GOHIGHLEVEL · TECHNICAL SOP

Meta Conversion API Implementation Guide

How to wire server-side conversion tracking into a GoHighLevel sub-account: what fires, when it fires, and how to prove it works before any ad spend goes on.

PLATFORM GoHighLevel + Meta Events Manager	EVENTS TRACKED Lead · Purchase · Schedule
STRUCTURE 3 dedicated workflows + 1 existing-workflow edit	BUILD TIME 1-2 hours including QA
AUTHOR Kenneth Villar · BrewedOps	WRITTEN FROM a live 2026 production build

1. Why server-side tracking

Browser-side pixel tracking alone is broken in 2026. Ad blockers strip the pixel from roughly 30 percent of users, iOS shortens cookie lifetimes, and third-party cookies are being retired. Anything the browser was supposed to report simply never arrives.

The Conversion API sends the event from the GHL server straight to Meta. Nothing to block, nothing to clear. What that buys:

- **Higher Event Match Quality** - Meta can identify the person reliably.
- **Accurate attribution** - the Facebook click ID is captured server-side and cannot be stripped.
- **Cleaner ad spend** - Meta's optimization learns from real conversions, not a sampled subset.
- **Privacy-durable** - survives regulation that keeps tightening on browser tracking.

Keep the browser pixel too. Browser signals stay useful for real-time reporting and retargeting; the server guarantees the conversions that matter actually get counted.

2. The three events

META EVENT	FIRES WHEN	WHAT META OPTIMIZES FOR
Lead	The post-purchase survey is submitted, or the website application form is submitted	Top-of-funnel intent
Purchase	Any funnel order completes - front-end offer, order bump, upsells, downsells	Direct return on ad spend
Schedule	A call is booked on the sales calendar	Back-of-funnel high intent

