# NIE Experts — System Documentation

## 00 · System overview

> Verified live 2026-06-30. NIE Experts Spain is an automated NIE-application service for Italians in Spain: a customer pays once, fills two short forms, and the system generates and delivers their official paperwork (EX-18 + Tasa 790/012) with no human in the critical path. This file is the map — every surface, the end-to-end data flow, and where each piece is documented.

### Purpose

A customer buys a NIE package (digital / expert / concierge) on the website. The backend then, automatically: opens one "practice" (case) per person, collects their data, generates the Spanish forms after a deliberate human-like delay, emails them, files them to Drive, shares the situation guide, and handles upsells, bookings, outcomes, reviews, refunds and delivery failures. The only routine human touch is the WhatsApp agent answering questions.

### Surfaces

| Surface | Role | Where it lives |
|---|---|---|
| Website | Checkout UI, affiliate signup/dashboard, Stripe webhook bridge, Discord pings | Vercel/Netlify (Next.js) — this repo |
| Stripe | Payments, upsells, refunds, disputes | Stripe dashboard (see AUDIT-FINDINGS B3 — test keys) |
| Supabase Postgres | System of record (orders, practices, tokens, affiliates, KPIs) | Project `kbmvtawsdhmzjnsonazr`, eu-west-1 — see `01-data-model.md` |
| n8n | All backend automation (37 workflows) | `n8n.nieexpertsspain.com`, n8n 2.61.0 — see `03-flow-catalog.md` |
| Tally | Fase-1 contact form + Fase-2 EX-18 questionnaire | Forms `LZk4ov` / `aQvplB` — see `02-tally-forms.md` |
| Notion | CRM mirror of orders (`NIE Website Orders` DB) | Synced by W15 every 5 min |
| Resend | Transactional email | `RESEND_FROM_EMAIL` (see AUDIT-FINDINGS B1/B2) |
| Google Drive | Generated docs + situation guides storage/sharing | Per-practice subfolders |
| Cal.com | Consultation / group-call booking | Booking webhook -> W12 |
| Discord | Ops alerts + error routing | Channels per `03-flow-catalog.md` Discord routing table |
| WhatsApp agent | Customer Q&A, payment links, escalation | Live path = W07 + GeneratePaymentLink (agent name "Milene") |

### End-to-end data flow

```
checkout (Stripe)
  -> W01  creates order row + sends Fase-1 invite (chases non-
responders)
  -> customer fills Fase-1 (Tally LZk4ov)
  -> W02  opens one practice per person + mints Fase-2 token, starts
tier onboarding
  -> customer fills Fase-2 (Tally aQvplB)
  -> W03  after a human-like 45-60 min delay: generates EX-18 + Tasa
790/012,
          emails them, files to Drive
  -> W04  shares the situation/tier guide
  -> upsells          (W09: priority appointment / consultation)
  -> consultation     (W12: Cal.com booking recorded on practice)
  -> outcome/review   (W17 outcome check, W20 dissatisfaction
escalation)
  -> errors recovered (W05: delivery-failure recovery, 2-attempt cap ->
Discord)
  -> refunds/disputes (W10 chargeback, W16 refund handler)
Orders sync to Notion every 5 min (W15).
```

Key timing note: documents do not arrive instantly. W03 waits a randomized 45-60 min (with a 5-20 min interstitial email) so delivery feels human. This is by design, not a fault.

## Handoff & scope

Node Agency built the system and has **handed off the running tooling and all credentials** (Supabase, n8n, Stripe, Tally, Notion, Resend, Drive, Cal.com, Discord, WhatsApp agent) to NIE Experts. From go-live onward, **day-to-day operation is NIE's responsibility** — monitoring Discord error channels, clearing Tally corrections, approving payouts, and watching the n8n execution log. The audit (`AUDIT-FINDINGS.md`) was a one-time read-only review; fixes are proposed, not applied.

| What | Where it's documented |
|---|---|
| Data model / tables / enums | `01-data-model.md` |
| Tally form fields | `02-tally-forms.md` |
| Every n8n workflow + error handling | `03-flow-catalog.md` |
| Go-live switches (keys, overrides) | `07-go-live-checklist.md` |
| Audit findings (blockers/defects/hardening) | `AUDIT-FINDINGS.md` |
| Operator runbooks (Italian, per-area) | `docs/handoff/00-panoramica.md` … `13-*.md` |
| Affiliate go-live / collaudo | `docs/handoff/13-affiliate-collaudo-go-live.md` |

## Status: pre-launch on Stripe TEST keys

The system is currently running on **Stripe test keys** (`sk_test…`) with email redirected to a single test inbox (`EMAIL_OVERRIDE_TO`) and a sandbox Resend sender. It is **not yet taking real money or sending real customer email**. The exact switches to flip — Stripe live keys + webhook secret, unset `EMAIL_OVERRIDE_TO`, verified `RESEND_FROM_EMAIL`, `TURNSTILE_SECRET_KEY`, and clearing test-mode Notion rows — are tracked in `07-go-live-checklist.md` and itemized as blockers B1-B5 in `AUDIT-FINDINGS.md`.

# 01 · Data model — Supabase

> Project `kbmvtawsdhmzjnsonazr` (NIE Experts Site, eu-west-1). Verified live 2026-06-30. Postgres `public`: **14 tables + 2 views**. All timestamps `timestamptz`. Access is **service-role only** (app/n8n use the service key, which bypasses RLS) — see Security.

## Entity map

```
orders (Stripe checkout) ──< practices (one per person) ──< tokens
(Fase-2 magic link)
                                        │                 ├──<
upsell_items
                                        │                 └──<
practice_satisfaction_events
affiliates ──< affiliate_commissions    └─ drive ids · notion id ·
lifecycle status
affiliate_clicks (by referral_code)
admins · event_log · agent_events · processed_events · signup_attempts
· kpi_snapshots · partner_appointments(view) · agent_practice_v(view)
```

`orders.id` and `practices.order_id` are both **Stripe checkout session ids** (`cs_…`). No DB-level FK between them (see AUDIT-FINDINGS H7).

# Core tables

### orders — purchase / checkout intent

`id` text **PK** (Stripe session id) · `tier` `tier_t` · `amount_eur` · `payer_email` · `payer_whatsapp` · `persons` · `status` text (free-text, def `pending`) · `tier_label` · `consulenza_flag` · `appuntamento_flag` · `appointment_qty` · `client_id` · `notion_page_id`/`notion_synced_at` · `stripe_sessions` jsonb · `reminders_sent` · `phase1_sent_at` · `contacts_collected` · `created_at`/`updated_at`. Trigger `set_updated_at`.

### practices — core entity (one NIE case per person)

`id` uuid **PK** · `order_id` text (→orders, no FK) · `full_name`/`gmail`/`whatsapp_intl` (PII, NOT NULL) · `situation` `situation_t` · `urgency`/`urgency_weeks` · `tier` `tier_t` · `status` `status_t` (def `created`). Flags: `is_head_of_household`, `priority_appt_flag`, `consultation_flag`, `drive_access_active`(def true), `refund_eligible`. Drive: `drive_*_file_id`, `drive_*_folder_id`, `share_permission_id`, `share_status`. Lifecycle ts: `docs_delivered_at`, `guide_shared_at`, `outcome_at`, `escalated_at`, `id_doc_purged_at`… Review engine: `dissatisfaction_score`(def 0), `objection_count`, `dissatisfaction_tags` text[], `review_hard_blocked`, `review_block_reason`, `review_eligible`, `review_requested`, `review_completed`, `rejection_count`. `id_doc_url` (PII, purged by W21) · `notion_page_id` · `language`. Trigger `set_updated_at`.

### tokens — single-use Fase-2 access

`token` uuid **PK** · `practice_id` →practices CASCADE · `used`/`used_at` · `expires_at`(def now()+14d) · `attempts`(def 0) · `max_attempts`(def 5). Minted W02 → consumed W03 (`NIE_LIB_TokenConsume`) → re-minted W05 on contact correction.

### upsell_items

`id` uuid PK · `practice_id` →practices CASCADE · `type` `upsell_t` · `stripe_payment_id` **UNIQUE** · `people` · `created_at`.

### practice_satisfaction_events — review-engine stream (append-only)

`id` uuid PK · `practice_id` →practices CASCADE · `event_type` · `weight` int · `note` · `source`(def agent). **RLS DISABLED** (D2). Written by `log_satisfaction_event()`.

# Affiliate tables

### affiliates

`id` uuid PK · `email` **UNIQUE** · `referral_code` **UNIQUE** · `status`(pending/approved/rejected) · `stripe_account_id` · `stripe_payouts_enabled`(def false) · `country`(def IT) · `magic_link_jti` · `last_magic_link_at` · `approved_at`.

### affiliate_commissions

`id` uuid PK · `affiliate_id` →affiliates · `order_id` text **UNIQUE** (→orders, no FK) · `product` · `amount_eur` · `status`(def pending) · `paid_at` · `stripe_transfer_id` · `customer_name`. CHECK `status ∈`

{pending,approved,paying,paid,refunded,clawed_back}. Amounts: digital €5 / expert €10 / concierge-plus €20.

### affiliate_clicks

id uuid PK · referral_code · created_at. Joins affiliates by value (no FK).

## Supporting

- **admins** — magic-link admin accounts ( email UNIQUE, magic_link_jti); gates payout-approval page.
- **agent_events** — outbound queue for the WhatsApp/automation agent ( event, practice_id no FK, payload jsonb, delivery_status).
- **event_log** — workflow/audit log ( workflow, level, event, order_id, practice_id, message, detail jsonb).
- **processed_events** — idempotency ledger ( event_id PK, source event_source_t).
- **signup_attempts** — affiliate-signup rate-limit (hashed IP).
- **kpi_snapshots** — daily KPI rollup, one row per snapshot_date (PK), ~40 metrics. Written by W13.

## Views

- **agent_practice_v** — agent projection of practices (id→practice_id, English-izes situation). Read model for the automation agent.
- **partner_appointments** — partner slice where tier='concierge' OR priority_appt_flag (name, situation, appointment date/url, partner_notes).

