FREE DIY BLUEPRINT

DIY AI Content Factory Architecture

The exact system — 11 knowledge docs, 6 workflows, the tool stack, and the self-maintenance loop. Build it yourself, step by step.

Built on Claude Code

The hands-on companion to the course · build your factory from zero

---

What you're building

A repeatable system where **Claude Code** reads a set of instruction files (your "brain") and runs a series of workflows that turn a keyword into a published, on-brand, SEO- and AI-search-optimized article — with you approving the key decisions and nothing going live without your click.

```
your-project/
├── CLAUDE.md          # the front desk — rules Claude reads every session
├── guidelines/        # THE BRAIN — 11 documents (Module 1)
├── workflows/         # THE ASSEMBLY LINE — 6 workflow files (Module 3)
├── articles/          # output — one folder per article, every step saved
├── reports/           # weekly performance reports (Module 4)
└── credentials/       # 🔒 API keys — NEVER shared, NEVER synced to Git
```

Time & cost expectations

	From scratch	With the course templates
Setup time	~1 week	2–3 days
Weekly running time (once live)	—	~5 hours
Tooling cost to start	$0 (free tools work)	$0–$129/mo (optional Ahrefs)

PART 0 — Before you start (½ day)

0.1 Accounts & installs you need

🟢 Get these ready first so you're not context-switching mid-build:

- **Claude Code** — the app you'll type into (desktop app, or claude.ai/code). A paid Claude plan is required for real usage.
- **Your website's CMS login** with admin/editor rights (WordPress, Webflow, Ghost, Sanity, etc.).
- **Google Search Console** — free; your site should already be verified. If not, verify it now (it takes a day for data to populate).
- **A keyword tool** — Ahrefs if you have it; otherwise the free stack in §2.6. Optional to start.
- **(Optional) Notion or Google Sheets** — your content calendar/dashboard.

⚠️ **Potential difficulty — "Do I need to know how to code?"** No. You'll be writing instructions in plain English and pasting a few templates. The most "technical" things are: opening a folder, copying files, and pasting an API key into a text file. If you can use Google Docs, you can do this.

⚠️ **Potential difficulty — Search Console has no data yet.** If your site was just verified, GSC needs a few days to collect data. Start Part 1 (the brain) while it fills up — you don't need GSC data until Module 4.

0.2 Create the project folder

🟢 Make one folder on your computer named e.g. `content-factory`. Open it in Claude Code (File → Open Folder, or point Claude Code at it). Everything lives here.

💡 Put this folder in a synced location (iCloud/Dropbox/OneDrive) **except** the `credentials/` folder (see §2.7). Better yet, use Git (Part 5) so your whole team stays in sync.

PART 1 — Build the Brain (Module 1) · the 11 documents

This is the highest-leverage part of the entire build. Everything downstream is only as good as these files.

1.1 Create the guidelines folder

🟢 Inside your project, create a folder called `guidelines/`. You'll drop 11 markdown (`.md`) files in here. 📄 Start from `/templates/guidelines/` — every file is a fill-in-the-blank skeleton.

1.2 Write the 3 foundation documents first

🟢 Do these three before anything else — they carry the most weight:

1. `tone-of-voice.md` — Extract, don't invent.
   - Gather your 3–4 best-performing pieces of content.
   - Paste them into Claude: "Describe the voice, rhythm, and personality of this writing as a set of rules a new writer could follow. Include sentence length, formality, humor, and 5 example phrases that sound like us and 5 that don't."
   - Edit the result into `tone-of-voice.md`. Decide explicitly: contractions? emoji? "we" or "I"? witty or authoritative?

⚠️ **Difficulty — "My existing content isn't great / doesn't exist yet."** Then borrow a target voice: pick 2–3 writers/brands whose tone you admire, paste their public content, and adapt. Or describe your voice in 10 adjectives and let Claude draft rules you refine. You can always sharpen it after the first few articles.

2. `positioning.md` — what you sell, who for, 3 differentiators, and **what you must NEVER claim** (honest limitations, e.g. "no mobile app," "no free plan"). Include approved product one-liners and any sourced stats you're allowed to cite.