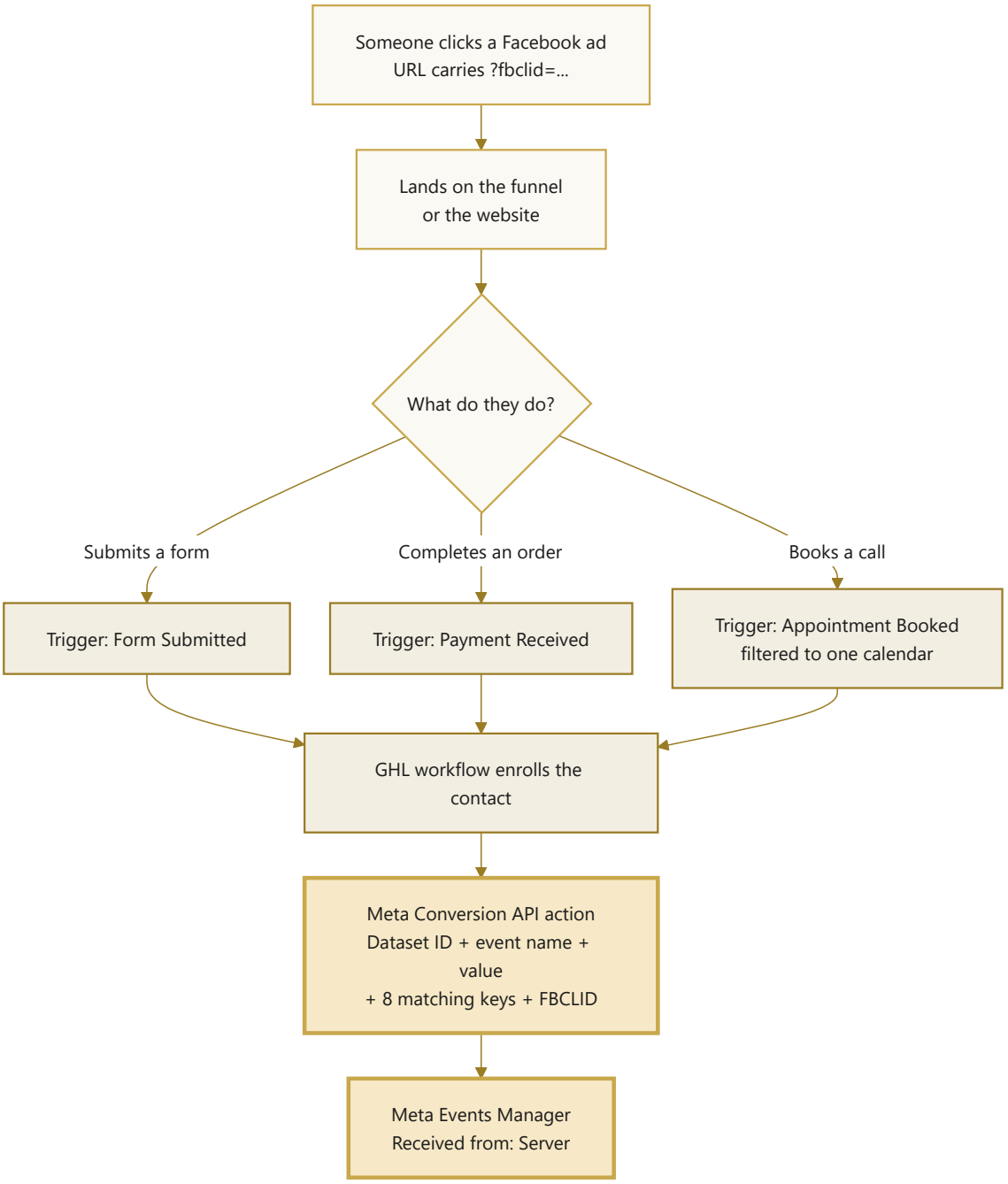

FIGURE 1 - ONE PATH, THREE ENTRY POINTS, ONE DESTINATION

3. What each event carries

Every fire attaches the same payload shape. Only the event name and the value change per workflow.

