FREE BUILD-ALONG GUIDE

LinkedIn Lead Gen Build It Yourself

The exact prompts I used, plus the real numbers from testing it

Fabian Markl
FabiMarkl.com

Start here

This is the exact set of prompts I used to build a LinkedIn lead generation system with Claude Code, on camera, from an empty folder.

You paste them in order. Claude writes all the code. You do not need to know how to program, and you never have to write a line of it yourself.

> 🎯 **WHAT YOU END UP WITH**
> A dashboard on your desktop listing people who match your ideal customer, scored 0 to 100, with their email address where one exists, and three ready-to-send messages written for each person: a LinkedIn connection note, a follow-up DM, and an email.

> ⚠ **WATCH OUT**
> Every number in this guide is measured, not estimated. I ran this system for real while writing it. Where something broke, I have said so and shown you the fix.

What you need before you start

1. **Claude Code**
   The tool that writes the code for you. It comes with a paid Claude plan.
2. **An Apify account**
   Free to start. This is what reads LinkedIn. Sign up at apify.com, then go to Settings, then API and Integrations, and copy your token.
3. **A Firecrawl account**
   Free to start. This is the backup email finder. Sign up at firecrawl.dev, then API Keys, then Create API Key.
4. **Optional: an Anthropic API key**
   Only if you want the message openers written by AI instead of from a template. Roughly 5 to 8 dollars a month. Everything works without it.

> 💡 **TIP**
> If any step below gets technical or throws an error, just paste the error back to Claude Code and say what happened in plain words. It has the whole system in front of it and can fix it. You do not need to understand the error.

PART 1

Read this first

Two things that will save you money and a lot of frustration.

THE VOLUME TRUTH

The most useful thing in this guide

LinkedIn lets you send roughly 20 to 25 connection requests a day before it starts throttling you. That is the real ceiling, and no tool changes it.

So a system that finds 500 people a day is mostly buying you people you cannot contact. Scraping 10 to 25 new leads a day matches what you can actually send, and costs about 6 to 11 dollars a month.

> 💡 **THINK OF IT LIKE…**
> It is like buying 500 stamps when you only have time to write 20 letters. The extra stamps are not an advantage, they are just money sitting in a drawer.

CONNECTION REQUESTS ARE ALWAYS MANUAL

This is not a setting

The system writes your LinkedIn connection notes and DMs. You copy and paste them yourself.

This is deliberate. LinkedIn actively detects automated connecting and restricts accounts that do it. Any tool that promises to automate LinkedIn connection requests is putting your account at risk.

> **Email is different.** You own your mailbox and nobody bans you from it, so email can be fully automated if you want it. That is a separate opt-in step, not part of this core build.

PART 2

The seven prompts

Paste each one into Claude Code, in order, and wait for it to finish before the next.

PROMPT 1

Your business details and keys

This one interviews you and sets up the project. Claude asks one question at a time and waits for each answer.

TRY THIS PROMPT

"I want to set up a LinkedIn lead-gen system. Interview me first, one question at a time, waiting for each answer before asking the next:

1. What business/niche am I targeting leads for?
2. My name
3. My expertise/role
4. The specific result I get clients
5. The benefit of that result
6. Target job titles I want to reach (e.g. 'marketing director, founder, head of growth')
7. Target company type or industry (e.g. 'AI SaaS companies, 10-200 employees')
8. My target country
9. My Apify API token, if I say I don't have one, tell me to go to apify.com, sign up free, then Settings > API & Integrations, and copy it
10. My Firecrawl API key, if I don't have one, tell me to go to firecrawl.dev, sign up free, then API Keys > Create API Key
11. Optionally, an Anthropic API key for AI-written messages (explain it's optional, costs roughly $5-8/month at volume, and everything still works without it)

Once you have all my answers, create:

.env with APIFY_TOKEN, FIRECRAWL_API_KEY, and ANTHROPIC_API_KEY (blank if skipped), plus .gitignore containing .env

CLAUDE.md with this structure, filled in with my answers:

 # LinkedIn Lead Gen, Project Instructions

Business
- Sender name, expertise, specific result, benefit, target job titles, target company type, target country

Scoring formula
- Job title match: exact match to target list = 40 points, adjacent/related title = 25, none = 0

