interweb

Generated on 17/10/2026 09:28

KNOWLEDGE BASE · PUBLIC DOCUMENT

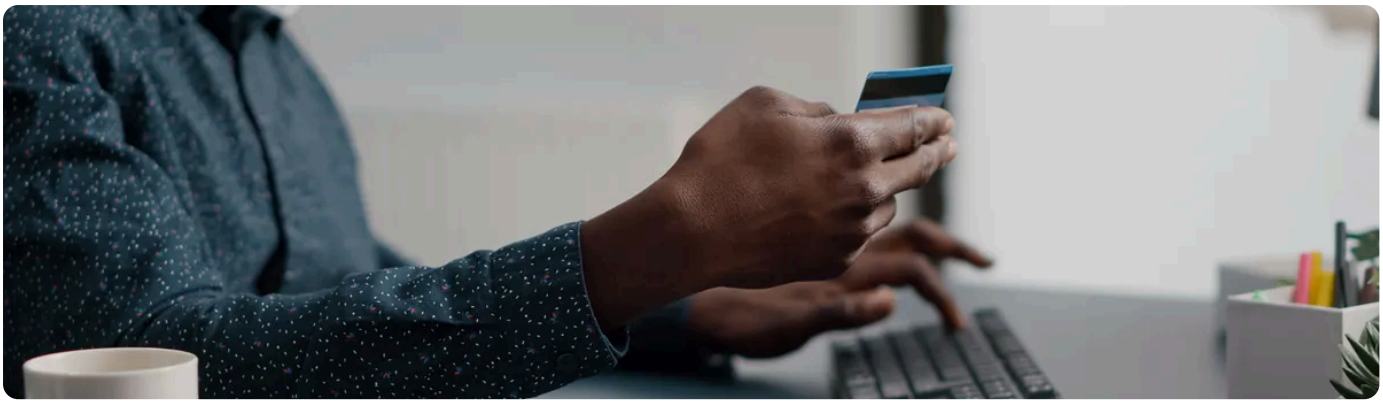
Synchronous or asynchronous: why a payment or SMS confirmation arrives later

Automation & n8n

CODE	VERSION	COUNTRY · LANGUAGE
RLX-AO-KB-2164	v2026.10.16	Angola · English

Interweb Angola
RLX Angola, Lda. · Group RLX Holdings
Tax ID (NIF) 500 260 5724 · Commercial registration no. 26464-25/250724

https://www.meu.interweb.ao/knowledgebase/2164



The short answer: with a **synchronous** request, your site asks and waits until it has the final answer, as when it reads the status of an order. With an **asynchronous** request, the API answers straight away with “received, processing” and the result arrives later, by another route. Mobile payments, payments by reference or bank transfer, and SMS delivery are asynchronous by nature: they depend on a person confirming on a phone, a bank or a mobile network, and no server should sit waiting for that. This is for anyone wiring payments, SMS or other APIs into a site.

What is usually synchronous, and what is not

Situation	Type	Why
Looking up an order, a customer, a balance	Synchronous	The answer already exists on the other side: it just has to be read.
A payment the customer confirms on the phone (with a PIN or a code)	Asynchronous	The API accepts the request, but the outcome depends on the customer accepting, declining or letting it time out.
A payment by reference or bank transfer	Asynchronous	The customer pays when they pay, sometimes long after leaving the site.
A card payment on the operator’s page	Looks synchronous	The customer sees the outcome at once, but the confirmation you can trust is the notice the operator sends to your server. See **callback URL or return URL**.
Sending an SMS	Asynchronous	The API accepts the message and returns an identifier. Whether it was delivered or failed is only known later, from the delivery report that comes back from the network.
An export, a large report, a video conversion	Asynchronous	It takes time. The API returns a job identifier and tells you (or lets you ask) when it is done.

Handling an asynchronous response

1. **Recognise the response.** A **202 Accepted** code, or a status such as “pending”, “processing” or “queued” in the JSON, means “got it, not done yet”. It is neither a success nor an error.

2. **Store the identifier** the API returns (for the transaction, the message, the job) with your order, before showing the customer anything. Without it, when the confirmation arrives you cannot tell which order it belongs to.

3. **Mark the order as pending,** not as paid or sent. On the thank-you page, tell the truth: “We have your order; as soon as the payment is confirmed, we will e-mail you.”

4. **Collect the result** one of two ways: the API calls an address of yours when it has the result (a callback or webhook), or your system asks for the status with the identifier every so often (polling). See **webhook or polling**.

5. **Change the status only on a verified final answer:** paid, declined, expired, delivered, failed. If the result comes by callback, make sure it really came from the service: **receiving a webhook in PHP and checking its signature**.

6	**Deal with what never arrives.** A scheduled task that looks for orders pending too long, asks the API for their status and decides (confirm, cancel, warn) catches the notices that got lost: **running a PHP script from a cron job**.

Do not make PHP wait for the confirmation inside the visitor's request (a loop with `sleep()` that keeps asking the API). Every waiting visitor ties up a process, the request runs into the maximum execution time, and the customer ends up on an error page. See **PHP limits** and **entry processes**.

The page that "updates by itself" is a little JavaScript that asks **your** server for the order status every so often (never the API directly, which would expose the key). Your server answers with what it already knows, which is whatever the callback or the scheduled task left there.

Payments stay pending on your site while the operator says they are paid? Open a ticket with the time of one payment and the callback address you configured, and we will check whether the operator's request reached the hosting.

Open a support ticket

SEE ALSO

Webhook or polling: how one system learns that something changed in another

Callback URL or return URL: why a customer coming back is not proof of payment

Timeouts, retries and idempotency: how not to charge a customer twice

Receiving a webhook in PHP and checking its signature

Running a PHP script from a cron job

How online payments work: gateway, processor and merchant account in plain words

interweb

Online version of this document

Interweb is a leading internet services company, with a mission to help businesses to be born and prosper through technology.

2020	37,000+	130+
FOUNDED	CLIENTS	SIGN-UPS A DAY

Interweb Angola

RLX Angola, Lda. · Group RLX Holdings

TAX ID (NIF)	500 260 5724
COMMERCIAL REGISTRATION NO.	26464-25/250724
REGISTERED OFFICE	Rua Direita do Patriota, Bairro Dangereux Belas, Luanda, Angola
CONTACT	info@interweb.ao · +244 942 011 191 · www.interweb.ao
OPENING HOURS	Monday to Friday - 08 AM to 04 PM Saturday & Sunday - Closed

RLX-AO-KB-2164 · v2026.10.16 · Generated on 17/10/2026 09:28