

Leverage Your *Subscription.*

How to turn the AI plan you already pay for into a team that works while you sleep: scheduled jobs, a local engine that costs nothing extra, a live site that never switches off, and exactly when it is worth paying for the API.

05:55

4 REEL SCRIPTS AND LINKEDIN
DRAFTS READY BEFORE I WAKE

£0

EXTRA ON TOP OF THE PLAN I
ALREADY PAY FOR

~\$186

WHAT THE SAME WORK WOULD
COST EACH MONTH ON THE
PAY-PER-USE API

START HERE

You are paying for a genius. Are you using it like a *search box*?

Most people open Claude, ask a question, copy the answer and close the tab. That is one-off use. But your subscription is a flat monthly fee, so **every hour it sits idle is capacity you paid for and did not use.**

The businesses pulling ahead do something different. They give the AI a standing job and a schedule, and it runs whether they are at their desk or asleep. This guide shows you the four levels, what each one costs, and how to climb them without a developer on the payroll.

LEVEL 1

Ask

One-off chats. Useful, but the work stops the moment you stop typing.

LEVEL 2

Schedule

The same brief runs daily or hourly on its own, inside the plan you already pay for.

LEVEL 3

Engine

Your own tool on your own computer, running on your subscription and catching up if it slept.

LEVEL 4

Site

A live website that runs day and night on the pay-per-use API, for when the laptop is shut or customers are involved.

£0 extra

My content engine writes **4 Reel scripts and a batch of LinkedIn drafts every morning, ready by 05:55.** It runs on the Claude plan I already had. The same work through the pay-per-use API would cost about \$1.55 per script: roughly \$186 a month.

One rule before you start: **automate the job you already do every week, not the one you wish you did.** A boring task that runs every single day beats a clever one you never finish.

The biggest difference is *the cron job*

In June 2026, Perplexity's CEO Aravind Srinivas sat down with Harry Stebbings on 20VC. He described customers spending upwards of \$10,000 a month on Perplexity's agent product, and not wasting it: their businesses run on agents working in loops. When he looked at what separates his heaviest users from everyone else, the answer was not better prompts.

"The single biggest differentiation ... is whether they run repetitive cron jobs."

ARAVIND SRINIVAS, CEO OF PERPLEXITY, ON 20VC (AT 17:49)



WATCH THE EPISODE

Perplexity CEO: Micron Will Be More Valuable Than Meta & How Export Controls Helped Not Hurt China

20VC with Harry Stebbings · 15 June 2026 · 1 hr 35 min

▶ [Jump to the cron job moment \(16:55\)](#)

▶ [Watch the full episode](#)

ONE-OFF USE

You ask, it answers, it stops.

A research report. A single email. A summary of one document. Valuable, but it only happens when you remember to ask, and it stops the moment you close the tab.

REPETITIVE USE

It keeps watching and keeps working.

Every inbound email gets triaged. Every alert gets investigated and routed to the right person. Every morning the briefing is waiting. His phrase for it: workflows that keep running all the time.

He argued this always-on use is where the money in AI is heading, and said Perplexity's revenue has more than tripled since the start of the year. He also named the catch: running a top model nonstop on servers is too expensive for most people, so the winning setups mix your own computer with the cloud. **That is precisely the setup this guide teaches.**

A cron job is a standing order *with a clock*

"Every weekday at 6am, do this." That is all a cron job is. The name comes from the scheduler that has quietly run the world's servers for decades. The difference now is what you can put inside it: instead of a fixed script, **a brief that an AI carries out with judgement.**

The anatomy of a job that works

1 Trigger When it runs: a time, or an event such as a new email or a form submission.	2 Input What it reads: your inbox, a folder, the news in your niche, your CRM.	3 Brief Standing instructions, written once and in detail, including your voice and your rules.	4 Output Where results land: a draft, a document, your phone. Never auto-sent at first.	5 Check How you know it worked: a one-line report every morning, including failures.
---	--	---	---	--

Six jobs a business can schedule this week

WHEN	JOB	WHAT LANDS ON YOUR DESK
06:00 daily	Morning briefing	Today's calendar, the five emails that matter, market news, and your top three priorities in one page.
07:00 daily	Lead research	Every new lead from yesterday researched: company, recent news, likely pain, and a tailored opening line.
Hourly	Inbox triage	Messages sorted and labelled, with reply drafts waiting for your approval.
Nightly	Content drafts	Posts, scripts and newsletters in your voice, built from the day's news and your own wins.
Monday 08:00	Weekly numbers	Sales, pipeline and marketing figures pulled together and written up as a short report.
Friday 16:00	Competitor watch	Changes to competitor pages, pricing and offers. Only what changed, nothing else.

Rule

Run the brief by hand three times before you schedule it. **If you would not trust the output while watching, you will not trust it while sleeping.**

YOUR OPTIONS

Three ways to run *while you sleep*

Claude subscribers already have two schedulers built in, and you can build a third. Here is how they compare.