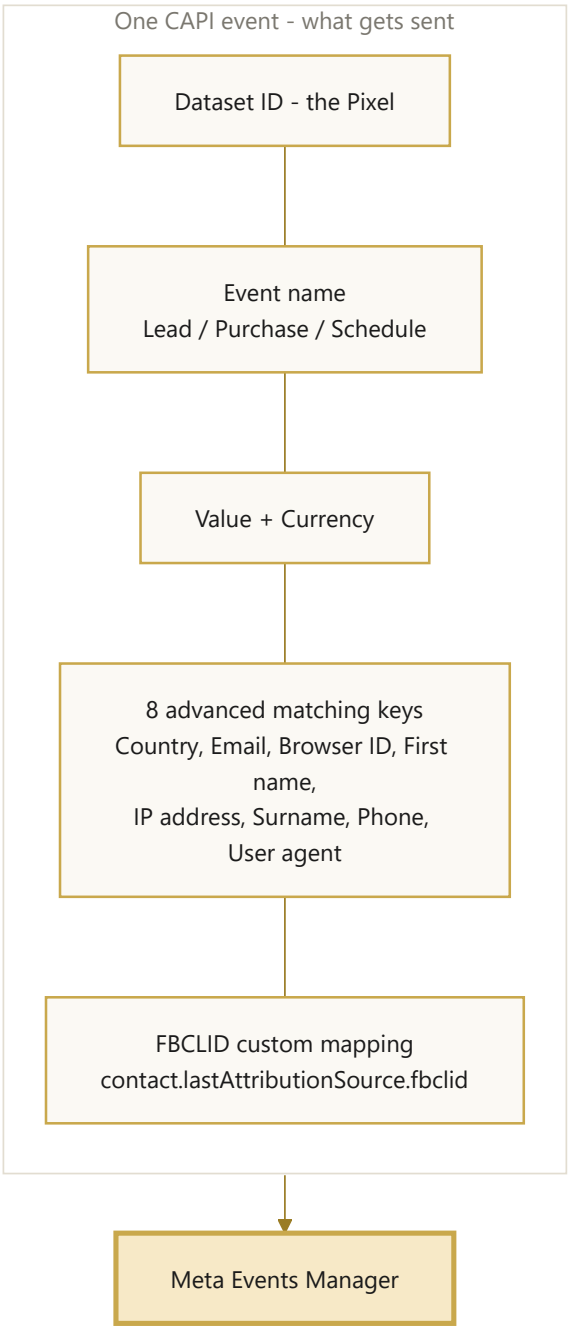

FIGURE 2 - THE PAYLOAD

The 8 matching keys

GHL attaches these automatically from the contact record. No mapping work needed. Together they are the maximum advanced matching set Meta supports, which is what produces the highest match quality score.

1. Country	2. Email address	3. Browser ID (fbp cookie, hashed)	4. First name
5. IP address	6. Surname	7. Phone number	8. User agent

The FBCLID mapping - the one thing not to skip

FBCLID is the unique identifier Facebook appends to every ad-click URL. Captured server-side, it lets Meta attribute the conversion to the exact ad that drove the click, even if the person blocks cookies or clears their browser. Without it, attribution falls back to probabilistic matching, which is far less accurate.

Turn **Custom Mapping** ON in every CAPI action and paste `{{contact.lastAttributionSource.fbclid}}` into the FBCLID field. That merge tag is a GHL default and is identical on every sub-account.

4. Before you open GHL

META SIDE

- [ ] Business Manager account owned by, or admin-shared with, the agency.
- [ ] Pixel created in Events Manager. Note the **Dataset ID** (16 digits).
- [ ] System User Access Token with `ads_management` and `business_management`, scoped to that dataset. It does not expire unless revoked.
- [ ] Test Event Code from Events Manager > Test Events. Format `TEST` + 5 digits.

GHL SIDE

- [ ] Agency-level access to the sub-account.
- [ ] At least one published funnel with order forms wired to GHL Products.
- [ ] At least one calendar for the booking flow.
- [ ] The merge tag `{{contact.lastAttributionSource.fbclid}}` resolving on a test contact who arrived from an ad-tagged URL.

> **HOUSEKEEPING**
> Create a folder in Automation named **Meta Conversion API Tracking** and keep all three workflows inside it. Use the same folder name on every account, so anyone opening a new sub-account knows where to look.

5. The build - three workflows

All three follow the same five steps. Only the trigger and two field values differ.