## Enums (exact)

| Type | Values |
|---|---|
| tier_t | digital, expert, concierge |
| situation_t | studente, lavoratore, in_cerca_di_lavoro, situazione_specifica |
| status_t | created, phase2_sent, docs_pending, docs_delivered, guide_shared, appointment_booked, nie_obtained, nie_rejected, error_email, error_whatsapp, disputed, abandoned, contact_corrected, archived, refunded, awaiting_outcome, presumed_obtained, nie_rejected_consultation_done, closed, pdf_generation_failed |
| upsell_t | appointment, consultation |
| event_source_t | stripe, tally, whatsapp, calcom, resend, drive |

## Functions & triggers

- set_updated_at() — BEFORE UPDATE on orders, practices, kpi_snapshots.
- log_satisfaction_event(p_practice_id, p_event_type, p_note, p_source) → records/weights a satisfaction event, updates dissatisfaction_score/tags, applies hard-block rules, returns (dissatisfaction_score, review_hard_blocked, review_eligible, escalate). Core of the review-gating engine (feeds W20; called by agent tool W22).

## Security model

- **RLS enabled on 13 tables but ZERO policies** → anon/authenticated = deny-all. App/n8n use the **service-role key** (bypasses RLS). Implication: whole posture depends on the service key never leaking; an anon key would silently get nothing. No defense-in-depth layer.
- practice_satisfaction_events has **RLS disabled** (inconsistent — D2).
- Views run definer-rights and **expose practices PII** without own RLS — never grant to public roles.
- Hardening: AUDIT-FINDINGS H1, D2, H6, H7.

# 02 · Tally forms reference

> Verified live 2026-06-30. Two published forms feed the pipeline. Fase-1 collects per-person contacts after checkout; Fase-2 is the EX-18 questionnaire. (A third form `zxLpXM` "NIE correzione" exists, 0 submissions.)

| Form | Tally ID | Role | Keys on | Submits to |
|---|---|---|---|---|
| Fase-1 "Form Iniziale" | `LZk4ov` | post-checkout contact collection | `order_id` + `people`/`priority_count`/`tier` (URL params) | W02 webhook |
| Fase-2 "Questionario" | `aQvplB` | EX-18 data (anagrafici, indirizzo, arrivo) | `token` + `email` (URL params) | W03 webhook |

## Fase-1 — `LZk4ov` (single page, conditional hide-logic)

Renders up to 4 "Persona" blocks; extra personas hidden by the `people` URL param. Gmail-rule warning disables submit. No page breaks. Hidden / URL params: `order_id` (passthrough) · `people` (1/2/3; 4 shows all) · `priority_count` (hides priority question when IS 0) · `tier` (hides priority question when CONTAINS concierge). Per persona (x4): `Nome completo` (text) · `Gmail — deve finire con @gmail.com` (email; if not gmail -> warning + disable submit) · `Numero WhatsApp` (phone, intl, default IT) · `Situazione` (choice) · `Urgenza reale` (choice). Closing: `A chi spetta l'appuntamento prioritario?` (checkboxes Persona 1-4; hidden when `priority_count IS 0` OR `tier CONTAINS concierge`) · language question (fully hidden/dead) · `Consenso WhatsApp` (checkbox, required). Enums (map to backend, W02 parser): Situazione = `Studente` / `Lavoratore` / `In cerca di Lavoro` / `Altro`; Urgenza = `Entro 7 giorni` / `Entro 2 settimane` / `Entro 1 mese` / `Mi sto portando avanti`. Known issues (AUDIT-FINDINGS H2/H4): required-flags inconsistent across personas (P3 name/email optional, P4 phone optional); Gmail rule differs (`gmail.com` on P1/P4 vs `@gmail.com` on P2/P3); language question orphaned.

## Fase-2 — `aQvplB` (6 pages, no conditional logic)

On completion redirects to `wa.me/34670557796`. Hidden params: `token`, `email`. Page 1 Intro. Page 2 Situazione: `Qual è la tua situazione attuale in Spagna?` (choice, required): `Studente` / `Lavoratore con contratto o pre-contratto già firmato dal datore di lavoro` / `Sono in cerca di lavoro` / `Altro`. Page 3 Dati anagrafici: `Numero del documento d'identità` (text, req, max 20) · `Numero di NIE bianco (opzionale)` (min5/max20) · `Nome` (req, max50) · `Cognome` (req, max20) · `Secondo cognome (opzionale)` (max20) · `Sesso` (dropdown Uomo/Donna, req) · `Data di nascita` (date, req) · `Città di nascita` (req, max40) · `Paese di nascita` (req, max30, default Italia) · `Nazionalità` (req, max30, default Italiano) · `Primo nome del padre` (opt, max30) · `Primo nome della madre` (opt, max30). Page 4 Indirizzo: `Via e numero civico` (req, max60) · `Città` (req) · `Codice postale` (req, no validation) · `Provincia` (req). Page 5 Contatto & arrivo: `Numero di telefono` (phone, req, intl/IT) · `Quando sei arrivato in Spagna?` (date, req). Page 6 Consenso: GDPR checkbox (required). Thank-you: "Richiesta inviata!" + PDF-by-email + guide notice + WhatsApp CTA -> `wa.me/34670557796` (agent name "Milene"). Situazione enum (W03 parser): `Studente` / `Lavoratore con contratto o pre-contratto già firmato dal datore di lavoro` / `Sono in cerca di lavoro` / `Altro`. Known issues (AUDIT-FINDINGS H3/H5): `Codice postale`/`Città`/`Provincia` unvalidated; situazione enum differs from Fase-1 (parsed separately, fragile); Sesso binary only.

## Automation note — filling Fase-2 programmatically

Fase-2 date fields are uncontrolled React inputs (defaultValue + popup). Value injection and even real keystrokes set the visible text but not React state (validation reads empty -> "Inserisci un valore"). The reliable path is invoking the field's React onChange handler directly. Relevant only for automated E2E testing — see 08-e2e-test-guide.md.

# 03 · Flow catalog — n8n backend

> Verified live 2026-07-10. Instance `n8n.nieexpertsspain.com`, n8n 2.61.0. **38 workflows: 33 active, 5 inactive** (inactive: W16, old prototype agent set NIE_WA_Agent + GetPractice/BookAppointment/Escalate). This is the per-flow reference: what triggers each one, what it does, what it writes to Supabase (`01-data-model.md`), and how errors route to Discord. Core customer-path flows get a full block; the rest are one-liners. For the customer-facing narrative see `00-overview.md`;

> for Tally field details see `02-tally-forms.md`.

# Core flows

## W01_ContactCollection — `Q0X0hnsv0v8iCLqO`

- **Trigger:** webhook (Stripe order) + schedule (reminder cron).
- **Purpose:** create the order on payment + send Fase-1 invite; chase non-responders.
- **Key steps:** webhook -> Idempotency -> dedupe -> Extract Order -> Insert Order Row -> Send Phase-1 Email (LIB) -> Log -> (group-call invite branch: Wait 3min -> Email). Cron: Select Pending -> reminders exhausted? -> Mark abbandonato / Send Reminder -> increment.
- **Writes to Supabase:** `orders` insert; status `abbandonato`; reminder counter.
- **Discord:** none (errors via global handler).

**Update 2026-07-04 — smart invite timing.** The Fase-1 invite no longer fires at payment. W01 now parks on a Wait node and sends when the buyer FINISHES the funnel:

- `/grazie` fires a beacon (tagged `source=beacon`) that resumes the wait instantly → invite with the FINAL tier/params (order re-read from the DB).
- The upsell pages send a presence heartbeat every 25s ( `orders.funnel_last_seen`); on a timer resume W01 checks `GREATEST(updated_at, funnel_last_seen)` — buyer still present → wait another round (5-min cycles, 30-min ceiling); silent ≥90s → send now.
- `orders.payer_name` is stored at insert (personalized invite + reminder).
- The reminder/abandon cron branch is multi-order safe (RETURNING + direct-input refs; fixed 2026-07-04 after the first two-order night).

## W02_PracticeOpening — `BgqU8XFTFAy36xnB`

- **Trigger:** webhook (Fase-1 submit).
- **Purpose:** fan a group order into one practice per person, mint per-person tokens, start tier onboarding.
- **Key steps:** webhook -> Verify Tally HMAC secret (forged -> Reject) -> Idempotency -> Parse Group Members -> loop -> Insert Practice -> Insert Token -> Switch Tier -> [Digital: Mark `guide_shared` -> trigger W04] / [others: Send Phase-2 Email -> Mark `phase2_sent`]. **WA welcome no longer fires here** — it is stamped at **Fase-2** submit +~17min via W03 `Stamp Welcome Due` -> W07 `Welcome Poller` (the Fase-1 `Send WA Welcome` node is disabled).
- **Writes to Supabase:** `practices` insert; `tokens` insert; status `guide_shared`/`phase2_sent`.
- **Discord:** none.

## W03_DocGeneration — `Q6xqGYjWjN947mIW`

- **Trigger:** webhook (Fase-2 submit).
- **Purpose:** after a randomized human-like delay, generate Tasa 790/012 + EX-18, email them, file to Drive.
- **Key steps:** webhook -> Extract Token -> Idempotency -> TokenConsume (invalid -> Reject) -> status `docs_pending` -> Calc Delays (interstitial 5-20m, docs total 45-60m) -> Wait Interstitial -> Email Interstitial -> Wait Docs -> Parse Anagrafici -> Tasa Generate (790/012) [error -> stub + Discord `tasa-790-error`, continues] -> Fetch EX-18 Template -> Doc Fill EX-18 -> Drive subfolder + upload + share -> Email Documents (LIB) -> status `docs_delivered` -> Log -> trigger W04.
- **Writes to Supabase:** `practices` status `docs_pending` -> `docs_delivered`; Drive ids; Store Anagrafici; Mark PDF Failed / Reset Token on failure.
- **Discord:** `tasa-790-error` (direct); `ex-18-errors` (via DiscordError) for PDF fail; `drive-errors` + W05 ingest for share fail.
- **Note:** docs only arrive ~45-60 min after submit, **by design**.