- Company type/industry match: 20 points if matches target industry, 10 if adjacent, 0 if not
- Has a linked website: +15
- Has a findable email via that website: +25
- Bands: 70+ strong fit, 40-69 decent, under 40 weak

Outreach copy rules
- Connection-request note: under 300 characters, no exceptions, this is LinkedIn's hard cap
- NEVER use an em dash or en dash anywhere in a message, including inside quoted text
- NEVER paste raw scraped post text into a message. Strip URLs, emoji, and repeated punctuation first. If fewer than 3 words remain, use a generic opener instead of a fragment
- Greeting: use their first name only, never 'Hi [Full Name]', LinkedIn norm is first-name-only outreach
- One sentence per line in the DM, blank line between each. Max 1 emoji in a connection note, 1-2 in the DM
- This system generates drafts ONLY, for both the connection note and the DM. It never sends or connects automatically. Nothing about the paid tier changes this, connection requests and DMs are always manual, on every tier, because LinkedIn detects and blocks automated connecting

Output
- leads_data.csv, all leads ever found
- LinkedIn Leads.html, the dashboard, regenerated in place each run
- li_scrape_log.csv, run history and every profile URL scraped, powers deduplication
- search_yield_log.csv, per search-term new vs repeat ratio

Then run: pip install requests python-dotenv"

PROMPT 2

Make Claude check the tool before it builds

Paste this on its own, before the build prompt. It forces Claude to go and look at the actual tools available and tell you which one it picked and why.

✍ **TRY THIS PROMPT**

"Before writing any code: search Apify's actor store for a LinkedIn people-search actor that can search by job title and company or industry. Compare 2-3 candidates, check their actual input schemas, and tell me which one you're using and why before you proceed."

⚠ **WATCH OUT**

Do not skip this and do not let it slide if Claude picks a tool without testing it. In my build, Claude chose a tool based on its description and started writing code, and that cost me about half an hour of debugging later. Ask it to run one real test call first.

PROMPT 3

The scraper

This is the biggest prompt and the heart of the system. Point 5 is the one that saves you the most money.

✍ TRY THIS PROMPT

"Build scrape_leads.py, a LinkedIn lead scraper. Read CLAUDE.md first for my business details. Requirements:

1. SEARCH TERMS: build a rotating list of search batches combining my target job titles with my target company type/industry, similar to how a hashtag rotation works but for job-title + industry pairs. Include enough combinations to avoid re-searching the same pair too often.

2. Each run uses the NEXT batch. Store the last batch index in scrape_state.json so runs rotate automatically.

3. Support a --test flag that uses 1 search term and 15 results and does NOT advance the rotation state.

4. Search profiles with the actor you just identified. Collect profile URL, name, headline, company, and their most recent post/activity if the actor provides it.

5. DEDUPLICATE BEFORE SPENDING MONEY: read every profile URL from the 'all_profiles_scraped' column of li_scrape_log.csv across all past runs, remove those from this run's list before any enrichment. This is the single most important cost control.

6. Log per-search-term yield to search_yield_log.csv: date, search term, profiles found, new profiles, new percent. Flag any term under 15 percent new as tapped out.

7. Enrich remaining profiles to get full headline, company, job title, and an externalUrl/website field if the profile has one.

8. Find contact info: if the profile has a linked website, use Firecrawl to try the homepage, /contact, and /about:

POST https://api.firecrawl.dev/v2/scrape
header: Authorization: Bearer FIRECRAWL_API_KEY
body: {"url": ..., "formats": ["markdown"]}
Regex the returned markdown for an email.

If no website is linked, leave email blank, this lead only gets a connection-request and DM draft, no email draft.

9. Score each lead with the formula in CLAUDE.md.

10. Append to leads_data.csv with columns: profile_url, full_name, headline, company, job_title, website, email, score, date_found, connection_sent, connection_accepted, dm_sent, email_sent, replied, follow_up_sent, personalization, connection_note_draft, dm_draft, email_draft

11. Append a run row to li_scrape_log.csv: date, search terms used, total profiles, new unique profiles, leads added, and ALL profile URLs scraped comma-separated.

12. At the end, run generate_drafts.py then regenerate_report.py, and open the dashboard.

Use direct REST calls with tokens from .env, never MCP, this has to run unattended on a schedule later."