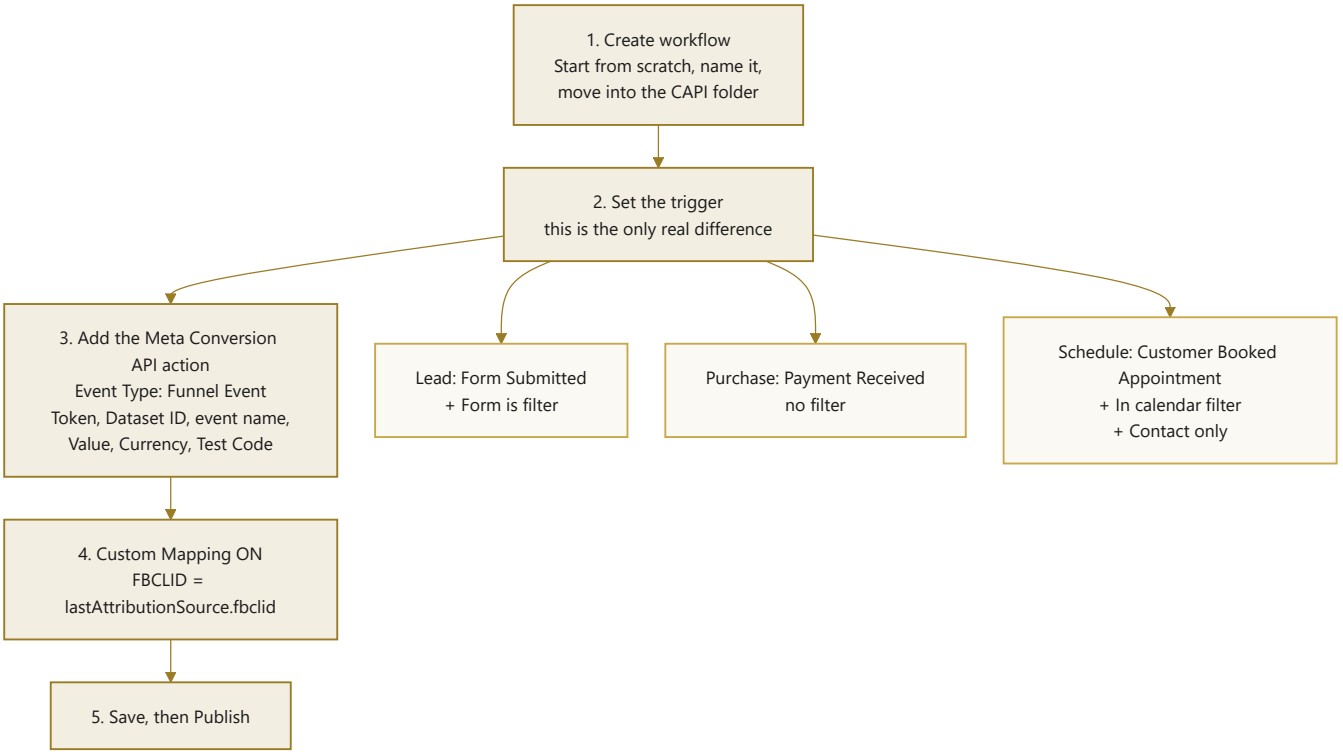

FIGURE 3 - THE SAME FIVE STEPS, THREE TIMES

Workflow 1 - Lead

Name it `Meta Pixel Conversion API – Lead`. Point it at the survey that sits on the thank-you page. Because that survey is completed after the purchase, it is a genuinely high-intent signal rather than a cold opt-in.

FIELD	VALUE
Trigger	Form Submitted → Form is *your thank-you survey*
Event Type	Funnel Event
Access Token	System User Access Token (from your secrets vault)
Dataset ID	the client's Pixel ID
Facebook Event Name	`Lead`
Value / Currency	`10` nominal intent score / `USD`
Test Code	during QA only - remove before live
Custom Mapping > FBCLID	`{{contact.lastAttributionSource.fbclid}}`

IF THE SURVEY IS NOT A GHL FORM

Use a webhook trigger instead and point the external form at it. The CAPI action underneath is identical.

Workflow 2 - Purchase

Name it `Meta Pixel Conversion API – Purchase`. Trigger on **Payment Received** with no filter, because every payment in the sub-account is a purchase signal worth sending. Payment Received beats Order Form

Submission here - it only fires on charges that actually succeeded, not on abandoned checkouts.

FIELD	VALUE
Trigger	Payment Received, no filter
Facebook Event Name	`Purchase`
Value	`{{payment.total_amount}}` - dynamic
Currency	`USD`
Custom Mapping > FBCLID	`{{contact.lastAttributionSource.fbclid}}`

DO NOT HARDCODE THE VALUE

Hardcoding the front-end price breaks ROAS reporting the moment a buyer also takes the order bump or an upsell. The merge tag pulls the real amount charged on each transaction, so Meta sees actual revenue per event.

THE $0 COUPON TRAP

Meta rejects events with a value of 0, so a 100 percent off test order fails validation. During QA only, hardcode `Value: 1` so the test passes - then switch it back to `{{payment.total_amount}}` before launch. In production that rejection is correct behaviour: free orders should not be reported as purchases, or Meta optimizes toward worthless conversions.