**Update 2026-07-03 — instant guide + instant data.** The guide (W04 trigger) and the anagrafici parse/store now run IMMEDIATELY at Fase-2 submit, in parallel with the delay chain; only Tasa/EX-18 generation sits behind the 5-20m interstitial + 45-60m docs waits. Token-reject alerts are enriched (practice resolved even for used tokens).

## W04_GuideSharing — `IMApcnLrq6ly6zIy`

- **Trigger:** webhook (called by W02/W03/W05).
- **Purpose:** share the situation/tier guide (+ common video for non-Concierge) from Drive and email the client.
- **Writes to Supabase:** status `guide_shared`; clears its own idempotency row on DriveShare failure so it can retry.
- **Discord:** none.

### W05_DeliveryErrors — `gDwe8cq2vBO7zQmQ`

- **Trigger:** Resend bounce, WhatsApp fail (also routes Drive fail), Tally correction.
- **Purpose:** central delivery-failure recovery — flag failures, request corrected contact details, then re-mint tokens and re-fire the original delivery. 2-attempt cap then escalate to Discord.
- **Writes to Supabase:** status `error_email`/`error_whatsapp`; contact update; attempt reset; token invalidate + re-mint.
- **Discord:** `delivery-errors` (email/wa); `drive-errors` (drive).

### W08_CartAndRejectionRecovery — `dF74sBdgBOMazMuZ`

- **Trigger:** cart expired, rejection.
- **Purpose:** abandoned-cart drip (3 touches with paid-rechecks that stop if converted) + post-rejection consult nudge.
- **Writes to Supabase:** logs; reads `orders` paid status.
- **Discord:** none.

### W09_Upsell — `WKQMFXs8DnW1KDoY`

- **Trigger:** webhook (upsell paid).
- **Purpose:** process post-purchase upsells (priority appointment vs consultation), flag practice/order.
- **Key steps:** Idempotency -> Valid type? -> has `practice_id`? -> Insert Upsell Item -> Switch [appointment: set priority flag + update order] / [consult: set consult flag + update order + email Cal.com link] -> Discord upsell notify.
- **Discord:** `upsell-post-purchase`; missing `practice_id` -> `technical-errors`.

### W12_CalCom — `iwV1RuWFm6lSrjxc`

- **Trigger:** webhook (Cal.com booking).
- **Purpose:** record a booked consultation/group call on the practice + notify staff.
- **Writes to Supabase:** `practices` appointment fields.
- **Discord:** `consulenza`. The `Notify New Appointment` node (-> `new-appointments`) was **disabled 2026-06-12**: W12's Cal.com consultation/group-call pings now go only to `consulenza`, so they don't pollute `new-appointments`, which is a **manual NIE-cita partner-booking channel** (see 04-integrations.md and the "Channels NOT driven by n8n" note below).

### W15_OrdersNotionSync — `qhMEns7SevAkfP8m`

- **Trigger:** schedule every 5 min.
- **Purpose:** upsert orders into the Notion CRM DB (find by order id -> update/create).
- **Writes to Supabase:** `orders.notion_page_id`.
- **Discord:** none. No per-node error handling (AUDIT-FINDINGS H9).

### W20_DissatisfactionEscalation — `qnvyUqRFBsNI1XH4`

- **Trigger:** schedule every 5 min.
- **Purpose:** poll flagged-dissatisfied clients and alert the team.
- **Discord:** `cliente-insoddisfatto`.

### W21_IDDocRetentionPurge — `ViMUDf0kNTA4vyu2`

- **Trigger:** schedule daily 04:30.
- **Purpose:** GDPR retention — delete identity docs + folder past the window.
- **Note:** no own error/idempotency handling (AUDIT-FINDINGS H8).

## LIB subflows (shared)

- **NIE_LIB_SendEmail** — Resend send + test override "GO-LIVE SWITCH".
- **NIE_LIB_DriveShare** — share as reader.
- **NIE_LIB_TokenConsume** — single-use token consume.
- **NIE_LIB_Idempotency** — `event_id` upsert; called at head of every webhook flow.
- **NIE_LIB_ErrorHandler** — global errorTrigger -> log + Discord `technical-errors`.
- **NIE_LIB_DiscordError** — typed router: technical -> `technical-errors`, ex18 -> `ex-18-errors`, tasa -> `tasa-790-error`, delivery -> `delivery-errors`, drive -> `drive-errors`.
- **NIE_LIB_NotifyAgent** — outbox -> WhatsApp agent.
- **NIE_LIB_Log** — log rows.

## Other active flows (one-liners)

- **W01b_TierUpgrade** — funnel tier upgrade.
- **W06_AccessRevocation** — revoke access by status.
- **W07_WhatsAppCoord** — bridge to WhatsApp agent.
- **W10_Chargeback** — dispute -> `chargeback` channel.
- **W13_KPIDashboard** — daily KPI snapshot (writes `kpi_snapshots`).
- **W14_AppointmentsSync** — `practices` -> appointments view.
- **W22_LogSatisfaction** — agent tool `log_satisfaction_event`.
- **GeneratePaymentLink** — agent tool.
- **RequestCorrection** — agent tool `nie-wa-request-correction`. Lets the WhatsApp agent fix a customer's email/WhatsApp or retrigger document regeneration straight from chat: resolves the practice by phone number, confirms the full name, then either reuses the W05 correzione machine (email/whatsapp) or re-mints a Fase-2 link for a document redo (customer re-enters, W03 regenerates + emails).
- **LIB_DriveRevoke** — Drive access revocation helper.

## Activation state (updated 2026-07-10)

Reactivated at go-live:

- **W11_PartnerView** — **ACTIVE** (partner writeback + rebooking re-arm).
- **W17_OutcomeSweep** (ex OutcomeCheck) — **ACTIVE, repurposed 2026-07-10**: its 15:00 "ask" branch is retired (W07's EOD `eod_outcome_check` does the ask, then flips `appointment_booked -> awaiting_outcome` in the same execution). W17 keeps only the daily 09:00 sweep: `awaiting_outcome` + 3 days silence -> `presumed_obtained` (+ W06 revoke + Christian notice), referral emails (E/C at outcome+7d, Digital at guide+10d), and the satisfaction-gated review email #09.
- **W18_Phase2Reminder** — **ACTIVE** (daily 10:00; chases `phase2_sent` stuck > 3 days; token filter aligned to no-expiry decision).

Inactive (do not rely on):

- **W16_RefundHandler** — **deactivated 2026-07-01** (was not-ready: placeholder Discord webhook + pending prerequisites; AUDIT-FINDINGS D3 resolved).
- **W19_AffiliateNotify** — **deleted 2026-07-01** (dead PromoteKit→Discord; affiliate pings come from the website; AUDIT-FINDINGS D5 resolved).
- **NIE_WA_Agent** + **GetPractice** / **BookAppointment** / **Escalate** — the OLD prototype agent set, off by design. **The live escalation path is NOT this Escalate workflow**: `escalate_to_christian` is served by W07's `nie-wa-escalate` webhook (active), and inbound tools ride W07 `nie-wa-in` — both live.

## Error-handling model

Two layers:

1. **Passive global** `NIE_LIB_ErrorHandler` — any uncaught node error -> `errors` table + `#technical-errors`.
2. **Active typed** `NIE_LIB_DiscordError` — humanizes the error, enriches it with practice data, and routes by type.

Supporting guarantees:

- **Idempotency** via `NIE_LIB_Idempotency` (`event_id` upsert) at the head of every webhook flow. W04 clears its idempotency row on failure to allow retry. W15 uses Notion find-by-order-id upsert.
- **Tokens** single-use: mint W02, consume W03, re-mint W05.
- **Graceful degradation:** W03 Tasa stub (EX-18 still ships); LIB sends log failures instead of throwing; W05 recovery loop with a 2-attempt cap then escalate.

## Discord routing (n8n side)

| Source | Channel |
|---|---|
| W12 | `consulenza` |
| W12 (`new-appointments` node) | *(disabled 2026-06-12 — `new-appointments` is a manual channel, see note below)* |
| W09 | `upsell-post-purchase` |
| W09 missing practice | `technical-errors` |
| W20 | `cliente-insoddisfatto` |
| W10 | `chargeback` |
| W03 Tasa | `tasa-790-error` |
| W03 PDF | `ex-18-errors` |
| W03 drive | `drive-errors` |
| W05 email/wa | `delivery-errors` |
| W05 drive | `drive-errors` |
| LIB_DiscordError | typed (see above) |
| LIB_ErrorHandler | `technical-errors` |

**Channels NOT driven by n8n:** `stripe-checkouts`, `affiliation`, `affiliation-payout` (website side); `manual-review` / `whatsapp-errors` (WA agent); and **`new-appointments`** — a **manual human channel**: Christian posts NIE **cita** requests (name, surname, city, urgency) and the external booking partner ("CITA PARÁ NIE EXTRANJERÍA") books the in-person extranjería appointment. This is why W12's `new-appointments` node is disabled — its automated Cal.com consultation pings would pollute the partner's manual queue.

# 04 · External integrations

> Verified live 2026-06-30. Every external service the system talks to, and the exact contract for each. Cross-refs: data model `01-data-model.md`, flow catalog `03-flow-catalog.md`, error routing `05-error-runbook.md`, env/secrets `06-env-secrets.md`, audit `AUDIT-FINDINGS.md`. The website (this repo) and n8n (`n8n.nieexpertsspain.com`) each hold their own credentials — both talk to most of these services.

