Holen Ventures

AGENTS & WORKFLOWS · BUILD GUIDE

Build Your Own Autonomous Marketing Agent

A marketing agent that checks traffic, monitors whether ChatGPT, Claude, and Perplexity recommend your products, tracks competitors, and tells you what to work on each week. Runs in n8n, on free tiers, in about an hour.

Henrik Holen

Updated June 2026 · ~1 hour · Intermediate

holenventures.com

Contents

WHAT YOU'LL BUILD, STEP BY STEP

WHY N8N, NOT A CLOUD AGENT

n8n is a workflow automation tool: you wire triggers and actions, and here an AI agent node, on a visual canvas, self-hosted or on their cloud. New to it? Their docs get you to a first workflow quickly. This is the second version of the build. An earlier one used Anthropic's Managed Agents API, a persistent Claude session in the cloud. It worked, but the plumbing was painful: Slack kept breaking, the agent burned credits polling for replies, and the whole thing felt fragile. The n8n version has native Slack triggers, no polling cost, and duplicating it for a second product takes five minutes.

What it does (and doesn't do)

The agent runs a three-step cycle, either on a weekly schedule or when you @mention it in Slack.

1. **Sense.** Pull traffic from your analytics, search performance from Google Search Console, and referral sources. Flag anything that moved more than 20% week on week. Check Google Trends to see whether your target queries are growing or shrinking as a category.
2. **Maintain.** Search the web for product mentions, and extract the "People Also Ask" questions for your target queries (these are the questions AI systems actually answer, so they make good article headings). Then hit the OpenAI, Claude, and Perplexity APIs directly to see whether they recommend your product when someone asks about your category. Call it your AI visibility, and it increasingly decides whether buyers find you at all.
3. **Ideate.** Propose up to three things to do this week, each with a data-grounded reason and an effort estimate.

It does not write, publish, or send anything. It observes and suggests, you decide what to act on. An earlier version had a content-creation step, dropped because the editing overhead ate the time savings.

Before you start

Services you'll need

Most of these have free tiers that comfortably cover a weekly run.

SERVICE	WHAT IT DOES	FREE TIER
n8n	Runs the workflow	Yes (self-hosted) or cloud free tier
OpenRouter	The agent's model, plus the Claude visibility check	Pay-as-you-go, cents per run
Web analytics (see below)	Traffic, pages, sources, conversions	Depends on provider
Google Search Console	Search impressions, clicks, keyword data	Free
Tavily	Web search and People Also Ask extraction	1,000 searches/month
Supadata	Web scraping	Free tier available
OpenAI	ChatGPT citation check	Pay-as-you-go
Perplexity	Perplexity citation check	Pay-as-you-go
Google Trends	Category trend direction	Free via pytrends or SerpApi

You'll also need a Slack workspace for the agent to post to and take questions from.

On the analytics tool: pick whatever you already use, as long as the agent can query it. Google Analytics 4 has an official MCP server. The examples here use Lodd (lodd.dev), an MCP-first analytics tool I built. Plausible has an API you can wire up over HTTP, but no MCP server yet.

> TIP
>
> Every tool here is swappable. Firecrawl or Apify instead of Supadata for scraping, Exa or SerpAPI instead of Tavily for search. What matters is the pattern: one tool for analytics, one for web search, one for scraping, and direct API calls to the AI platforms you want to check. Use whatever you're comfortable with. The OpenAI, Claude, and Perplexity calls are fractions of a cent per run; GSC and Trends are free.

Choose a model that can tool-call

Settle this before anything else in the build. The wrong model here doesn't fail loudly, it fabricates.

The agent is a multi-tool loop: it decides which of eight tools to call, calls them, reads the results, and calls more. That ability is **tool calling**, and not every model does it well. It tracks how hard the lab tuned the model for agentic use, not how clever the model sounds or how big it is.