Workflow 3 - Schedule

Name it `Meta Pixel Conversion API – Schedule`. This is the one with a detail that is easy to get wrong.

1. Add New Trigger → **Customer Booked Appointment**.
2. Under "Who should be enrolled?", pick the **Contact only** card. Guests must not fire CAPI - only the person who actually booked.
3. Filters → **+ Add filters** → **In calendar** = your sales call calendar.
4. Save. The trigger card should now read *Contact Mode is any of* "*contact*" plus your *In calendar* filter.
5. Add the CAPI action - event name `Schedule`, Value `10`, Currency `USD`, Custom Mapping ON.

WHY THE CALENDAR FILTER MATTERS

A sub-account usually has several calendars - coaching, discovery, podcast guests, support. Without the filter, every booking on any calendar fires a Schedule event and pollutes the optimization data with appointments that have nothing to do with the ad.

TUNING THE SCHEDULE VALUE LATER

`10` is a placeholder. For sharper bid optimization, raise it to the expected revenue per booking - if 1 in 10 calls closes a $30K offer, that is `3000`. If 1 in 10 closes a $5K offer, that is `500`. This only changes how Meta weighs the conversion when bidding; it does not touch real revenue. Agree the number with whoever runs the ads.

6. Adding CAPI to an existing workflow

The website application workflow needs to report Leads too - those applicants came from ads as well, and Meta needs to credit the ad that drove the inquiry.