## Integration map

| Service | Used by | Purpose | Contract anchor |
|---|---|---|---|
| Stripe | website + n8n | Checkout, upsells, refunds, disputes, Connect payouts | Checkout Session id (`cs_…`) = order id |
| Resend | website + n8n | Transactional email | `RESEND_FROM_EMAIL` sender |
| Notion | n8n (W15) + website | CRM mirror of orders | `Supabase Order ID` = `cs_…` |
| Discord | website + n8n | Ops alerts + typed error routing | webhook URLs per channel |
| Google Drive | n8n | Generated-doc + guide storage/sharing | file/folder ids on `practices` |
| Cal.com | n8n (W12) | Consultation / group-call booking | booking webhook |
| WhatsApp agent | n8n (W07) | Customer Q&A, escalation, payment links | outbox `agent_events` |

## Stripe

**Checkout.** Stripe Checkout Sessions, API version pinned `2026-04-22.dahlia`. Mode is decided by the key: `isTestMode = STRIPE_SECRET_KEY.startsWith('sk_test')`.

- **Live mode** → line items reference hardcoded **live Product IDs**.
- **Test mode** → line items use inline `product_data` (sandbox products created ad hoc).

**Order id.** The Stripe Checkout Session id (`cs_…`) is the canonical order key. It becomes `orders.id` and `practices.order_id`, and in Notion it is triplicated into `Supabase Order ID` / `Client ID` / `Stripe Sessions` (H10). There is no DB-level FK between `orders` and `practices` (AUDIT-FINDINGS H7).

**Friend discount.** `resolveFriendDiscount()` returns **exactly one** of two shapes — never both:

- `discounts: [{ coupon }]` — when the order is referred (server-set `nie_affiliate_id`), is a **base** product, and a matching coupon exists. Coupon is server-resolved, never client-supplied.
- `allow_promotion_codes: true` — otherwise (lets the buyer type a public promo code).

Live friend coupons: `B2T1aavc` (digital), `ArPaJugz` (expert), `B1sqQ2mS` (concierge-plus, €10). Test mode uses the sandbox coupon set (`FRIEND_COUPON_*_TEST`). Discount values: Digital €5 / Expert €10 / Concierge+ €10 (changed from €20 on 2026-07-03 — commission stays €20).

**Affiliate updates (2026-07-03/04):** US ambassadors supported (accounts created under the FULL service agreement — the recipient agreement is not offered for US; EEA→US transfers allowed, ~0.25% cross-border fee platform-absorbed). Payout threshold removed (`AFFILIATE_PAYOUT_MIN_EUR` default 1). Dashboard: IBAN button handles add AND update (`account_update` link for completed accounts), connected-state copy, package column (Pacchetto) in the conversions table. Ambassadors receive a commission email on each referred base order; upsells generate no commission, no ping, no email.

### Main webhook — `api/stripe-webhook.js`

Raw body, signature-verified with `STRIPE_WEBHOOK_SECRET`. Durability rule: it **bridges to n8n first** (W01 / W01b / W09) and **returns HTTP 500 on a retryable failure before any non-idempotent side effect**, so Stripe retries and orders are never lost. Side effects run only after the bridge succeeds.

| Stripe event | What the webhook does |
|---|---|
| `checkout.session.completed` | 1) bridge to n8n W01 / W01b / W09 (durable, 500-on-retryable). Then: 2) Notion sync, 3) confirmation email (Resend), 4) Discord `notifyPurchase`, 5) affiliate Discord ping (referred orders), 6) commission ledger row (`affiliate_commissions` `pending`) |
| `checkout.session.expired` | bridge to **W08** (abandoned-checkout handling) |
| `charge.refunded` | commission **clawback** + bridge to **W16** (refund handler) |
| `charge.dispute.created` | Notion **Dispute** status + bridge to **W10** (chargeback) |

> Christian: a successful purchase is the trigger for everything downstream. If you see a `#stripe-checkouts` Discord ping but **no** Notion row within ~5 min, check the n8n W01/W15 executions — the bridge or the sync failed.

### Connect webhook — `api/stripe-connect-webhook.js`

Separate endpoint, **separate secret** (`STRIPE_CONNECT_WEBHOOK_SECRET`). Handles affiliate Connect account events: | Event | Action | |---|---| | `account.updated` | `setPayoutsEnabled` → writes `affiliates.stripe_payouts_enabled` |

`maxDuration` note: the main webhook runs many sequential awaits; AUDIT-FINDINGS D4 recommends adding `maxDuration: 30` in `vercel.json` (matches the invoices repo). The n8n bridge runs first/durably so orders survive, but email/Discord/commission can be cut on the default timeout.

---

## Resend (email)

Sender = `RESEND_FROM_EMAIL`, default `onboarding@resend.dev` — this is the **Resend sandbox sender and MUST change at go-live** (AUDIT-FINDINGS B1). If `RESEND_API_KEY` is missing, email is **skipped** (logged, not thrown).

**Test redirect.** `EMAIL_OVERRIDE_TO` redirects **all** outbound mail to one inbox (subject prefixed `[TEST→…]`). UNSET at go-live (B2).

**BCC.** `admin@nieexpertsspain.com` is BCC'd on the **order confirmation** and the **ambassador welcome** for observability. The **magic-link email has NO bcc** (it carries a single-use login token).

**Delay.** `EMAIL_DELAY_MINUTES` controls send delay: order/upsell confirmation default **5 min**, ambassador welcome **0**, magic-link **never delayed**.

| Email | Trigger | Branding / extras | E |
|---|---|---|---|
| Order / upsell confirmation | `checkout.session.completed` | navy/gold IT branding; no booking CTA (separate consulenza invite email since 2026-07-03) | y |
| Ambassador welcome | affiliate signup approved | referral link `nieexpertsspain.com/r/<code>` | y |
| Magic-link | affiliate/admin login | 15-min portal login token | **n** |

**Bounce webhook (REQUIRED, F2):** W05's bounce-recovery only works if Resend POSTs to `https://n8n.nieexpertsspain.com/webhook/nie-resend-bounce` (events `email.bounced`, `email.complained`). Configure in the Resend dashboard — not verifiable via the send-only API key.

The **n8n side** sends via `NIE_LIB_SendEmail` (also Resend) with its own test override — so both halves of the system respect test redirection independently.

---

# Notion (CRM)

Database **"NIE Website Orders"**, id `36dfd253-5094-80ae-83e3-fdd8a10425c4`. Written two ways:

- **W15** every 5 min — find by order id → update or create (full sync).
- **Website webhook** — dispute status on `charge.dispute.created`.

**Upsert key = `Supabase Order ID`** (the `cs_…` session id).

| Property | Type | Notes |
|---|---|---|
| Cliente | title | currently = **email** (no real name — H10) |
| Email | email | |
| Amount EUR | number | |
| People | number | |
| Appointment Qty | number | |
| Tier | select | option duplicates `Concierge⁺` vs `Concierge+` never match (D1) |
| Stato Pratica | select | `New` / `Welcome Inviato` / `Dispute` / `Rimborsato` / `Guida Condivisa` |
| Supabase Order ID | text | = `cs_…` (upsert key) |
| Client ID | text | = same `cs_…` (triplicated — H10) |
| Stripe Sessions | text | = same `cs_…` (triplicated — H10) |
| Consulenza / Appuntamento | checkbox | |
| Materiali da Consegnare | formula | **empty/dead** formula (H10) |
| Created / Last Update | date | |

Known issues: Tier select duplicates (D1); title=email + id triplication + dead formula (H10). See `AUDIT-FINDINGS.md`.

---

# Discord (ops alerts)

Server **"NIE Experts Spain"** . The **website** uses three webhooks; **n8n** uses its own typed error/event channels (see `05-error-runbook.md`).

**Website webhooks** | Env var | Channel | |---|---| | `DISCORD_WEBHOOK_URL` | `#stripe-checkouts` (also used for upsell embeds) | | `DISCORD_AFFILIATE_WEBHOOK_URL` | `#affiliation` | | `DISCORD_PAYOUT_WEBHOOK_URL` | `#affiliation-payout` |

### Discord routing (website side)

| Event | Channel | Webhook |
|---|---|---|
| New purchase | `#stripe-checkouts` | `DISCORD_WEBHOOK_URL` |
| Upsell | `#stripe-checkouts` (same — title differs only) | `DISCORD_WEBHOOK_URL` |
| Referred order | `#affiliation` | `DISCORD_AFFILIATE_WEBHOOK_URL` |
| Weekly payout review | `#affiliation-payout` | `DISCORD_PAYOUT_WEBHOOK_URL` |

> AUDIT-FINDINGS H11: landing and upsell **share one webhook** — `#stripe-checkouts` and `#upsell-post-purchase` are not actually separate repo-side; only the embed title differs.

**Full channel list (server-wide)**

- **Website Orders:** `stripe-checkouts`, `upsell-post-purchase`, `new-appointments`, `consulenza`, `affiliation`, `affiliation-payout`, `chargeback`
- **WhatsApp Agent:** `manual-review`, `whatsapp-errors`, `cliente-insoddisfatto`
- **Errors:** `technical-errors`, `ex-18-errors`, `tasa-790-error`, `drive-errors`, `delivery-errors`

> **`new-appointments` is a MANUAL channel, not automated.** Christian posts NIE **cita** booking requests here (name, surname, city, urgency) and the external booking partner ("CITA PARÁ NIE EXTRANJERÍA") books the in-person **extranjería** appointment. W12's Cal.com consultation/group-call pings deliberately go to `consulenza` (not here) so they don't pollute the partner's booking queue — that's why W12's `new-appointments` node is disabled. **Why it stays manual (by design):** the government cita needs human handling — choosing the office and coordinating with the partner — so this channel is intentionally *not* automated. **Optional (Christian's call, not built):** a light Notion helper could remove only the copy-paste — Christian fills the cita details on the Notion row, marks it *ready*, and n8n posts the formatted request here; the human still decides the office/details. Keeping it fully manual is an equally valid — and likely preferred — choice.

---

## Google Drive

Per-practice subfolders. n8n:

- Uploads the **EX-18** and **Tasa 790/012** PDFs and shares each as **reader** via `NIE_LIB_DriveShare`.
- Shares the situation **guide** in **W04**.
- **Purges ID documents** in **W21** (GDPR).

File and folder ids are persisted on `practices` (`drive_*_file_id`, `drive_*_folder_id`, `share_permission_id`, `share_status`). Failures route to `#drive-errors` (see runbook); W05 attempts recovery for delivery-side failures.