Do this:

- Pick a model that scores well on public function-calling benchmarks. A current Qwen flagship (`qwen/qwen3.7-max`) is a solid default, near the top of the benchmarks and cheap on output tokens. Claude Haiku and the GPT-mini tier also tool-call reliably. Rock-bottom free and quantised models mostly don't.
- Verify before trusting any marketing page. OpenRouter exposes a `supported_parameters` list per model, you want `tools` in it.
- At one weekly run plus a few ad-hoc questions, the cost gap between capable models is cents. Optimise for tool-calling reliability, not token price.

> ● THE SILENT FAILURE
>
> A free open-source model tried first (gpt-oss) produced worthless results in a dangerous way: they looked plausible. It couldn't tool-call, so instead of pulling real data it wrote a confident report from thin air. "Traffic is up", "visibility unclear", no numbers. Without knowing the real traffic figures, that report is easy to believe. A model that can't tool-call doesn't error, it hallucinates a nice-looking report. That's worse than a crash, because a crash tells you something's wrong. The prompt in Step 5 is built to force this failure into the open.

Set it up, step by step

Seven steps from an empty workflow to a weekly report landing in Slack. About an hour end to end, most of it spent binding credentials one at a time.

Step 1: Confirm your analytics has an MCP server (or an API)

The agent needs to reach your analytics programmatically. The cleanest path is an MCP server it can point at, which auto-exposes the tools. If your analytics doesn't have one, a plain REST API works too, you just wire it up by hand. Either way it has to be able to ask "how many visitors this week, which pages, and where from?" and get an answer back.

- **GA4:** its MCP server handles this.
- **Lodd:** MCP endpoint `https://api.lodd.dev/mcp` with Bearer token auth.
- **Anything else:** check for an MCP server first (easiest), otherwise a plain REST API you wire up with an HTTP Request node (more manual). Either works.

Step 2: Collect your API keys

Gather these before opening n8n:

1. **Analytics token.** For Lodd: Dashboard → Settings → API keys.
2. **Google Search Console.** Verify your site at search.google.com/search-console if you haven't.
3. **Tavily API key.** From your tavily.com dashboard.
4. **Supadata API key.** From supadata.ai.
5. **OpenAI API key.** platform.openai.com → API keys → Create new.
6. **Perplexity API key.** docs.perplexity.ai → Settings → API keys.
7. **OpenRouter API key.** openrouter.ai → Keys. This is the agent's brain, and it also powers the Claude visibility check. It lets you swap models by changing one string.
8. **Slack bot token.** Needs a few more steps, see Step 3.

> NOTE ON GOOGLE TRENDS
>
> There's a free Python library (pytrends), but on n8n Cloud the simplest route is SerpApi's Google Trends API, which needs its own key. See the Google Trends part of Step 5.

Step 3: Set up the Slack bot

Go to api.slack.com/apps and create a new app. Name it something like "Marketing Agent" and select your workspace.

1. Under **OAuth & Permissions**, add these bot token scopes: `app_mentions:read`, `channels:history`, `channels:read`, `chat:write`.
2. Under **Event Subscriptions**, enable events and subscribe to `app_mention`. The request URL is your n8n webhook URL, which you'll get in Step 5, so you may need to come back here.
3. Install the app, copy the **Bot User OAuth Token**, and invite the bot to the channel where you want reports.

Step 4: Import the workflow into n8n

[Add your workflow link here.] Once you've finished building it, export the workflow and host the JSON as a public GitHub Gist (easiest) or in a repo, then drop the link here.

In n8n, go to **Workflows → Import from File** and select the JSON. You'll see the full workflow with two trigger paths feeding into the AI agent.

Step 5: Configure the workflow

The workflow has a model node plus several tools that each need credentials. Bind them one at a time and test after each.

> ● EXPECT THIS TO BE THE FIDDLY PART
>
> Every "it returned nothing" or "it errored" that comes up in setup is an unbound or wrong credential, never the logic.