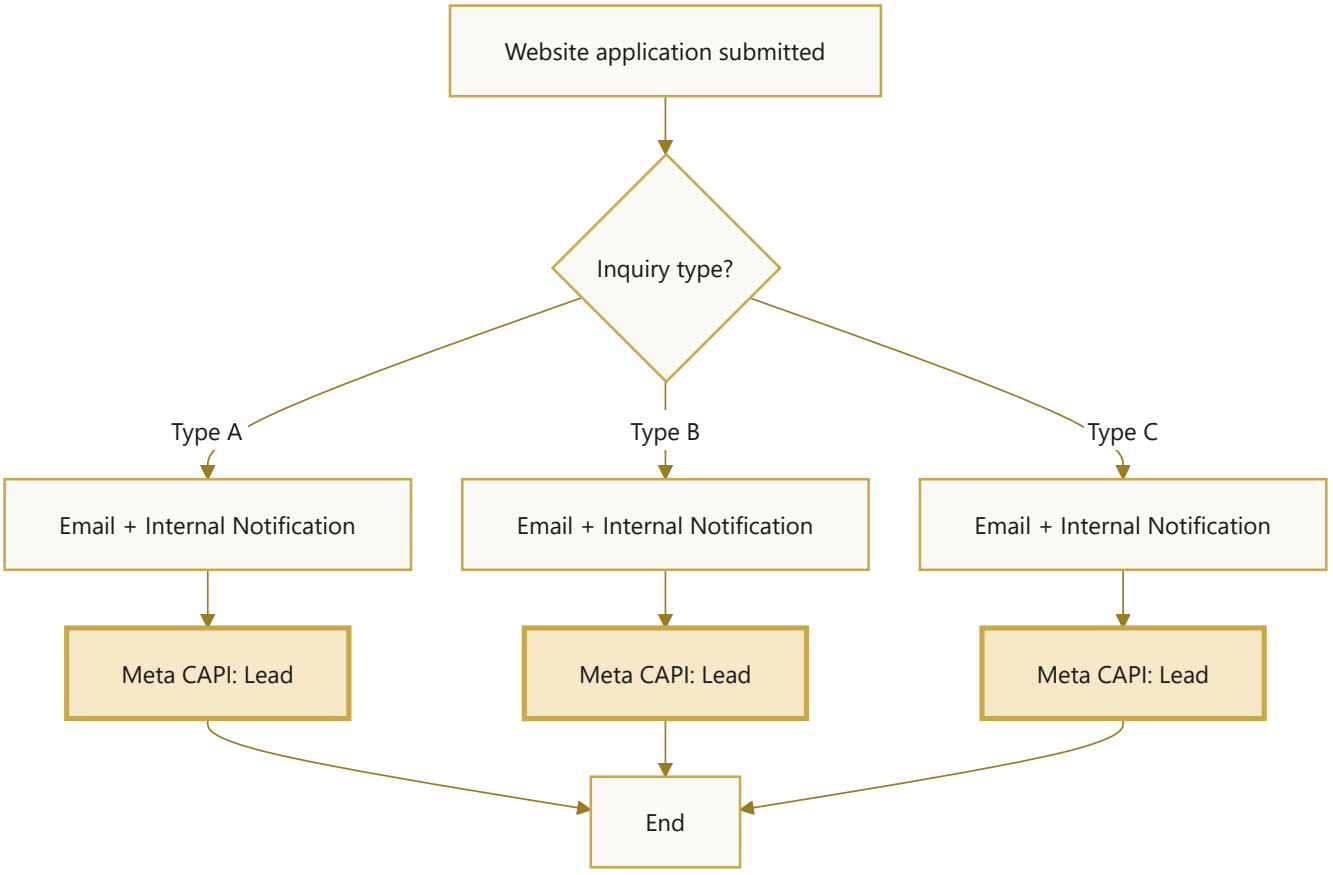

FIGURE 4 - ONE CAPI ACTION PER BRANCH, NOT ONE SHARED ACTION AFTER THE MERGE

Use one CAPI action per branch rather than a single shared action after they merge. The event names are identical today, but keeping them split means the inquiry type can be added later as custom data for audience targeting, without rebuilding the workflow.

Config on each branch matches the Lead workflow, with two differences: leave **Value** empty (Meta default) and leave **Test Code** empty if this workflow is going straight to live.

7. Proving it works

Nothing goes live until every event has been seen landing in Meta. The Test Events tab exists exactly for this.

