FREE GUIDE · HUSNAINBUKHARI.COM

GoHighLevel API Integration Kit

A developer kit for the GoHighLevel v2 API — OAuth setup, core endpoints, rate-limit handling, webhook patterns, and n8n integration recipes, plus a GHL-vs-HubSpot decision table.

By Syed Husnain Haider Bukhari — AI engineer & founder, Revolutionary Technologies · www.husnainbukhari.com

GoHighLevel API Integration Kit

This kit is the reference I hand to engineers when they start a GoHighLevel build on my team. It is deliberately practical: the OAuth steps that actually work, the endpoints you will use every day, the rate-limit tactics that keep production quiet, and a GoHighLevel-vs-HubSpot decision table you can use live in a client call. It expands on the blog post rather than repeating it, so keep both open.

1. The mental model: Agency, Location, and tokens

GoHighLevel is multi-tenant by design. Get this hierarchy right and everything else falls into place.

- **Agency (Company):** the top-level account. An agency owns many locations. Agency-level tokens can provision sub-accounts and mint location tokens.
- **Location (Sub-account):** where the real data lives — contacts, opportunities, calendars, conversations, custom fields. Most integration work happens here with a location-scoped token.
- **Marketplace app:** the OAuth client you register. Users install it, you receive tokens, and scopes on the app control what you can read and write.

> Rule of thumb: if you are reading or writing business data, you want a Location token. If you are provisioning accounts or reading agency-wide config, you want a Company token.

2. v2 OAuth setup, step by step

1. Register a marketplace app and add your redirect URI. Save the client ID and client secret in a secret manager, never in code.

2. Redirect the user to the authorization URL with `client_id`, `redirect_uri`, `response_type=code`, and a space-separated `scope` list.
3. Exchange the returned `code` at the token endpoint for an access token (~24h life) and a long-lived refresh token.
4. Store the refresh token per install. Rotate the access token on a schedule before expiry, not reactively after a 401.
5. For agency installs, call the locations token endpoint to swap the company token for a location token per sub-account.
6. On every request, send `Authorization: Bearer <token>` and the dated `Version` header.

Minimum scopes by use case

Use case	Scopes to request
Lead ingestion	contacts.write, contacts.readonly
Pipeline / deal sync	opportunities.write, opportunities.readonly
Booking agent	calendars.readonly, calendars/events.write
Messaging / inbox	conversations.write, conversations/message.write

Request the narrowest scopes that do the job. Every scope you add is a permission a security-conscious client will ask you to justify.

3. Core object cheat sheet

Object	Belongs to	Key fields	Watch out for
Contact	Location	email, phone, tags, source, customFields	Dedupes on email/phone — use upsert
Opportunity	Pipeline in Location	pipelineId, stageId, contactId, value, status	Fetch and cache pipeline/stage IDs first
Calendar	Location	calendarId, free slots, appointment times	Query free slots before creating events
Conversation	Contact in Location	messages, channel (SMS/email), direction	Real messages — add opt-out + quiet hours

Contact upsert template

```
POST /contacts/upsert
Authorization: Bearer {location_token}
Version: 2021-07-28
Content-Type: application/json

{
  "locationId": "LOC_123",
  "email": "jane@example.com",
  "phone": "+15551234567",
  "firstName": "Jane",
  "tags": ["inbound-lead"],
  "customFields": [{ "id": "CF_loan_amount", "value": "350000" }]
}
```

4. Rate-limit survival playbook

Treat these as ranges to design against, not guarantees. Read the response headers for your real remaining allowance on every call.

- **Burst:** roughly 100 requests per 10-second window per resource, per token.
- **Daily:** a ceiling in the low tens of thousands per location.

Backoff pseudocode

```
attempt = 0
while attempt < MAX_RETRIES:
    res = send(request)
    if res.status != 429:
        return res
    delay = min(BASE * 2**attempt, CAP) + random_jitter()
    sleep(delay)
    attempt += 1
raise RateLimitError
```

Budget worksheet

1. List every scheduled job and its record count.
2. Estimate requests per record (upsert = 1, opportunity move = 2, etc.).
3. Divide the daily total by your location ceiling. If you are over 60%, pace the job or move to webhooks.
4. Cache reference IDs (pipelines, stages, calendars, custom fields) to remove repeat lookups from the budget.

5. Production webhook checklist

- Verify the payload signature before trusting a single byte.
- Return 200 within ~2 seconds; push the payload to a queue and process asynchronously.
- Key processing on event/resource ID so duplicate deliveries are idempotent.
- Log the raw body for later debugging.
- Handle out-of-order retries so a late delivery does not overwrite newer data.
- Prefer webhooks over polling everywhere you can — it is free rate-limit budget.

6. Integration recipes

Recipe A — n8n lead-to-CRM

1. Webhook/trigger node receives the form or ad lead.
2. Enrich (clearbit-style lookup or your own source).
3. GoHighLevel contact upsert with tags for routing.
4. Create opportunity in the correct pipeline stage.
5. Notify the sales rep via Slack/SMS.