> ● NOT EVERYTHING HERE IS MCP, DESPITE THE NAME
>
> Three tools (Lodd, Tavily, Supadata) are real MCP servers: point n8n's MCP Client node at an `/mcp` endpoint and it auto-discovers the toolbox, handling its own auth. The other five (OpenAI, Claude, Perplexity, GSC, Google Trends) are plain HTTP API calls: you use n8n's HTTP Request tool node and hand-write the one request you want, binding the credential yourself. That's why the API-based tools below need more fiddling than the MCP ones.

The model node

Connect an OpenRouter Chat Model node (or your provider's node) to the AI Agent. Paste your OpenRouter key and set the model to one that tool-calls well (e.g. `qwen/qwen3.7-max`). See "Choose a model" above for why this is the highest-stakes choice in the build.

MCP tools (analytics, Tavily, Supadata)

Each connects via n8n's MCP Client node. Add the endpoint URL, set authentication, and n8n auto-discovers the tools.

- **GA4:** follow Google's MCP setup guide. OAuth auth.
- **Lodd:** endpoint `https://api.lodd.dev/mcp`, Bearer token auth. 42 analytics tools auto-discovered.
- **Tavily:** endpoint `https://mcp.tavily.com/mcp/?tavilyApiKey=YOUR_KEY`. Gives web search and People Also Ask extraction.
- **Supadata:** endpoint `https://api.supadata.ai/mcp`, Header auth (`X-API-Key`). Handles scraping and YouTube transcripts.

Google Search Console

Two routes, depending on how you host n8n.

Self-hosted: use mcp-gsc, an open-source GSC MCP server exposing search analytics, URL inspection, indexing checks and sitemap status. It's a command-line (stdio) server, so: create a Google Cloud service account with Search Console API access, download its JSON key, add the service account's email as a user on your GSC property, clone mcp-gsc and point it at the key, then add an MCP Client node using stdio transport. Cloud n8n can't reach a stdio server, so this is self-hosted only.

n8n Cloud: skip the MCP server and call the Search Console REST API from an HTTP Request tool. Configure it like this:

- **Method/URL:** `POST https://searchconsole.googleapis.com/webmasters/v3/sites/sc-domain%3Ayourdomain.com/searchAnalytics/query`
- **Property identifier:** `sc-domain:yourdomain.com` is a *domain* property (the `%3A` is the URL-encoded colon). For a URL-prefix property, use the encoded `https://yourdomain.com/` form.
- **Auth:** generic credential type → Google OAuth2, scope `https://www.googleapis.com/auth/webmasters.readonly`.
- **Body (JSON),** using n8n date expressions so the window is always current:

```
{
  "startDate": "{{ $today.minus(28, 'days').toFormat('yyyy-MM-dd') }}",
  "endDate": "{{ $today.toFormat('yyyy-MM-dd') }}",
  "dimensions": ["query"],
  "rowLimit": 25
}
```

COMMON GOTCHA

In Google Cloud you must create a **Web application** OAuth client, not a Desktop one. Desktop clients only allow a `localhost` redirect, and n8n Cloud needs to redirect to its own hosted callback. When n8n creates the OAuth2 credential it shows a redirect URL, copy that into your Web client's "Authorised redirect URIs", then run the consent flow. A Google CLI tool you authorised locally is a Desktop client and can't be reused here, make a second Web client in the same project.

Once connected, the agent can pull impressions, clicks, CTR, and average position by query and page. High impressions with low clicks usually means Google shows your page but the title or description isn't working.

Google Trends

No official API, and this is the one tool to treat as optional.

- **Self-hosted:** the free pytrends library, run via an Execute Command node calling a small Python script, or wrap that script in a tiny HTTP endpoint.
- **n8n Cloud:** SerpApi's Google Trends API via an HTTP Request tool. Needs a key (httpQueryAuth credential, key goes in as `api_key`). The free tier covers a weekly run.

Either way the agent asks the same thing per target query: is interest UP, DOWN, or FLAT over the last 3 months. That separates "my site is declining" from "the whole category is declining", which are different problems with different fixes. But it's the flakiest tool, so if you want to ship faster, leave it out and add it later.

GENERAL N8N LESSON

If any single tool node errors on a missing credential, it aborts the entire agent run. An unconfigured Trends node takes the whole weekly report down with it. Either configure a tool properly or disconnect the node, never leave it half-wired.