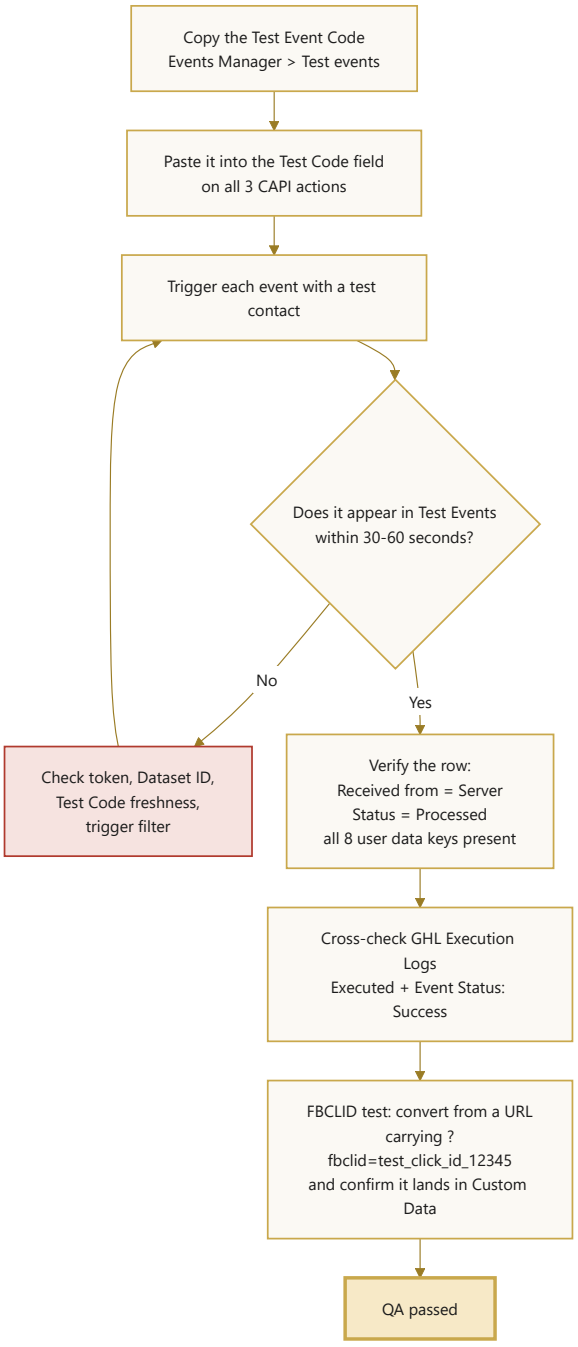

FIGURE 5 - THE QA LOOP

EVENT	TEST ACTION
Lead	Submit the thank-you survey as a test contact.
Purchase	Place an order using a 100 percent off coupon. Set Value to `1` for this phase - $0 is rejected.
Schedule	Book a test appointment on the sales calendar.

End-to-end firing time is normally 2-3 seconds. Anything past 30 seconds points at a token problem or a trigger filter that is not matching. If any of the 8 user data keys are missing, the test contact record is incomplete - fill in country, email, first name, last name and phone, then re-test.

8. Production cutover

Every item below, before a dollar of ad spend goes on. Skipping any one of them either breaks attribution or poisons the data.

- [ ] **Remove the Test Event Code from all three workflows.** Leave it in and every real conversion routes into Test Events, where Meta's optimization never sees it.
- [ ] **Purchase Value back to** `{{payment.total_amount}}` if it was hardcoded to `1` for testing.
- [ ] **Schedule Value matches what the ads team agreed.**
- [ ] **Publish the website application workflow** if it was held in Draft during testing.
- [ ] **Custom Mapping FBCLID confirmed ON** on all four workflows - the three dedicated plus the website one.
- [ ] **Pixel shows recent activity from both Browser and Server** in Events Manager > Overview.
- [ ] **Smoke test one of each event without a test code** and watch it land in Overview, not Test Events, inside 60 seconds.
- [ ] **Access token location documented** in the secrets vault. System user tokens do not expire, but they are tied to the admin who created them - if that person leaves, regenerate.

MESSAGE TO THE ADS TEAM WHEN CUTOVER IS DONE

CAPI cutover complete. All 3 standard events (Lead, Purchase, Schedule) firing server-side with full advanced matching plus FBCLID. Test event codes removed from all workflows. Production Purchase value is dynamic. Pixel ID and event names confirmed. Safe to enable ad spend.

9. Browser pixel + server: avoiding double counts

Funnel pages usually fire their own browser-side events already - `PageView`, `SubscribedButtonClick`, and often a custom event on the order form. The server fires the standard `Purchase`. There are two textbook ways to stop those colliding.