---

## Cal.com

Booking webhook → **W12** records the appointment on the practice (`partner_appointments` view surfaces it). The consultation upsell email carries the Cal.com booking link directly.

---

## WhatsApp

Outbound is the **outbox pattern**: n8n calls `NIE_LIB_NotifyAgent`, which enqueues to `agent_events` for an external WhatsApp agent (W07 coordinator). The Tally Fase-2 thank-you screen and the agent name "Milene" hand the customer off to **`wa.me/34670557796`** for live questions.

# 05 · Error runbook

> Verified live 2026-06-30. What to do when a Discord error channel lights up. Written for both Christian (run-the-business: which channel means what, what to click) and devs (the recovery internals). Cross-refs: integrations `04-integrations.md`, flow catalog `03-flow-catalog.md`, audit `AUDIT-FINDINGS.md`.

# The two-layer error model

The system alerts on errors in **two layers** that work together:

1. **Passive global net — `NIE_LIB_ErrorHandler`.** Wired as the error workflow on n8n executions. Any **uncaught** node failure anywhere falls through to here and posts to **`#technical-errors`**. This is the safety net: if nothing more specific caught it, it lands here.
2. **Active typed router — `NIE_LIB_DiscordError`.** Called deliberately at known failure points (PDF fill, Tasa, Drive, delivery) and routes to a **specific** channel with context. This is the diagnosis layer: the channel itself tells you what broke.

> Christian: think of `#technical-errors` as "something unexpected broke — a dev should look", and the named channels (`#ex-18-errors`, `#tasa-790-error`, etc.) as "this specific step failed — here's what to check." A named channel is more actionable; `#technical-errors` usually needs Node.

# Channel → meaning → cause → action

| Discord channel | What it means | Likely cause | What to do |
|---|---|---|---|
| `#technical-errors` | Any uncaught n8n node failure, or missing `practice_id` (W09) | Anything unhandled — schema drift, API down, bad payload | Open the n8n execution, read the error node, **retry** the execution; if it repeats, **escalate to Node** |
| `#ex-18-errors` | EX-18 PDF fill failed in **W03** | Template/field mapping issue or bad *anagrafici* | Check the Fase-2 data for that practice; effect: token reset + practice marked `pdf_generation_failed`. **Re-mint the token** if the customer must re-submit |
| `#tasa-790-error` | Tasa 790/012 generation failed in **W03** | Tasa endpoint/template issue | W03 **falls back to a stub** so the EX-18 still ships — the Tasa may be **missing**. Regenerate the Tasa manually / check the Tasa endpoint |
| `#drive-errors` | Drive share or upload failed (**W03 / W05**) | Drive quota or permissions | Check Drive quota/permissions. **W05 attempts recovery** automatically |
| `#delivery-errors` | Email bounce or WhatsApp failure after **≥2 correction attempts** (**W05**) | Bad email/phone, persistent bounce | Contact the client by **another channel**; verify the corrected contact details |
| `#cliente-insoddisfatto` | A client was flagged **dissatisfied** (**W20**) | Dissatisfaction score crossed threshold | **Human follow-up**. Note: review-gating may be intentionally blocking a review request for this client |
| `#chargeback` | A Stripe **dispute** opened (**W10**) | Customer disputed the charge | **Gather evidence and respond in Stripe** before the deadline |

# How recovery works automatically (devs)

The system is built to self-heal the common cases — do **not** rush to intervene before checking these:

- **Idempotency everywhere.** Every n8n webhook flow calls `NIE_LIB_Idempotency` first; `processed_events` is the ledger. A retried event does **not** double-process.
- **W05 delivery recovery.** Dedicated delivery-failure loop: up to **2 correction attempts**, then it escalates to `#delivery-errors`. Most transient bounces clear here without a human.
- **Token re-mint on contact correction.** When a contact is corrected, W05 re-mints the Fase-2 token so the customer can re-enter cleanly.
- **W03 Tasa stub fallback.** If Tasa generation fails, W03 substitutes a stub so the **EX-18 still ships** on time; only the Tasa is missing (flagged to `#tasa-790-error`).
- **Stripe webhook 500-on-retryable.** The Stripe webhook returns HTTP 500 on a retryable failure **before** non-idempotent side effects, so **Stripe retries** — orders are

never silently dropped (see `04-integrations.md`).

## What is NOT auto-handled — needs a human

| Situation | Why it needs you | Reference |
|---|---|---|
| **Tasa stub gap** | The stub keeps EX-18 on time but leaves the Tasa missing — someone must regenerate it | `#tasa-790-error` |
| **GDPR purge failure (W21)** | W21 has **no own error/idempotency handling and no retry** — a failed Drive delete only surfaces via the global net | AUDIT-FINDINGS **H8** |
| **Notion sync failure (W15)** | W15 has **no per-node error handling** — a sync can fail quietly; reconcile against Supabase | AUDIT-FINDINGS **H9** |
| **Anything in `#technical-errors`** | By definition uncaught — read the execution, retry, escalate to Node if it repeats | — |

> Christian: if a channel is in the "needs a human" table, the system will **not** fix it on its own. Everything else, give it a few minutes first — it usually recovers.

# 06 · Environment variables & secrets

> Verified live 2026-06-30. Every environment variable the website reads, what it does, and how sensitive it is for the test→live switch. Cross-refs: integrations `04-integrations.md`, go-live `07-go-live-checklist.md`, audit `AUDIT-FINDINGS.md`. **Sensitivity** = how much care the test/live transition needs: LIVE-critical (wrong value = real money or broken prod), changes behavior, low.

## Stripe

| Var | Purpose | Sensitivity |
|---|---|---|
| `STRIPE_SECRET_KEY` | Stripe API key. `sk_test…` prefix toggles **test mode** (inline products); live key → hardcoded live Product IDs | LIVE-critical — swap starts real charges |
| `STRIPE_WEBHOOK_SECRET` | Verifies `api/stripe-webhook.js` signatures | must match the live endpoint |
| `STRIPE_CONNECT_WEBHOOK_SECRET` | Verifies `api/stripe-connect-webhook.js` (separate endpoint) | must match the live Connect endpoint |
| `FRIEND_COUPON_DIGITAL` / `_EXPERT` / `_CONCIERGE` | Live friend-discount coupon ids per tier (`B2T1aavc` / `ArPaJugz` / `B1sqQ2mS`) | |
| `FRIEND_COUPON_DIGITAL_TEST` / `_EXPERT_TEST` / `_CONCIERGE_TEST` | Sandbox coupon ids used in test mode | test-only |

> **Changing the friend-discount coupons (no code change needed):** these env vars are currently **unset** — the code falls back to the hardcoded ids above ( `api/create-checkout-session.js` ~line 44). To switch to a different coupon: create the new coupon in Stripe (any duration — "once"/"forever" is irrelevant for one-time payments; just don't set a redemption limit), then set the matching `FRIEND_COUPON_*` env var in Vercel to the new coupon **ID** and redeploy. The env var wins over the hardcoded fallback. Never delete a coupon that is still referenced (env value, or the fallback when the env is unset) — referred-friend checkouts would start failing to apply the discount.

## URLs

| Var | Purpose | Sensitivity |
|---|---|---|
| `URL` | Base site URL | |
| `PUBLIC_BASE_URL` / `VERCEL_URL` | Public base for building links/redirects | |

## Email (Resend)

| Var | Purpose | Sensitivity |
|---|---|---|
| `RESEND_API_KEY` | Resend auth. **Missing → email skipped** (logged, not thrown) | |
| `RESEND_FROM_EMAIL` | Sender. Default `onboarding@resend.dev` is the **Resend sandbox** — change to a verified `@nieexpertsspain.com` sender at go-live (B1) | |
| `EMAIL_OVERRIDE_TO` | Redirects **all** mail to one test inbox (subject `[TEST→…]`). **UNSET at go-live** (B2) | — leaving it set means no customer gets mail |
| `EMAIL_DELAY_MINUTES` | Send delay (order confirmation default 5, welcome 0, magic-link never) | |
| `EMAIL_HEADER_STYLE` | Branded header style toggle | |

## Notion

| Var | Purpose | Sensitivity |
|---|---|---|
| `NOTION_API_KEY` | Notion integration token | |
| `NOTION_DATABASE_ID` | "NIE Website Orders" DB id | |

## n8n bridge

| Var | Purpose | Sensitivity |
|---|---|---|
| `N8N_W01_WEBHOOK_URL` | Order creation / Fase-1 invite bridge | |
| `N8N_W01B_WEBHOOK_URL` | Companion W01b bridge | |
| `N8N_W08_WEBHOOK_URL` | Abandoned-checkout (`checkout.session.expired`) | |
| `N8N_W09_WEBHOOK_URL` | Upsell handler | |
| `N8N_W10_WEBHOOK_URL` | Chargeback / dispute | |
| `N8N_W16_WEBHOOK_URL` | Refund handler | |
| `N8N_BRIDGE_SECRET` | Shared secret authenticating website→n8n bridge calls | shared secret — rotate if exposed |

## Discord

| Var | Purpose | Sensitivity |
|---|---|---|
| `DISCORD_WEBHOOK_URL` | `#stripe-checkouts` (also upsell embeds) | |
| `DISCORD_AFFILIATE_WEBHOOK_URL` | `#affiliation` | |
| `DISCORD_PAYOUT_WEBHOOK_URL` | `#affiliation-payout` | |

## Supabase

| Var | Purpose | Sensitivity |
|---|---|---|
| `SUPABASE_URL` | Project URL ( `kbmvtawsdhmzjnsonazr`) | |
| `SUPABASE_SERVICE_ROLE_KEY` | Service-role key — **bypasses RLS**; the entire security posture rests on it never leaking | highly sensitive |

