interweb

Gerado a 15/11/2026 00:27

BASE DE CONHECIMENTO · DOCUMENTO PÚBLICO

Erro de CORS entre o seu front end e a sua API: como resolver

Node.js e JavaScript

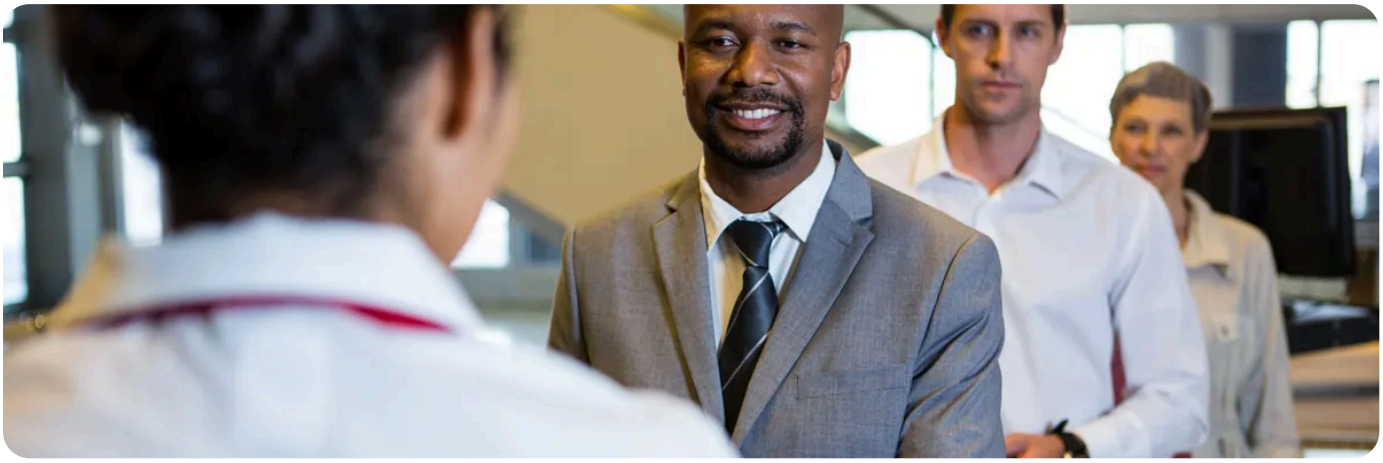
CÓDIGO	VERSÃO	PAÍS · IDIOMA
RLX-AO-KB-1082	v2026.10.10	Angola · Português

Interweb Angola
RLX Angola, Lda. · Grupo RLX Holdings
NIF 500 260 5724 · Matrícula 26464-25/250724

https://www.meu.interweb.ao/knowledgebase/1082

Erro de CORS entre o seu front end e a sua API: como resolver

Node.js e JavaScript · Criado em **10 de outubro de 2026** · 2 min de leitura



«Access to fetch at … has been blocked by CORS policy: No 'Access-Control-Allow-Origin' header is present». É uma das mensagens mais procuradas por quem separa o site (React, Vue) da API (Node, Python). **A resposta curta:** o bloqueio é feito pelo **navegador**, não pelo servidor, e só se levanta se a **API** disser, em cabeçalhos da resposta, que aquele site de origem pode ler a resposta. O pedido chegou à API; o navegador é que não deixa o seu JavaScript ver o resultado.

O que é uma «origem»

A origem é o conjunto **protocolo + domínio + porta**. `https://asuaempresa.ao`, `https://www.asuaempresa.ao`, `http://asuaempresa.ao` e `https://api.asuaempresa.ao` são quatro origens diferentes. Se o site é servido por uma e a API por outra, é um pedido entre origens, e o navegador exige a autorização. Para pedidos que alteram dados ou levam cabeçalhos próprios, envia primeiro um pedido de «pré-verificação» (`OPTIONS`); se a API não responde bem a esse, o pedido verdadeiro nem sai.

Pôr a API a autorizar a origem certa

A API é em…	Biblioteca	Configuração (exemplo)
Express (Node.js)	`cors` (`npm install cors`)	`app.use(cors({ origin: "https://asuaempresa.ao" }))`
Flask	`flask-cors`	`CORS(app, origins=["https://asuaempresa.ao"])`
Django	`django-cors-headers`	`CORS_ALLOWED_ORIGINS = ["https://asuaempresa.ao"]` e o *middleware* no topo da lista
FastAPI	Vem com o `CORSMiddleware`	`app.add_middleware(CORSMiddleware, allow_origins=["https://asuaempresa.ao"])`

Indique a origem **exacta** do seu site, com o protocolo e sem barra final. Depois reinicie a API: **aplicações Node**, e as de Python, mantêm o código antigo até serem reiniciadas.

O erro de CORS pode estar a esconder outro.

Se a API devolve um 500 ou um 503, a página de erro não leva os cabeçalhos de CORS, e o navegador mostra o erro de CORS em vez do verdadeiro. Antes de mexer na configuração, abra o endereço da API directamente (ou use `curl -i`) e veja se responde bem. Se não responde, o problema não é CORS: **erro 503 numa aplicação Node.js**.

Não ponha `*` por preguiça.

Autorizar qualquer origem é aceitável para uma API pública só de leitura. Para uma API com contas, senhas ou cookies, não: abre a porta a qualquer site. Além disso, o navegador rejeita `*` quando o pedido leva cookies ou credenciais. Liste as origens que usa de facto.

A solução que evita o problema:

Servir o site e a API na *mesma origem*, por exemplo a API num caminho `/api` do mesmo domínio, atrás de um proxy. Sem pedidos entre origens, não há CORS. Num VPS é natural: **nginx como proxy inverso**.

PRECISA DE AJUDA?

A API responde ao curl e o navegador continua a bloquear? Diga-nos o domínio do site, o da API e o que a consola do navegador mostra.

Abrir um pedido de suporte →

VEJA TAMBÉM

- **Publicar um site React ou Vue: a pasta de build e o 404 ao atualizar** →
- **Correr uma aplicação Flask ou FastAPI** →
- **Nginx como proxy inverso à frente de um contentor** →

interweb

Interweb Angola

RLX Angola, Lda. · Grupo RLX Holdings

NIF	500 260 5724
MATRÍCULA	26464-25/250724
SEDE	Rua Direita do Patriota, Bairro Dangereux Belas, Luanda, Angola
CONTACTOS	info@interweb.ao · +244 942 011 191 · www.interweb.ao
HORÁRIO	2ª. à 6ª. Feira - 08 à 16 Horas Sábado & Domingo - Fechado

Versão online deste documento

RLX-AO-KB-1082 · v2026.10.10 · Gerado a 15/11/2026 00:27