PROMPT 4

Personalization from their recent posts

This one was not in my original plan. I added it during the build because a message that references someone's actual post reads completely differently from one that references their job title.

✍ TRY THIS PROMPT

"Find an Apify actor that scrapes a LinkedIn profile's recent posts. Before running it, tell me what it will cost, and tell me specifically whether I still get charged for profiles that have no posts.

Then run it on the profiles I already have, fetching only the single most recent post from the last 30 days per person.

For anyone who has one, save a short one-sentence summary of that post to a last_post_summary column in leads_data.csv. Strip URLs, emoji and repeated punctuation from the post text before summarizing it. If a person has no recent post, leave the column blank and the system will use a generic opener for them.

Do not write the outreach messages yet."

💡 TIP

Ask what something costs before you run it, every time. It takes one extra sentence and it is how you avoid surprises on the bill.

PROMPT 5

The message writer

✍ TRY THIS PROMPT

"Build generate_drafts.py. It writes a connection-request note and a follow-up DM for every lead in leads_data.csv, and an email draft ONLY for leads with an email found. It NEVER sends or connects, drafts only.

Follow the outreach copy rules in CLAUDE.md exactly. Specifics:

CONNECTION-REQUEST NOTE (exact structure, under 300 characters total, this is a hard platform limit):

Hi [First name], [one-sentence genuine opener referencing their role/company/recent post]. Would love to connect.

FOLLOW-UP DM TEMPLATE (sent after they accept, exact structure):

Thanks for connecting, [First name]!

Quick one, I made a short video on how [their role/industry]s are getting [SPECIFIC_RESULT from CLAUDE.md]. Want me to send it over? (smiley emoji)

[NAME]

EMAIL TEMPLATE (only generated when an email was found):

[Greeting]

[genuine opener]

Would you like to know how to get [SPECIFIC_RESULT from CLAUDE.md]?

I just made a quick video for you showing exactly how to do this.

I'm [NAME], [EXPERTISE] helping [target audience plural] [BENEFIT].

Just reply "yes" and I'll send you the video (smiley emoji)

[NAME]

THE OPENER (shared logic for the note, DM, and email):
- If ANTHROPIC_API_KEY is set: call Claude Haiku (model claude-haiku-4-5-20251001, max_tokens 60) with the lead's headline, company, and most recent post/activity if available, asking for ONE casual sentence under 15 words referencing something genuine. Explicitly tell it not to start with a greeting. Sanitize the result and strip any leading greeting it adds anyway.
- If no key, call fails, or no recent activity was scraped: fall back to a generic opener referencing their job title and company only.

ALSO BUILD:
- greeting(): first name only, always, per the LinkedIn-norm rule in CLAUDE.md.
- clean_quoted_text(): strips URLs, emoji, repeated punctuation, and em/en dashes from any scraped text before it goes in a message.
- enforce_connection_note_length(): hard-truncate or regenerate if the note exceeds 300 characters, never let one through over the limit, it will fail to send.

CRITICAL: if a lead's connection_sent or email_sent column is filled in, never regenerate that draft, that's the record of what was actually sent. Undelivered drafts can always be regenerated."

PROMPT 6

The dashboard

✍ **TRY THIS PROMPT**

"Build regenerate_report.py, producing a single self-contained LinkedIn Leads.html in this folder, overwritten in place each run.

Design:
- Purple gradient hero header with the title and a "last updated" timestamp
- Four stat cards: Total leads, With email, Connections sent, Replied
- A "How the score works" panel showing the point breakdown and the 70+/40-69/under 40 bands
- A search box and a status dropdown filter
- A table: avatar circle with initial, name linking to their LinkedIn profile, company, job title, score pill, status badge, chevron
- Clicking a row expands it to show the connection-request note, follow-up DM, and email draft (if one exists) side by side, each with a Copy button, each showing a live character count for the connection note so you can see at a glance if it's under 300
- overflow-x: auto on the table wrapper, not overflow: hidden
- Escape quotes properly in onclick handlers

Then create a Windows shortcut to this HTML file on my Desktop, using [Environment]::GetFolderPath("Desktop")."