## Affiliate payout

| Var | Purpose | Sensitivity |
|---|---|---|
| `CRON_SECRET` | Authorizes payout cron endpoints (`payout-notify`, `payout-approve`). **Fail-closed** | payout-critical |
| `PAYOUT_DRY_RUN` | `=1` is an **absolute no-money brake** (no transfers execute). Leave `=1` until really paying | money brake |
| `AFFILIATE_HOLD_DAYS` | Days a commission is held before payable (default **30**) | |
| `AFFILIATE_PAYOUT_MIN_EUR` | Minimum payout threshold (default **€1** — no threshold, Christian 2026-07-03) | |
| `AFFILIATE_AUTH_SECRET` | HMAC secret for magic-link / session / payout tokens. **Throws if unset** | |
| `AFFILIATE_ADMIN_TOKEN` | Admin auth for affiliate admin actions | |
| `AFFILIATE_DASHBOARD_URL` | Base URL for the affiliate dashboard links | |

## Security / captcha

| Var | Purpose | Sensitivity |
|---|---|---|
| `TURNSTILE_SECRET_KEY` | Cloudflare Turnstile for signup captcha. **Unset = dev passthrough** (no captcha); **set = fail-closed**. Prod must set (B5) | |

## Secrets handling

- Secrets live in **Vercel env** (website) and **macOS Keychain** (local), referenced by name only — **never committed** to the repo.
- **Rotate any secret shared during setup** — including Stripe test keys, Supabase PAT, Notion, Resend, and Tally keys pasted in chat — before or at go-live.
- The **n8n instance holds its own credentials internally** : Postgres (Supabase service), Resend, Google Drive, and Discord. These are separate from the website's env and are part of the handed-over tooling — rotate them on the same go-live pass.
- Service-role and HMAC secrets ( `SUPABASE_SERVICE_ROLE_KEY`, `AFFILIATE_AUTH_SECRET`, `N8N_BRIDGE_SECRET`, `CRON_SECRET`) are the crown jewels: a leak of any one weakens a whole subsystem. See AUDIT-FINDINGS **H1** for the RLS-zero-policies posture that makes the Supabase key especially load-bearing.

# 07 · Go-live runbook

> Verified live 2026-06-30 (D2/D4 status updated 2026-07-01). This is the step-by-step procedure to take the platform from **Stripe TEST** (current state) to **real customers and real money**. Written to be followed top to bottom by whoever runs the launch. Cross-refs: env reference `06-env-secrets.md`, integrations `04-integrations.md`, issues `AUDIT-FINDINGS.md`.

> **Node Agency prepared this; NIE Experts executes it (with Node's help).** Do the sections **in order**. Everything is reversible **except the Stripe live-key swap (§1), which is the moment real charges begin** — it is deliberately last. Budget ~30–45 min.

---

## 0. How env changes are made (read once)

Every "set X" below means: **Vercel → project `nie-experts-site` (team Node AI) → Settings → Environment Variables → Production**, add/edit the variable, Save.

**Important:** environment changes **do not take effect until the app is redeployed.** After you finish the env edits, do §8 (Redeploy) once — that applies all of them together. Editing an env var alone changes nothing on the live site until a redeploy.

Keep a scratch note of the old value of anything you change, so you can roll back fast.

---

## 1. Stripe — live keys, webhooks, products (blocker B3)

Do the read-only checks (1a–1d) first; do the **key swap (1e) last of everything** , after §2–§7

are green.

**1a. Get the live API key**

- Stripe Dashboard → toggle **View test data OFF** (top-right) → Developers → API keys → reveal **Secret key** (`sk_live_…`).
- This is the value for `STRIPE_SECRET_KEY`. The code auto-detects live vs test from the `sk_live` / `sk_test` prefix — nothing else toggles the mode.

**1b. Create the live main webhook**

- Developers → Webhooks (in **live** mode) → **Add endpoint**.
- URL: `https://nieexpertsspain.com/webhook/stripe`
- Events to send: `checkout.session.completed`, `checkout.session.expired`, `charge.refunded`, `charge.dispute.created`, `charge.dispute.closed`, `payment_intent.payment_failed`.
- After creating, reveal the **Signing secret** (`whsec_…`) → this is `STRIPE_WEBHOOK_SECRET`.

**1c. Create the live Connect webhook** (for affiliate payouts)

- Developers → Webhooks → **Add endpoint** → URL `https://nieexpertsspain.com/webhook/stripe-connect` → event `account.updated`.
- Signing secret → `STRIPE_CONNECT_WEBHOOK_SECRET`.
- Note: this endpoint returns 500 until its secret is set (so Stripe retries) — set it before relying on payouts.

**1d. Verify catalogue**

- [ ] Confirm the **live Product IDs** used in checkout exist in the live Dashboard (Products).
- [ ] Confirm the **live friend coupons** exist: `B2T1aavc` (digital) / `ArPaJugz` (expert) / `B1sqQ2mS` (concierge-plus, €10). If a coupon is missing the checkout still works — it silently falls back to the promo-code box (no crash) — but the referred discount won't auto-apply.

**1e. The swap (irreversible — do LAST)**

- [ ] Set `STRIPE_SECRET_KEY` = `sk_live_…`
- [ ] Set `STRIPE_WEBHOOK_SECRET` = the live main signing secret
- [ ] Set `STRIPE_CONNECT_WEBHOOK_SECRET` = the live Connect signing secret

**Verify (after §8 redeploy):** a real card at checkout completes; Stripe → Webhooks shows `200` on `checkout.session.completed`; the order appears in Supabase + Notion.

---

# 2. Email — Resend (blockers B1, B2)

**2a. Verify a sending domain** (if not already done)

- Resend → Domains → Add `nieexpertsspain.com` → add the shown **DNS records** (SPF/DKIM) at the domain registrar → wait for **Verified**.

**2b. Set the sender**

- [ ] Set `RESEND_FROM_EMAIL` to a verified sender, e.g. `NIE Experts Spain <noreply@nieexpertsspain.com>`.

- Why: it currently defaults to `onboarding@resend.dev` (Resend's sandbox) — customers would get mail from a random-looking sandbox address, and deliverability suffers.

**2c. Turn off the test redirect (critical)**

- [ ] **Delete `EMAIL_OVERRIDE_TO`.** While it is set, **every** email (order confirmations, magic links, ambassador welcomes) is redirected to that one test inbox with a `[TEST→…]` subject prefix, so **no real customer receives anything**.

**Verify:** after redeploy, a test order confirmation arrives at the real buyer address (BCC lands at `admin@nieexpertsspain.com`).

---

# 3. Affiliate payout (business decisions)

These control when/whether affiliate money moves. Safe defaults shown.

- [ ] `AFFILIATE_HOLD_DAYS` — maturity window before a commission is payable (default **30**).
- [x] `AFFILIATE_PAYOUT_MIN_EUR` — **no threshold** (default now €1; decided by Christian 2026-07-03). Every matured commission pays out on Friday.
- [ ] `CRON_SECRET` — must be set; the Friday payout cron and the approve→execute call are **fail-closed** without it.
- [ ] Confirm `admin@nieexpertsspain.com` is in the `admins` table (gates the payout-approval page).
- [ ] **`PAYOUT_DRY_RUN` — leave `=1`** (absolute no-money brake) until you actually intend to pay affiliates. Flip to `0` only when you're ready for the first real payout run.

Payout flow reminder: Friday 10:00 Madrid → Discord `#affiliation-payout` review card → admin logs in + approves → transfers execute (idempotent claim-fence, so no double-pay). See `docs/handoff/13-affiliate-collaudo-go-live.md`.

---

# 4. Security

- [ ] **Set `TURNSTILE_SECRET_KEY`** — Cloudflare Dashboard → Turnstile → create a widget for `nieexpertsspain.com` → copy the **Secret key**. Unset = the signup captcha is in dev passthrough (bots can sign up).
- [ ] **Rotate every secret shared during setup** : Supabase PAT ( `sbp_…`), Notion key ( `ntn_…`), Resend key (`re_…`), Tally key ( `tly-…`), and the shared Stripe **test** keys. Rotate at the source (each provider's dashboard) and update Vercel/n8n where used.
- [ ] Review the **RLS posture** (AUDIT-FINDINGS **H1**): access is service-role-only (RLS is on with no policies; the service key bypasses it). Confirm `SUPABASE_SERVICE_ROLE_KEY` is not exposed anywhere public, and the views `agent_practice_v` / `partner_appointments` (which expose PII) are not granted to public roles.

---

# 5. Data hygiene (Notion)

- [ ] **Clear test orders**: Notion "NIE Website Orders" → filter `Supabase Order ID` contains `cs_test_` → archive/delete those rows so the live board starts clean (B4).
- [ ] **Merge duplicate Tier options** (D1): the select field has `Concierge⁺` (superscript) **and** `Concierge+` (ASCII) — they never match and split orders. Also `Expert + Appuntamento + Consulenza` vs `Expert + Consulenza + Appuntamento`. Edit the property → merge each pair into one canonical option, and confirm what the W15 sync writes matches the survivor.

---

# 6. n8n reconciliation

### Workflow activation state (the flip — done 2026-07-10)

- [x] **W11_PartnerView** — **ACTIVE** (partner appointment writeback + rebooking re-arm).
- [x] **W17_OutcomeSweep** (ex OutcomeCheck) — **ACTIVE, repurposed**: 15:00 "ask" branch disabled (W07's EOD sends `eod_outcome_check` and flips `appointment_booked → awaiting_outcome`); keeps the daily 09:00 sweep — `presumed_obtained` after 3 days silence (+ W06 revoke + Christian notice), referral emails, satisfaction-gated review email #09.
- [x] **W18_Phase2Reminder** — **ACTIVE** (daily 10:00 chase of `phase2_sent` stuck > 3 days; token filter aligned to the no-expiry decision).
- [x] **W07_WhatsAppCoord** — ACTIVE (live agent bridge — inbound tools, escalation, read API, outbound pollers). **Never deactivate.**

- Everything else per `docs/STATUS.md`; intentionally inactive: W16 (below), old prototype agent set (NIE_WA_Agent + GetPractice/BookAppointment/Escalate).