AI visibility checks (OpenAI, Claude, Perplexity)

Three more HTTP Request tool nodes, calling the chat completions APIs directly to see whether the assistants mention your product. Claude reuses your existing OpenRouter key, so it needs no extra credential. Claude (via OpenRouter), pick a current Claude slug from OpenRouter's model list:

```
POST https://openrouter.ai/api/v1/chat/completions
Authorization: Bearer YOUR_OPENROUTER_KEY

{
  "model": "anthropic/claude-sonnet-4.6",
  "messages": [{
    "role": "user",
    "content": "What [your category] tools exist? List specific products."
  }]
}
```

Point the OpenAI node at `api.openai.com/v1/chat/completions` and the Perplexity node at `api.perplexity.ai/chat/completions` with the same question. Replace `[your category]` with what your product does: "web analytics tools", "equipment tracking software", "teleprompter apps for Mac", whatever fits.

Memory, routing, and Slack

Connect a Simple Memory node (last 10 messages) so the @mention path keeps context for follow-ups; the weekly run starts fresh and doesn't need it. Route output through an If node that checks which trigger started the run: scheduled runs post to a dedicated weekly-report channel, ad-hoc questions reply where you asked. Paste your Bot User OAuth Token into the Slack Trigger node and both Send Message nodes, and set the channel IDs.

The agent prompt

The Edit Fields node after the schedule trigger holds the weekly prompt. Customise it for your product:

```
You are a marketing agent for [YOUR PRODUCT]. Run the following cycle.

TOOL USE IS MANDATORY. You have these tools: [list them]. You MUST
actually call the relevant tool for every section below. Do NOT answer
from memory or prior knowledge. If you have not called a tool, you do
not know the answer.

Every factual claim MUST be backed by raw evidence from a tool call:
an exact number, an exact date range, or a source URL. Bare statements
like "traffic is up" or "visibility unclear" are forbidden.

If a tool was not called, returned nothing, or errored, say so with the
literal token TOOL NOT CALLED or NO DATA for that line.

SENSE:
- Analytics for the last 7 days vs the previous 7. Exact visitor/
  pageview counts, which pages grew or declined (with numbers), new
  referral sources, and event counts.
- Google Search Console for the last 28 days. Top queries by impressions
  with exact impressions, CTR, and position. Flag any page with high
  impressions but low clicks (title/description problem).
- Google Trends for [YOUR TARGET QUERIES]. UP, DOWN, or FLAT over the
  last 3 months, citing the tool result.

MAINTAIN:
- Web search for mentions of [YOUR PRODUCT] in the last week, with URLs.
- Web search for "People Also Ask" questions for the target queries.
  List them and flag which are not yet covered by existing content.
- Ask OpenAI, Perplexity, and Claude (via OpenRouter): "What [YOUR
  CATEGORY] tools exist? List products." State whether [YOUR PRODUCT]
  appears, and quote the relevant part.
- For any competitor claim, include the source URL or mark UNVERIFIED.

IDEATE: Based ONLY on the tool results above, propose up to 3 actions.
Each: what to do, why (citing a specific number or finding), effort
(quick/medium/significant). PAA questions not covered by existing
content are strong article candidates.

Rules: lead with findings, not process. Every claim carries its
evidence. If a section has nothing to report, say "all clear", but only
after the tool was actually called. Under 500 words.
```

● DON'T WATER THIS PROMPT DOWN

The "mandatory tool use" and "every claim carries evidence" rules are what stop a weaker model from skipping the tools and inventing a tidy report. The `TOOL NOT CALLED` / `NO DATA` tokens turn a silent failure into a visible one, you'll see exactly which tool didn't fire instead of getting smooth fiction.

THE AGENT HAS NO CLOCK, AND THAT WILL BITE YOU

An early version kept comparing "this week" against a December week that didn't exist: the model guessed today's date, got it wrong, and anchored every comparison to nonsense. The fix is two lines. Put the real date in the agent's **system message** (`Today's date is {{ $now.toFormat('yyyy-MM-dd') }}`), and tell it to use your analytics tool's relative periods (`7d`, `30d`) instead of doing date maths itself.

