 Too Many Requests · Retry-After

RFC 6750 (2012)
Authorization: Bearer

RFC 9457 (2023)
Problem Details · application/problem+json

OWASP API Security Top 10 (2023)
API1 autorização por objeto · API4 consumo sem limite

IETF draft-ietf-httpapi-idempotency-key-header-08
rascunho de 22/10/2026 · não é RFC