- [x] **W16_RefundHandler** (D3) — **deactivated 2026-07-01** (it had a placeholder Discord webhook + pending prerequisites, so it was not ready). To enable refund automation later: replace the placeholder chargeback webhook URL, set `N8N_W16_WEBHOOK_URL`, re-enable its Discord node, then activate.
- [x] **W19_AffiliateNotify** (D5) — **deleted 2026-07-01** (dead PromoteKit→Discord flow; affiliate pings come from the website).
- [ ] (Reference) `#new-appointments` stays a **manual** partner-booking channel — no change; see `04-integrations.md`.

---

## 7. Already applied by Node Agency (no action)

- **D4** — `api/stripe-webhook.js` `maxDuration: 30` added to `vercel.json` (committed 2026-07-01) so the order side effects (n8n bridge → Notion → Resend → Discord → commission) can't be cut on the default timeout. Live after the next production deploy.
- **D2** — RLS enabled on `practice_satisfaction_events` (applied live 2026-07-01; migration `20260701120000_rls_practice_satisfaction_events.sql`). All base tables now have RLS on.

---

## 8. Redeploy to apply

Env-var edits are inert until a redeploy. Do this once, after §1–§6 are set:

- [ ] Vercel → project `nie-experts-site` → Deployments → **Redeploy** the latest production deployment (or push any commit to `main`).
- [ ] Wait for **Ready**.

---

## 9. Post-go-live smoke test

Right after the swap, prove the live path with one real, low-cost transaction:

- [ ] Buy the cheapest package with a **real card** → checkout completes.
- [ ] Stripe → Payments shows the charge; Webhooks shows `200` on `checkout.session.completed`.
- [ ] Order confirmation email arrives (BCC at `admin@`); `#stripe-checkouts` Discord ping fires; the order appears in Notion (`Stato Pratica = New`) within ~5 min.
- [ ] Walk Fase-1 → Fase-2 once to confirm the live token + doc pipeline (docs arrive ~45–60 min later by design — see `08-e2e-test-guide.md`).
- [ ] **Refund** the test charge in Stripe → confirm the commission clawback + Notion status behave.

**Rollback:** if anything is wrong, set `STRIPE_SECRET_KEY` back to the `sk_test_…` value and redeploy — you're instantly back in test mode, no real charges. (Re-set `EMAIL_OVERRIDE_TO` too if you want to stop live mail while debugging.)

---

### Final gate

- [ ] §2–§7 confirmed → perform the **Stripe live-key swap (§1e) last** → redeploy (§8) → run the smoke test (§9). This is the irreversible moment real charges begin.

# 08 · End-to-end test guide

> Verified live 2026-06-30. A hands-on walkthrough Christian can run himself **in TEST mode — no real money moves**. Each step lists the trigger and a "what to expect" callout. Cross-refs: flow catalog `03-flow-catalog.md`, Tally fields `02-tally-forms.md`, audit `AUDIT-FINDINGS.md`.

> The system is on Stripe **test keys** today, so this whole flow is safe to run end-to-end. No card is charged and no real payout moves.

## Step 1 — Buy a package

On the site, pick a package and check out with the Stripe **test card**: `4242 4242 4242 4242` · any **future** expiry · any **CVC** · any postcode.

#### What to expect

- Redirect to the thank-you page.
- An **order confirmation email** arrives (BCC `admin@nieexpertsspain.com`) — in test mode it's redirected by `EMAIL_OVERRIDE_TO`, subject prefixed `[TEST→…]`.
- A **`#stripe-checkouts`** Discord ping.
- The order appears in **Notion** with **Stato Pratica = New** within **~5 min** (W15 sync runs every 5 min).

> If the Discord ping fires but no Notion row shows up after ~5 min, check the n8n W15

> execution — that's the sync step.

## Step 2 — Fase-1 contact form

The **Fase-1 email** arrives. Open the link and, for **each person**, fill: Gmail (must end `@gmail.com`), WhatsApp (`+39…`), situation, urgency. Submit.

**What to expect**

- One **practice is created per person** (W02).
- Then either a **Fase-2 email with a personal token link** (single practice / Expert · Concierge tiers) **or the guide directly** (Digital tier).

## Step 3 — Fase-2 questionnaire (EX-18 data)

Open the Fase-2 link and complete all **6 pages**: situation, *anagrafici*, address, phone, arrival date, consent. Submit.

**What to expect**

- Redirect to **WhatsApp** (`wa.me/34670557796`).
- **W03 starts** the document-generation flow.

## Step 4 — Document generation (DELAYED BY DESIGN)

Generation is **deliberately slow** to feel human — this is not a fault.

**What to expect (timeline)**

- **5–20 min:** an interstitial "we're working on it" email.
- **~45–60 min total:** the **EX-18** and **Tasa 790/012** PDFs arrive by email and are filed to the client's **Drive folder**.
- Notion **Stato Pratica advances**.
- The **guide is shared** (W04).

## Step 5 — Verify in n8n

Open the **W03 execution** in n8n. It should show: webhook received → **waiting** (the delay nodes) → **success**.

**What to expect**

- `#ex-18-errors` and `#tasa-790-error` are **clean** (no posts for your run).
- If `#ex-18-errors` fired: check the Fase-2 *anagrafici* for that practice; the practice is marked `pdf_generation_failed` and the token is reset (re-mint if needed). See `05-error-runbook.md`.
- If `#tasa-790-error` fired: W03 ships the EX-18 with a **Tasa stub**, so the Tasa may be missing — regenerate it manually.

## Speeding up the test

To verify generation without the 45–60 min wait, the **Wait nodes (Calc Delays)** in n8n **W03** can be **temporarily shortened**. Caveat: this affects only **NEW** submissions — **in-flight** practices keep their original timing. Restore the delays before go-live so real deliveries feel human.

## Automation caveat (scripted tests only)

If you script the Fase-2 step, note the date-field quirk from `02-tally-forms.md`: Fase-2 date fields are **uncontrolled React inputs** (defaultValue + popup). Value injection and even real keystrokes set the visible text but **not React state**, so validation reads empty → "Inserisci un valore". The reliable path is invoking the field's React `onChange` handler directly.

---

**Remember:** in TEST mode no card is charged and no real payout moves.

# 09 · Making changes after handoff (via Bento / an AI assistant)

> How to connect an MCP-capable AI assistant ("Bento") to the NIE systems so you can safely make changes after handoff — edit the automation flows, the website code, the database, and the CRM. Written tool-generically: the URLs/keys below apply whatever the assistant; the exact "add a connector" screen is Bento-specific.

## What you connect (and what it lets you change)

| System | Connect via | Lets you |
|---|---|---|
| **n8n** (automation flows W01–W22) | **n8n MCP** | list / read / edit / validate / deploy workflows |
| **Website code** | **GitHub repo** | change checkout, emails, affiliate, pages |
| **Supabase** (database) | **Supabase MCP** or a PAT | read schema, run SQL, add migrations |
| **Notion** (CRM) | **Notion integration** | read / update the orders board |

---

## 1. n8n — the automation flows

- **Instance:** `https://n8n.nieexpertsspain.com`
- **Get an API key:** n8n → **Settings** → **n8n API** → **Create an API key** . Copy it (shown once).
- **Connect (MCP config example):**

```
{
  "mcpServers": {
    "n8n-mcp": {
      "command": "npx",
      "args": ["n8n-mcp"],
      "env": {
        "N8N_API_URL": "https://n8n.nieexpertsspain.com",
        "N8N_API_KEY": "<your n8n API key>"
      }
    }
  }
}
```

- Now the assistant can list workflows, read a flow's structure, edit nodes, validate, and activate/deactivate. **Flow reference:** `03-flow-catalog.md`.
- **Safe practice:** duplicate a flow (or test it) before changing a live one; the MCP can `validate` before you activate. Snapshot the workflow JSON first so you can restore.

## 2. Website code — the GitHub repo

- **Repo (private):** `https://github.com/nieexpertsspain/nie-experts-site`
- **Connect:** give Bento's GitHub connector access to that repo (or use a GitHub Personal Access Token with `repo` scope, or the `gh` CLI signed into the `nieexpertsspain` org).
- **Deploy model:** Vercel **auto-deploys on push to `main`** (project `nie-experts-site`, team Node AI). So the change workflow is:
  1. branch → make the change → open a **PR** → review → **merge to `main`** → Vercel deploys automatically.
  2. Verify the deployment is **Ready** in Vercel.
- **Never commit secrets.** All keys live in **Vercel → Settings → Environment Variables** (and in n8n's own credentials), never in git. Env reference: `06-env-secrets.md`.

## 3. Supabase — the database

- **Project ref:** `kbmvtawsdhmzjnsonazr` (region eu-west-1).
- **Connect:** the **Supabase MCP** (OAuth), or a **Personal Access Token** created at `https://supabase.com/dashboard/account/tokens`.
- Lets the assistant read the schema, run read queries, and (carefully) DDL. **Schema reference:** `01-data-model.md`.
- **Schema changes go through migrations:** add a timestamped file in `supabase/migrations/` (e.g. `20260701120000_<name>.sql`) so the change is tracked and reproducible — see the D2 RLS migration as a template.

## 4. Notion — the CRM

- **Database:** "NIE Website Orders" (id `36dfd253-5094-80ae-83e3-fdd8a10425c4`).
- **Connect:** create an internal integration at `https://www.notion.so/my-integrations`, copy its token, then **share the database with that integration** (Notion → the DB → ⋯ → Connections → add it).
- Lets the assistant read/update the orders board (Tier, Stato Pratica, etc.). **Mapping:** `04-integrations.md`.

---

## The safe change workflow (always)

1. **Stay in test mode** until a deliberate go-live. The whole system keys off the Stripe `sk_test_`/`sk_live_` prefix — see `07-go-live-checklist.md`. Don't touch live Stripe keys for routine changes.
2. **Code:** branch → change → PR → merge to `main` → auto-deploy → verify. Small, reviewable commits.
3. **n8n:** validate + test a flow before activating; snapshot before editing a live one.
4. **DB:** never edit data blindly; schema changes as migrations.
5. **Secrets:** generate/rotate at the source (provider dashboard) and set them in Vercel/n8n env — never in git or chat.
6. **When unsure what a piece does,** read the matching doc in this set first ( `00`–`08`, `AUDIT-FINDINGS`).