⚠️ **Difficulty — the AI invents features you don't have.** This is always caused by a thin positioning doc. Add an explicit "❌ Never claim" list. Example: "Never say we integrate with X. Never cite a customer count — we don't publish one." The factory obeys what's written; write the limits down.

3. `icp.md` — your customer segments. For each: role, top 3 pains, and **the exact words they use** (real phrases from calls/reviews, not marketing-speak).

⚠️ **Difficulty — "I don't know their exact words."** Mine them: read 20 reviews of you or competitors (G2, Capterra, Trustpilot, App Store), skim 2–3 relevant subreddits, and pull your last 10 sales-call notes. Copy real sentences verbatim. This becomes some of your most valuable content fuel.

1.3 Write the remaining 8 documents

🟢 Fill these from the 📄 templates — a paragraph or a short list each is enough to start:

File	The one thing it must contain
`seo.md`	Your title/meta patterns + where the primary keyword goes (H1, first 100 words, 2 H2s)
`aeo.md`	"Lead every section with a 1–2 sentence direct answer; use numbers not adjectives; add structured data"
`funnel.md`	For TOFU / MOFU / BOFU: how hard to sell, CTA depth, internal-link count, product-mention cap
`linking.md`	Internal link count per article + your priority/"money" pages + anchor rules
`humanization.md`	Your **banned words/openers/structures** kill-list + rhythm rules
`formatting.md`	Heading case, paragraph length, bullets vs tables, emoji policy
`faq.md`	How many FAQs, answer length, "source real questions — never invent"
`visuals.md`	Your colors/fonts + when to use screenshot vs diagram vs chart vs AI image

💡 **Steal a starting kill-list for `humanization.md`:** ban delve, leverage, unlock, elevate, seamless, robust, in today's fast-paced world, it's important to note, whether you're X or Y, it's not just X — it's Y, in conclusion. Add your own as you spot them.

1.4 Wire the brain to CLAUDE.md

🟢 Create `CLAUDE.md` in the project root (📄 template in `/templates/CLAUDE.md`). This is the index Claude reads first, every session:

```
# Content Factory — Operating Rules

## Read before writing anything
Always read the relevant files in `guidelines/` before producing content:
- Voice & humanization: guidelines/tone-of-voice.md, guidelines/humanization.md
- What we sell: guidelines/positioning.md, guidelines/icp.md
- How we rank: guidelines/seo.md, guidelines/aeo.md, guidelines/funnel.md
- How we write & look: guidelines/linking.md, guidelines/formatting.md, guidelines/faq.

## Hard rules (never break)
1. Never fabricate statistics, quotes, or sources. Flag unverifiable claims — don't inv
2. Every workflow saves its output to a file before the next step runs.
3. Publish to the CMS as a DRAFT only. Going live is a separate step I approve explicit
4. Every article has a named author byline and sounds like a real person (humanization.
5. Guidelines override everything. If unsure, follow the guideline and tell me.
```

⚠️ **Difficulty — Claude seems to "ignore" the guidelines.** Three usual causes: (1) the files are too long/vague — tighten them into rules, not essays; (2) `CLAUDE.md` doesn't point to them — check the paths; (3) the guideline contradicts itself — pick one rule per topic, one home file. When in doubt, ask Claude: "Which guideline rule did you apply here?" — it'll reveal what it did and didn't read.

✅ **Part 1 done when:** all 11 files exist (even if rough) and `CLAUDE.md` links them.

PART 2 — Connect the Tools (Module 2)

Tools connect to Claude Code through **MCP connectors** — think of them as USB ports. Plug in once, use forever.

2.1 How to add a connector (general shape)

🟢 In Claude Code, open the connectors/MCP settings and add each service. Most follow: choose the service → authorize with your account (OAuth) or paste an API key → confirm it appears as "connected." Test each with a plain request like "list my connected tools" or "pull my top 10 Search Console queries."

⚠️ **Difficulty — a connector won't authorize / "disconnected."** Re-authorize from scratch (remove and re-add). Check the key hasn't expired and has the right scopes/permissions. Corporate Google/Microsoft accounts sometimes block third-party access — try a personal or properly-permissioned account, or ask your admin. Don't block the whole build on one connector; connect the others and come back.