Recipe B — Custom booking agent

1. Agent asks for the desired day/time.
2. Call calendar free-slots endpoint.
3. Present real availability; create appointment on the chosen slot.
4. Write a confirmation back to the conversation thread.

7. GoHighLevel vs HubSpot decision table

Factor	GoHighLevel	HubSpot
Pricing model	Flat monthly, unlimited sub-accounts & contacts	Scales by contact tier and seat
White-label	Yes — resell as your own SaaS	No
Reporting depth	Adequate	Best-in-class
API polish & docs	Good, improving	Excellent
All-in-one breadth	SMS, email, funnels, booking, pipelines	Marketing/Sales/Service hubs, add-on cost
Best for	Agencies, local service, solo mortgage brokers	Mid-market, RevOps teams, regulated lenders

Vertical quick-picks

- **Agency reselling CRM:** GoHighLevel, for white-label + unlimited sub-accounts.
- **Mortgage / lending (independent):** GoHighLevel, for SMS-heavy follow-up at low cost.
- **Multi-branch regulated lender:** HubSpot, for governance and audit controls.
- **Clinic / med spa:** GoHighLevel for booking + reminders in one login.

8. Custom fields: the discipline that saves you

The single most common cause of a messy GoHighLevel integration is uncontrolled custom fields. Every field you create is a field you must map, sync, and maintain forever. Before you add one, ask whether it drives a decision, a message, or a report. If it does not, it is noise. Here is the governance I apply on every client build.

- **Name by convention.** Prefix fields by domain (loan_, appt_, lead_) so a new engineer can scan them in seconds.
- **Store the field IDs in config.** Custom fields are referenced by ID in the API, not by name. Fetch them once, cache them, and reference the config — never hardcode a raw ID in business logic.
- **Type deliberately.** A phone field, a date field, and a dropdown behave differently in workflows. Pick the type that matches how you will filter and automate on it.
- **Audit quarterly.** Fields accumulate. Once a quarter, list every custom field and delete the ones no workflow or report touches.

Field-mapping template

Source field	GHL custom field ID	Type	Drives
loan_amount	CF_loan_amount	number	Pipeline routing + reporting
preferred_time	CF_pref_time	text	Booking agent prompt
lead_source	CF_lead_source	dropdown	Attribution report

9. Migration checklist: moving into GoHighLevel

Most GHL projects start with a migration from a spreadsheet, an old CRM, or a HubSpot export. Rushing the import is how you end up with duplicate contacts and broken automations firing on historical data. Follow this order.

1. **Freeze the source.** Export a clean snapshot and stop writing to the old system for the cutover window.
2. **Map fields first.** Build the field-mapping table above before importing a single row.
3. **Pause automations.** Turn off workflows during import so a bulk load does not trigger thousands of SMS to old leads.
4. **Dedupe on the way in.** Use upsert keyed on email/phone so re-runs are safe.
5. **Import in batches.** Pace the load to stay under the daily rate ceiling; verify a sample batch before running the full set.
6. **Re-enable automations.** Turn workflows back on only after verifying data integrity on a sample.
7. **Reconcile counts.** Compare source and destination record counts and spot-check ten records end to end.

10. Troubleshooting quick reference

Symptom	Likely cause	Fix
401 Unauthorized	Expired access token	Refresh proactively before expiry; check token scope (company vs location)
Inconsistent endpoint behavior	Missing Version header	Set the dated Version header as a default on your HTTP client
429 Too Many Requests	Rate-limit exceeded	Exponential backoff with jitter; read remaining-count headers
Duplicate contacts	Using create instead of upsert	Switch to upsert; dedupe on email/phone within the location
Webhook fires twice	At-least-once delivery	Make processing idempotent on resource/event ID
Cannot read location data	Using a company-scoped token	Exchange for a location token via the locations endpoint

11. Pre-launch readiness checklist

- OAuth tokens stored in a secret manager, refresh rotation scheduled and tested.
- Version header set as a client default on every request.
- Rate-limit backoff with jitter proven under load, not just in a single-call test.
- Webhook signature verification live; receiver returns 200 in under two seconds.
- Processing idempotent under duplicate and out-of-order delivery.
- Pipeline, stage, calendar, and custom-field IDs cached in config.
- Structured logging that answers "what did the system do for this contact" in one query.
- Opt-out and quiet-hours logic in place before any outbound SMS goes live.
- A rollback path: how you pause automations and stop writes if something goes wrong.

Work with me

I build and maintain GoHighLevel integrations, AI booking and qualification agents on top of them, and the prospecting pipelines that feed them — I fill CRMs with qualified leads using ProLeads before they ever hit GHL. If you want a proven integration or an honest GHL-vs-HubSpot recommendation for your business, work with Syed Husnain Haider Bukhari and we will map your CRM setup end to end.