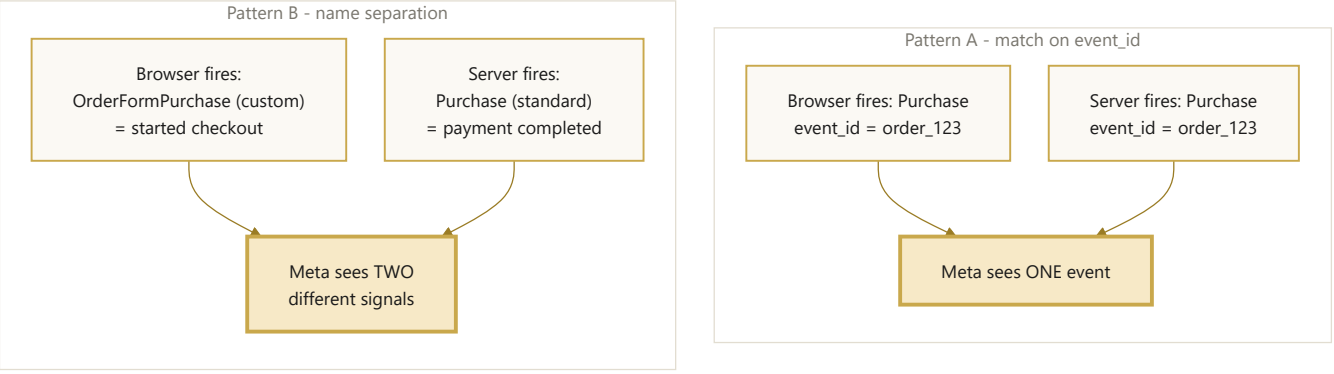

FIGURE 6 - TWO VALID DEDUP PATTERNS

Name separation is the practical default in GHL, for three reasons: the browser-side custom event is usually already live before the CAPI work starts; GHL's native CAPI action does not readily expose an `event_id` field

for browser-server matching; and name separation reaches the same end - no double counting - without touching the funnel build at all.

Tell the ads team to optimize against the server-side `Purchase`. That is the completed-payment signal and it carries the highest match quality. The browser-side order-form event is a secondary signal, useful for retargeting people who started checkout and did not finish.

If a team specifically wants event_id dedup, it needs an `event_id` generated per conversion (typically the contact or payment ID), injected into the browser pixel via custom code on the checkout page and mirrored into the CAPI call - which GHL native does not support, so Stape.io or server-side GTM is required. Verify in Events Manager > Diagnostics > Deduplication.

10. When something is wrong

SYMPTOM	WHERE TO LOOK
Fires in GHL Execution Logs but never appears in Meta	Access token expired or revoked. Test Code stale - Meta retires codes unused for about 7 days. Dataset ID does not match the pixel selected in the Test Events dropdown.
Event lands but Event Match Quality is low	The contact record is missing one of the 8 matching fields. Fill in first name, last name, email, phone and country, then re-test - the score recovers within 2-3 events.
FBCLID missing on the event	Usually correct behaviour. `lastAttributionSource.fbclid` is only populated for people who arrived by clicking an ad. Organic and direct traffic will not have one.
Purchase fires with value 0	A 100 percent off coupon. Meta rejects it, which is right in production. For testing only, hardcode `1`.
The contact never enrolls at all	Trigger filter too restrictive. Check the Stats tab to see whether the trigger fired, then Enrollment History for why the contact was rejected.
Purchases counted twice in Meta	Browser and server both firing the same event name with no dedup. Either align event names plus event IDs, or move to name separation and tell the ads team which event to optimize against.

11. Reusing this on the next account

CHANGES EVERY TIME	STAYS THE SAME
GHL Location ID	The `Meta Conversion API Tracking` folder
Meta Dataset ID	Three dedicated workflows + one website workflow edit
System User Access Token	Event names `Lead`, `Purchase`, `Schedule` - never invent custom names for these
Test Event Code (fresh per QA pass)	Currency, swapped per market
Workflow names per client convention	Custom Mapping FBCLID with the same merge tag
Trigger filters - form, calendar, payment names	The 8 matching keys, attached automatically
Production value for Schedule	

ORDER OF WORK

1. Get the Pixel ID and access token before opening GHL.
2. Generate the Test Event Code.
3. Confirm at least one funnel and one calendar are live in the sub-account.
4. Build the folder and three workflows - 30 to 45 minutes.
5. Add the CAPI Lead action to the website application workflow, if there is one.
6. Run the full QA pass in section 7.
7. Work the cutover checklist in section 8.
8. Send the go-live confirmation to the ads team.

Total for a competent operator, including QA: one to two hours.

Written from a live production build, May 2026. If Meta or GoHighLevel changes and this guide falls out of date, update it in place rather than starting a new one.