2.2 Connect Google Search Console (free — do this first)

🟢 Authorize the GSC connector with the Google account that owns your verified property. Test: "What are my top 20 GSC queries by impressions in the last 28 days?"

⚠️ **Difficulty — "No data available."** The property must be verified and have ≥ a few days of history. Make sure you picked the exact property (https vs http, www vs non-www, domain vs URL-prefix). If you have multiple properties, name the right one.

2.3 Connect your CMS (biggest time-saver)

🟢 Connect via the available method for your platform:

- **WordPress** — REST API (application password) or a connector/plugin.
- **Webflow** — CMS API token + collection ID.
- **Ghost** — Admin API key.
- **Sanity / headless** — a write token + a small publish script.

Test with a draft: "Create a draft post titled 'zzz-test' with one paragraph — do not publish." Confirm it appears in your CMS as a draft, then delete it.

⚠️ **Difficulty — formatting breaks on publish (bold, headings, tables vanish).** Every CMS has its own content format. Do a throwaway test post and check what survives. If tables/embeds get stripped, note it in `formatting.md` and have the publish workflow output the CMS-friendly format. Get this contract right once and never fight it again.

⚠️ **Difficulty — images won't upload / hot-linking.** Upload images to the CMS's own media library via the publish step; don't link to files on your computer or a temp URL. Test one image end to end before trusting the pipeline.

⚠️ **Difficulty — it published live instead of as a draft.** Stop and fix the workflow immediately. The publish workflow must set draft/unpublished status. Re-read §Hard rule 3. This is the one mistake with public consequences.

2.4 Set up the visuals pipeline

🟢 Pick your level:

- **Simplest:** auto-generate a branded cover + simple diagrams/charts from a small template (fastest, most consistent).
- **On-brand:** connect Figma for designed graphics.
- **Illustrative:** use AI image generation for hero/section images. Define the recipe in `visuals.md` and reference it from the visuals workflow.

⚠️ **Difficulty — visuals look generic/off-brand.** Put your hex colors, fonts, and 2–3 example images directly in `visuals.md`. "On-brand" is a spec, not a vibe — write the spec.

2.5 (Optional) Content calendar — Notion or Sheets

🟢 Create a simple board/sheet with columns: Idea · Keyword · Funnel stage · Status (idea/in progress/review/published) · URL · Published date. Connect it so Claude can update rows automatically.

2.6 No Ahrefs? Use the free stack.

🟢 You do **not** need paid tools to start:

- **Google Search Console** (free) — most accurate for your site.
- **Google Keyword Planner** (free with a Google Ads account) — volumes.
- **Ubersuggest / Keywords Everywhere** (cheap) — quick volume + ideas.
- **Claude + web search** (free) — competitor analysis, People-Also-Ask, angle-finding.

💡 Start free, prove it on your own site, upgrade to Ahrefs later when the traffic pays for it.

2.7 🔒 Lock your keys away (do NOT skip)

🟢 Create a `credentials/` folder. Put every API key/token in there. Then block it from Git/sync:

- Create a `.gitignore` file in your project root containing the line: `credentials/`
- Never paste a key into a chat, a public file, or a screenshot.

⚠️ **Difficulty — "I think I committed a key to Git / shared it."** Treat it as compromised: **rotate (regenerate) that key immediately** in the tool's dashboard, then re-add the new one to `credentials/`. Rotating is free and takes a minute; a leaked key can cost you a lot.

✅ **Part 2 done when:** `list my connected tools` shows GSC + your CMS (at minimum), a test draft posted and deleted cleanly, and `credentials/` is gitignored.

PART 3 — Build the Assembly Line (Module 3) · 6 workflows

A workflow is just a plain-English instruction file. Create a `workflows/` folder and add these six (📄 templates in `/templates/workflows/`). Each file describes one job and says save your output to a file.

3.1 The stage principle (why this works)

Each workflow does **one** job and **saves its output** before the next runs. This gives you quality (focused steps) and debuggability (open the files, see exactly where it went wrong, fix that one instruction, re-run from there).