## Where to generate each credential

| Credential | Where |
|---|---|
| n8n API key | n8n → Settings → n8n API |
| GitHub access | GitHub org `nieexpertsspain` (connector or PAT, `repo` scope) |
| Supabase PAT | supabase.com/dashboard/account/tokens |
| Notion token | notion.so/my-integrations (then share the DB) |
| Vercel env vars | Vercel → project `nie-experts-site` → Settings → Environment Variables |

## Guardrails recap

- Deploys are automatic on `main` — treat `main` as production.
- Site stays in **Stripe test mode** until go-live ( `07`).
- Rotate any key that's ever been shared; keys live in env, not git.
- Open items / known issues to be aware of before changing things: `AUDIT-FINDINGS.md`.

# NIE Experts Spain — Live System Audit (findings)

> Audited 2026-06-30 by Node Agency against the **live** surfaces (Supabase `kbmvtawsdhmzjnsonazr`, Tally forms `LZk4ov`/`aQvplB`, Notion `NIE Website Orders`, n8n `n8n.nieexpertsspain.com`, Resend, and the deployed website code). Read-only audit — nothing was changed in production. Fixes are proposed, not applied.

Severity: go-live blocker · defect · hardening / quality

---

### Go-live blockers (expected pre-launch — see `07-go-live-checklist.md`)

| # | Finding | Surface | Action |
|---|---|---|---|
| B1 | `RESEND_FROM_EMAIL` defaults to `onboarding@resend.dev` (Resend sandbox) | code/Resend | Set to a verified `@nieexpertsspain.com` sender |
| B2 | `EMAIL_OVERRIDE_TO` is set — **every** email redirects to one test inbox (subject prefixed `[TEST→…]`) | code | Unset at go-live |
| B3 | Stripe running on **test keys** (`sk_test…`) | Stripe | Swap to live keys + live webhook secrets |
| B4 | Notion `NIE Website Orders` holds **test-mode orders** (`cs_test_…`) | Notion | Archive/clear test rows before launch |
| B5 | `TURNSTILE_SECRET_KEY` unset → signup captcha in dev passthrough | code | Set in prod |

## Defects (fix before or shortly after launch)

| # | Finding | Surface | Proposed fix |
|---|---|---|---|
| D1 | **Notion Tier duplicates** — `Concierge⁺` (superscript ⁺, pink) vs `Concierge+` (ASCII +, gray) are distinct options that never match; `Expert + Appuntamento + Consulenza` vs `Expert + Consulenza + Appuntamento` are the same bundle twice | Notion | Merge to one canonical option pe confirm what W15 writes |
| D2 | `practice_satisfaction_events` has **RLS DISABLED** while parent `practices` has RLS enabled — objection notes (PII-adjacent) unprotected if any non-service key reaches the DB | Supabase | `ALTER TABLE public.practice_satisfacti ENABLE ROW LEVEL SECURITY;` |
| D3 | **RESOLVED (2026-07-01)** — W16_RefundHandler was `active=true` but genuinely not-ready (placeholder Discord webhook + pending prerequisites), so it was **deactivated** to match its `[INATTIVO]` label. To enable refund automation later: replace the placeholder chargeback webhook URL, set `N8N_W16_WEBHOOK_URL`, re-enable its Discord node, then activate. | n8n | Done |
| D4 | `api/stripe-webhook.js` has **no `maxDuration`** in `vercel.json` while it runs many sequential awaits (n8n bridge → Notion → Resend → Discord → commission). The n8n bridge runs first/durably so orders survive, but email/Discord/commission can be cut on the default timeout | code | Add `maxDuration: 30` for the v (matches invoices repo `b983e29`) |
| D5 | **RESOLVED (2026-07-01)** — dead **W19_AffiliateNotify** (PromoteKit → Discord) **deleted**. Affiliate Discord pings come from the website (`DISCORD_AFFILIATE_WEBHOOK_URL` / `DISCORD_PAYOUT_WEBHOOK_URL`). | n8n | Done |

## Hardening / quality

| # | Finding | |
|---|---|---|
| H1 | **RLS enabled but zero policies** on 13 tables. Safe only because access is service-role-only (bypasses RLS). The whole model rests on the service key never leaking — no defense-in-depth. Views `agent_practice_v` / `partner_appointments` expose `practices` PII (name, gmail, whatsapp) without their own RLS | Supabas |
| H2 | Tally Fase-1 **inconsistent required-flags** across personas (P3 name/email optional, P4 phone optional); **Gmail validation differs** (P1/P4 check `gmail.com`, P2/P3 check `@gmail.com`) | Tally — **11, API-** personas Gmail + (Christia confirme Defense- form: W `Members` name+g enforces `NOT NUL` `fase1_c` escalatio person is rule: the label-tex (Tally va only); re server-si personas |
| H3 | Tally Fase-2 `Codice postale`, `Città`, `Provincia` have **no validation** (postal code free-text despite "5 cifre" helper) | Tally |
| H4 | Tally Fase-1 "lingua preferita" question is **fully hidden/orphaned** (dead field); Fase-2 father/mother names optional while rest of anagrafici required | Tally |
| H5 | **Situazione enum mismatch** between forms (Fase-1 `Lavoratore` / `In cerca di Lavoro` vs Fase-2 `Lavoratore con contratto…` / `Sono in cerca di lavoro`). Parsed separately by W02 vs W03 so it works today, but fragile | Tally/n8 |
| H6 | `orders.status` is **free-text** (no CHECK/enum) unlike the enum-typed `practices.status` — typos not rejected | Supabas |
| H7 | **Missing FKs**: `practices.order_id→orders`, `affiliate_commissions.order_id→orders`, `agent_events.practice_id/event_log.practice_id→practices`, `affiliate_clicks.referral_code→affiliates` | Supabas |
| H8 | **W21 GDPR purge** has no own error/idempotency handling — a failed Drive delete only surfaces via the global ErrorHandler, not retried | n8n |
| H9 | **W15 Notion sync** has no per-node error handling | n8n |
| H10 | Notion `Cliente` title = the **email** (no real name); session id **triplicated** into `Supabase Order ID`/`Client ID`/`Stripe Sessions`; `Materiali da Consegnare` formula is **empty/dead** | Notion |
| H11 | Discord **landing + upsell share one webhook** (`DISCORD_WEBHOOK_URL`) — `#stripe-checkouts` and `#upsell-post-purchase` not actually separate repo-side (only embed title differs) | code/Dis |
| H12 | Stale **PromoteKit / LinkJolt** comments remain in code (decommissioned; Supabase ledger is canonical) | code |

# Verified healthy (no action)

- **Money paths sound:** payout = admin-session + signed-token + `CRON_SECRET` gated, `PAYOUT_DRY_RUN` brake, claim-fence ( `pending→paying→paid`), Stripe idempotency keys, self-referral guard fails closed. Friend discount only granted on **server-set** `nie_affiliate_id` (not client-spoofable).
- **Idempotency everywhere:** every n8n webhook flow calls `NIE_LIB_Idempotency` first; the Stripe webhook bridges to n8n before any non-idempotent side effect and returns 500 on retryable failure so Stripe retries.
- **Graceful degradation:** W03 Tasa falls back to a stub so EX-18 still ships; Resend/Drive/Agent sends log failures instead of throwing; W05 is a dedicated delivery-failure recovery loop (2-attempt cap → escalate).
- **Two-layer error alerting:** passive global `NIE_LIB_ErrorHandler` (→ `#technical-errors`) + active typed `NIE_LIB_DiscordError` routing per-error-channel.

# Open operational flags (updated 2026-07-04)

The complete list of known-open items after the July hardening round. Nothing outside this list is silently unhandled.

| # | Flag | Owner | Detail |
|---|---|---|---|
| F1 | **`product` metadata on the 4 post-buy Payment Links** | Christian | The post-buy CRM sync (practice → parent order – Notion row) is built but inert until each Payment Lir carries metadata key `product` = `expert-to-consulenza-postbuy` / `expert-to-appuntament postbuy` / `digital-to-consulenza-postbuy` / `digital-to-appuntamento-postbuy` |
| F2 | **Resend bounce webhook** | Niek/Christian | W05's bounce-recovery loop only works if Resend POSTs bounces to n8n. Dashboard: Resend → Webl → Add → `https://n8n.nieexpertsspain.com/webhook/r resend-bounce`, events `email.bounced` + `email.complained`. Unverifiable via API (send-on key) |
| F3 | Notion Tier select duplicates (`Concierge⁺` vs `Concierge+`) | Christian | Manual merge in Notion (D1) |
| F4 | Go-live checklist execution | Niek+Christian | `07-go-live-checklist.md` — live keys, sender domain, unset override, cleanup `--execute`, smoke incl. refund |
| F5 | Credential expiries | Niek | n8n API key ~2026-08-01; Bento `nie_agent` JWT 2027-01-01 (and never revoke the Supabase legacy HS256 key while Bento runs on it) |
| F6 | Smart invite wait — full-loop live test | next test buy | Components individually verified (endpoint one-shc workflow graph, expressions); the full pay → linger beacon → instant-invite dance awaits one real purch |
| F7 | Cita automation | parked | `#new-appointments` stays manual by design; optic Notion-driven auto-post if ever wanted |

## Accepted by design (documented, not bugs)

- Tab-closer who returns hours later via the Stripe receipt and upgrades → invite email copy may show the pre-upgrade package; the practice always self-corrects (W02 reads the live order row at form submit).
- Anonymous cart abandoners are greeted "Buongiorno," (no name exists pre-Fase-1 without checkout name capture).
- US affiliate payouts carry Stripe's ~0.25% cross-border fee (platform-absorbed).
- Fase-1 reminder greets the payer only — group members' names do not exist until Fase-1 is submitted.
- Funnel campers hit the 30-minute invite ceiling; data still self-corrects at submit.