Step 6: Set the schedule and test

Set the Schedule Trigger to run weekly. Saturday mornings work well, so the report is waiting at the start of the week. Then run a manual test and confirm the agent:

1. Pulls analytics (real traffic numbers)
2. Pulls search data from GSC (impressions, clicks, top queries)
3. Reports trend direction from Google Trends
4. Extracts People Also Ask questions
5. Checks ChatGPT, Claude, and Perplexity for citation visibility
6. Posts a report to your Slack channel

If something fails, it's almost always a credential issue. Check the API keys, the MCP endpoints, and the Slack bot scopes.

TWO TESTING GOTCHAS

A partial run won't post to Slack. n8n's "Test Workflow" often runs only part of the graph and stops at the branching point, never reaching the Slack node. To verify the full path, run it end to end.

Empty tool results look like broken tools but usually aren't. A partial run that skips the tool-gathering branch makes the agent report `NO DATA` everywhere. Run the complete cycle before concluding anything's wrong.

Step 7: Ask it questions

Once the weekly cycle works, @mention the bot in Slack:

> *@marketing-agent what drove last Tuesday's traffic spike?*
>
> *@marketing-agent are any competitors running a sale right now?*

The agent uses the same tools for ad-hoc questions as for the weekly cycle, and the Simple Memory node keeps the last 10 exchanges, so you can have a back-and-forth.

THE @MENTION PATH HAS A SPECIFIC FAILURE MODE

The Slack Trigger node has a webhook URL, and your Slack app's Event Subscription must point at that exact URL. If you rebuild or re-import the workflow, n8n regenerates the webhook ID, the old URL goes dead, and @mentions silently stop arriving (the scheduled report still works, because that path has no webhook). If mentions stop, copy the Trigger node's current Production webhook URL back into Slack's Event Subscriptions.

Customising for your product

The workflow is a template. The things you'll want to change:

- **Target queries.** The citation checks ask ChatGPT, Claude, and Perplexity about your category. Pick the 3-5 queries your ideal customer would type into an AI assistant ("best [category] for small teams", "[competitor] alternative", "how to [solve the problem]") and put the most important one in the HTTP tool prompts.
- **Competitor list.** Name specific competitors in the agent prompt so it knows what to watch for. Always require source URLs, mark anything it can't cite as `UNVERIFIED`. An early version attributed a competitor's negative review to the wrong product because two names were similar.
- **Anomaly threshold.** The prompt flags changes over 20%. Under 100 visitors/week, raise it to 30-40% so you're not alerted on normal variance.
- **Report format.** "Under 500 words" keeps reports scannable. Keep it tight, you can always ask follow-ups in Slack.

IDEATION IS ONLY AS GOOD AS YOUR DATA

On a new or low-traffic site the suggestions will be generic. After a few months of data they get specific and useful. The citation checking is the most valuable part, knowing whether ChatGPT, Claude, and Perplexity recommend your product, and exactly what they say, is hard to get any other way.

Going further

- **Content creation.** Dropped from this build, but if your product voice is straightforward and you're happy editing AI drafts, add a create step that writes posts from the ideation output. PAA questions make good headings.
- **SEO health audits.** Crawl your sitemap weekly, check every page for title tags, meta descriptions, JSON-LD, and heading hierarchy. Auto-fix the mechanical stuff, flag the rest. This works well as a separate workflow.
- **Multi-product.** Duplicate the workflow per product: different Slack channel, different analytics site, different target queries. The template is the same.

Build something with this?

This is one of a series of build guides. If you want help setting up something similar, or just want the weekly letter on building in the AI era, here's where to go.

MORE BUILDS	holenventures.com/agents-and-workflows
THE LETTER	holenventures.substack.com
WORK WITH ME	holenventures.com/work-with-me
EMAIL	henrik@holenventures.com