3.2 Workflow 1 — Idea generation → `workflows/1-ideas.md`

Job: combine keyword demand + competitor gaps → a **prioritized** shortlist (not 200 keywords). Run it: "Run idea generation for the topic: employee onboarding." Output → `articles/_ideas/ideas-<date>.md`.

⚠️ **Difficulty — ideas are generic or off-target.** Feed it your `icp.md` and money pages, and constrain it: "Only topics my ICP would search near a buying decision, with beatable competition." Reject anything not tied to a real reader.

3.3 Workflow 2 — Research + outline → `workflows/2-research.md`

Job: read the live SERP for the chosen keyword — who ranks, what angle, what questions recur, **what they all miss** — then produce an outline with a fresh angle and the FAQ questions. Output → `articles/<slug>/research.md` + `outline.md`. **This is a checkpoint. You approve the outline before drafting.**

⚠️ **Difficulty — it "researches" without really reading competitors.** Require it to fetch and quote specifics from the top 3–5 results and name the gap in one sentence. If it can't name what everyone misses, the topic may be too crowded — re-scope.

3.4 Workflow 3 — Draft → `workflows/3-draft.md`

Job: write the full article from the approved outline, through the brain (voice, positioning, funnel, linking). Output → `articles/<slug>/draft.md`.

⚠️ **Difficulty — draft is too long / too salesy / wrong reader.** Length: set a word target in the outline sized to the competition (cap ~2,500). Salesiness: enforce `funnel.md` (TOFU teaches,

BOFU sells). Reader: confirm the ICP at the top of the outline.

3.5 Workflow 4 — Visuals → `workflows/4-visuals.md`

Job: generate the cover + any diagrams/charts/screenshots per `visuals.md`. Output → `articles/<slug>/assets/`.

3.6 Workflow 5 — Proofread + humanize → `workflows/5-qa.md`

Job: run the draft against `humanization.md` (kill robot tells) + a quality checklist (facts sourced? claims honest? each section answers its question? links real?). Fix what it can, flag the rest. Output → `articles/<slug>/final.md`.

⚠️ **Difficulty — output still "smells like AI."** Your kill-list is incomplete. Every time you catch a tell, add it to `humanization.md`. Also require sentence-length variation and ≥1 first-person, specific detail per section. The QA workflow is only as sharp as the guideline behind it.

3.7 Workflow 6 — Publish → `workflows/6-publish.md`

Job: format for your CMS, upload images, set title + meta + internal links, push as a **draft**, return the review link. Going live is your separate, explicit approval.

⚠️ **Difficulty — internal links point to made-up URLs.** Give the workflow your real sitemap or a link list (from GSC / your CMS) and instruct: "Only link to URLs that exist in this list." Never let it guess URLs.

3.8 Do a full run-through

🟢 Run one article end to end: idea → approve outline → draft → visuals → QA → publish-as-draft → read it in the CMS → hit publish → load the live URL. Time it. This is your "30 minutes total" proof.

⚠️ **Difficulty — a stage fails midway (tool down, page won't fetch).** Because every stage saved its file, you don't restart. Fix the issue and say "re-run from stage 3 for ". That's the whole point of the staged design.

✅ **Part 3 done when:** one real article is live on your site, produced through all six workflows with you approving the outline and the publish.

PART 4 — Automated Performance Analysis (Module 4)

4.1 Build the weekly report → `workflows/7-weekly-report.md`

🟢 Job: pull GSC data and score every article on four signals, then output a ranked "update these" list to `reports/report-<date>.md`.

The scoring logic (put this in the workflow):

- **Decay** — avg position dropped ≥3 vs previous period → +3
- **Striking distance** — currently ranks position 5–20 → +2
- **Staleness** — not updated in N months on a fast-moving topic → +2
- **Lost feature** — lost a featured snippet / AI citation → +2
- Sort by score, highest first.

⚠️ **Difficulty — everything looks "fine" or the list is empty.** Widen the window (28 → 90 days) so trends show, and make sure GSC has enough history. Early on, focus on striking distance — quick wins sitting on page two.