PROMPT 7

Test it for real

✍ **TRY THIS PROMPT**

"Run: python scrape_leads.py --test
Show me the real output and open the dashboard."

⚠ **WATCH OUT**

Check the connection note is genuinely under 300 characters for every lead, not just on average. And read a few openers out loud. If one sounds creepy or generic, fix it now, before you scale up.

OPTIONAL

Run it every day by itself

Only do this once you are happy with the messages it writes, and once you have checked the cost against your API credit.

✍ **TRY THIS PROMPT**

"Register this in Windows Task Scheduler to run daily at 8am:
- Use the FULL absolute path to python.exe
- -StartWhenAvailable
- -MultipleInstances IgnoreNew
- Send the output to run_log.txt so I can see what happened if it fails
- Then read back the task state to confirm it actually registered"

PART 3

What I learned testing it

The real numbers, and the five things that broke.

THE NUMBERS

Measured across 39 real leads

These are the rates I actually got. Yours will vary by industry, but this is the right order of magnitude to plan with.

What	Rate
Profiles that were new, after deduplication	65%
Leads with a website linked on their profile	77%
Leads where an email was actually found	69%
Leads who posted on LinkedIn in the last 30 days	49%

What it costs

What you are buying	Cost
One LinkedIn profile, scraped and enriched	$0.010
One full lead, with email attempt, post and messages	$0.019
One actual email address	$0.027

An email address costs more than a lead because only about 7 in 10 leads yield one. To get 100 email addresses, expect to scrape roughly 145 profiles.

✨ **REAL EXAMPLE**

At 25 new leads a day, which is what LinkedIn realistically lets you contact, this runs at about 14 dollars a month. At 10 a day it is about 6 dollars a month.

THE FIVE THINGS THAT BROKE

So you recognise them faster than I did

1. The free tier caps each run at 10 profiles

My code asked for 25 at a time. Instead of an error, the whole batch came back as a single message saying free users are limited to 10 items per run, and nothing got saved.

Fix: tell Claude to process profiles in chunks of 10.

2. A field that looked like text was actually a bundle

The tool returns the profile it looked up in a field that turned out to hold a small package of data rather than a plain web address, so matching results back to the right person silently failed and every lead came back half empty.

Fix: if enriched data is mysteriously missing, ask Claude to print one raw result and check the actual shape of what came back.

3. Apostrophes broke the dashboard buttons

Message drafts contain words like "I'll" and "you're". Put those directly into a button and the apostrophe ends the button's text early and breaks the page. 13 of my 15 leads had one.

Fix: this is why Prompt 6 says to escape quotes properly. Leave that line in.

4. A test run that looked successful but did nothing

The test flag reuses the same search term every time. On the second run, deduplication correctly recognised all 15 people as already seen, so it added zero leads. The run succeeded, but none of the interesting parts of the system actually ran.

Fix: to genuinely test the whole chain, point it at a search term you have not used yet.

5. One small bug threw away a whole paid run

A single missing line crashed the script after it had already paid to search, enrich and fetch posts for 12 people. Because results were only saved at the very end, all of that work was lost.

Fix: ask Claude to handle each lead separately so one bad lead cannot destroy a run you have already paid for.

IS THE POST PERSONALIZATION WORTH IT?

Short answer: yes

It adds about 14 percent to the cost. You do pay a small fee for people who have no posts, roughly half the normal rate, and about half the people you find will not have posted recently.

It is still worth it. Compare these two openers, both produced by the system:

Without a post:: noticed you're a co-founder at Stealth Startup

With a post:: loved seeing the Amplificx rebrand to Asterion and the vision behind it

The second one reads like a person wrote it, because a person did the thing it refers to.

TWO QUESTIONS I COULD NOT ANSWER UP FRONT

Now answered

Does the search tool give you their posts?: No. Posts need a second, separate tool. Budget for that from the start if personalization matters to you.

Is the email channel worth keeping?: Yes. 77 percent of profiles link a website and 69 percent yield a real email. That is a proper channel, not a rare bonus.

> ☑ **BEFORE YOU RUN IT AT VOLUME**
> Read ten of the generated messages out loud. If any of them would make you cringe to receive, fix the wording before you scale up. The system will happily write a thousand bad messages just as fast as it writes ten good ones.

More free guides and build-alongs
fabimarkl.com/library