4.2 The update (refresh) workflow → `workflows/8-refresh.md`

🟢 Job: pick an article from the report → pull the live version → re-check the market → propose a **change plan** (surgical edits, not a rewrite) → you approve → apply only those edits, protecting images/links/layout → stage a draft.

⚠️ **Difficulty — the refresh rewrites the whole article and tanks it.** Refresh ≠ rewrite. Instruct: "Change only what the plan names. Keep all existing images, links, and structure byte-for-byte unless listed." A rewrite throws away ranking history and backlinks — protect them.

4.3 Bonus — monthly content audit

📄 Use the `/templates/monthly-audit-prompt.md`: which topics are we winning/losing, any two articles cannibalizing the same keyword, obvious content gaps, anything to merge or retire.

4.4 Put it on a schedule

🟢 Schedule the weekly report (e.g., Monday 8am) so it lands in your inbox or Notion. Your habit becomes: coffee → open the list → kick off 2–3 refreshes → done.

✅ **Part 4 done when:** a weekly report generates automatically and you've run one successful refresh from it.

PART 5 — Run it, keep it safe, scale it

5.1 Save & share with Git (no commands to memorize)

Ask Claude in plain English: "Save and push my changes" (commits + pushes), "Pull the latest changes" (get teammates' work), "Put my changes on a branch and open a pull request" (for review). Habits: pull before you start, save/push when you finish.

⚠️ **Difficulty — "merge conflict."** Don't panic. Say "help me resolve this conflict" and follow along. Never force-overwrite shared work.

5.2 Improve the machine, not the article

When output disappoints: open the stage files, find where it went wrong, **fix that guideline or workflow file**, and re-run. Never hand-patch an article without fixing the instruction — that's the difference between a tool and a factory that improves with every run.

5.3 The weekly rhythm (~5 hours)

1. Open the weekly report; kick off 2–3 refreshes. (30 min)
2. Run idea generation; pick next week's topics. (20 min)
3. Produce new articles: approve outlines, read finals, publish. (~30 min each)
4. Once a month: run the content audit.

Appendix A — Troubleshooting quick table

Symptom	Most likely cause	Fix
Output sounds generic	Thin `tone-of-voice.md` / `positioning.md`	Enrich with real examples + rules
AI invents features/stats	Missing "never claim" list; no source rule	Add limits to `positioning.md`; enforce hard rule 1
Still smells like AI	Incomplete kill-list	Grow `humanization.md` every time you spot a tell
Connector won't authorize	Expired key / wrong scopes / corp account	Re-add key; check permissions; try proper account
"No GSC data"	Wrong property / not enough history	Pick exact property; wait a few days; widen window
Formatting breaks on publish	CMS format mismatch	Test post; codify CMS-friendly format in the workflow
Internal links 404	Guessed URLs	Feed it the real sitemap; "only link listed URLs"
Published live by accident	Workflow not set to draft	Fix workflow to stage drafts; go-live is separate
Refresh tanked an article	Rewrote instead of edited	Enforce surgical-edit + preservation rules
Leaked an API key	Committed/shared credentials	Rotate the key now; keep `credentials/` gitignored

Appendix B — Glossary (plain English)

- **Claude Code** — the app that reads your files and runs the workflows.
- **MCP / connector** — a "cable" that lets Claude use another tool (Ahrefs, GSC, your CMS).

- **CLAUDE.md** — the index file Claude reads first every session.
- **Guidelines / the brain** — the 11 documents that make AI write and sell like you.
- **Workflow** — a plain-English instruction file for one job in the assembly line.
- **SEO** — optimizing to rank in Google's blue links.
- **AEO / GEO** — optimizing to be quoted/cited by AI search (ChatGPT, Perplexity, AI Overviews).
- **TOFU/MOFU/BOFU** — top/middle/bottom of funnel; how ready the reader is to buy.
- **Draft-only publishing** — the factory stages a draft; you click go-live.
- **Refresh** — surgical update of a live article that protects its ranking history.
- **Striking distance** — keywords ranking positions 5–20; a nudge from real traffic.

Companion to the "How to Build an AI Content Factory" course. Build the brain first. Keep a human on the decisions. Fix the instruction, not the article.