	Desktop scheduled task	Cloud routine	Your own engine + live site
RUNS ON	Your computer, inside the Claude desktop app	Anthropic's cloud	Your computer, plus a hosting provider for the site
COMPUTER ON?	Yes: awake, with the app open	No	Engine yes, site no
YOUR LOCAL FILES	Yes	No: works from a GitHub repo and connected apps	Yes, on the engine side
PAID FROM	Your subscription	Your subscription, with a daily cap on runs	Subscription locally; pay-per-use API when hosted
MOST OFTEN	Every minute	Every hour	Anything you choose
MISSED A RUN?	One catch-up run when the computer wakes	Not affected by your computer	Your design: mine catches up on wake
BEST FOR	Personal jobs on your own files	Repo and connector work with the lid shut	Products, teams and anything customer-facing

START HERE

No code needed for the first two

In the Claude desktop app, open the Code tab, then Routines, then New routine. Choose **Local** for a desktop task or **Cloud** for a routine. Or simply describe it in a chat: "every weekday at 7am, research yesterday's new leads".

NOT ON CLAUDE?

Others have lighter versions

ChatGPT has Tasks, Gemini has scheduled actions (Google AI Pro and Ultra), and Perplexity has Tasks (Pro and Enterprise). Great for briefings and reminders, but they live inside the chat app: they cannot reach your own files or run your own tools.

Routines are a research preview, so limits may change. Your remaining daily runs are shown at claude.ai/code/routines. A desktop task skips runs while your computer sleeps; turn on "Keep computer awake" in the desktop app's settings if timing matters.

CASE STUDY

Inside the content engine *I built*

I post short videos and LinkedIn content every day. Writing them used to take my mornings. Now a small engine on my Mac does the first draft of everything before I wake up, and it costs nothing on top of the Claude plan I already pay for.

- 05:30** ● **My Mac's scheduler starts the job.** If the laptop was asleep, it starts the moment I open the lid instead. No run is ever silently lost.
- then** ● **It reads the day's news in my niche** and picks the stories worth talking about, then checks my bank of real stories from my own client work.
- then** ● **It writes 3 news scripts and 1 story from my own work.** Each comes with hooks, the full script, a teleprompter version, an Instagram caption, a LinkedIn post, its sources and a fact check.
- 05:55** ● **Everything is saved to a cloud database.** I read it on my phone over coffee, through the live site. The laptop never needs to be in the room.

4 + drafts

REEL SCRIPTS PLUS
LINKEDIN DRAFTS
EVERY MORNING

~25 min

FOR THE WHOLE RUN,
WHILE I SLEEP

£0

EXTRA ON TOP OF MY
CLAUDE PLAN

\$1.55

PER SCRIPT IF THE
SAME JOB RAN ON
THE API

THE "WAIT OR PAY" BUTTON

Nothing spends money silently

When I want an extra script in the middle of the day, the live site asks me: **wait for my Mac** (free, picked up within 15 minutes) or **write it now on the API** (about \$1.55). I choose, every time.

WHAT MADE IT RELIABLE

- **One shared database**, so the laptop and the site always agree.
- **A scheduler that catches up** after sleep instead of skipping.
- **A one-line report** every morning, failures included.
- **One automatic retry** when the AI's output comes back malformed.

Local engine or live site? *Use both.*

YOUR LOCAL VERSION

Free, private and fast

- **Free to run:** uses the subscription you already pay for.
- **Private:** reads files that never leave your machine.
- **Fast to change:** edit, refresh, done. No deploy.
- **Your own hardware:** uses the computer you already own instead of rented servers.

Limits: runs only while the computer is awake, and only you can use it.

YOUR LIVE SITE

Always on, anywhere, shareable

- **Works whether your computer is on or off.**
- **Open it from your phone,** anywhere.
- **Share it** with your team or your customers.
- **Reliable schedules:** hosting providers run cron jobs on their own clocks.

Limits: a website cannot log in to your personal subscription, so it pays per use.

So which API does the site use?

The Claude API. You create a key at platform.claude.com, add a card, and pay per million tokens (in / out): **Haiku 4.5 \$1 / \$5, Sonnet 5 \$2 / \$10, Opus 5.5 \$4 / \$20.** Web search is \$10 per 1,000 searches. Overnight work that can wait up to 24 hours can go through the **Batch API at 50% off.** The key lives in your hosting provider's settings, never in your code.

The best of both: the site is the front door, your computer is the engine

1 Everything lives in one cloud database

The site and the engine read and write the same data, so they never disagree.

2 Jobs started on the site wait in a queue

Your computer checks the queue and does the work on your subscription.

3 Scheduled work runs locally for free

Your engine handles the daily run, catching up if it slept through it.

4 Urgent work goes to the API, on request

Only when you choose "now", and only for that one job.

Turn your project into a *desktop app*

A tool you have to start from a terminal is a tool you stop using. Give it an icon. There are three ways, from two minutes of effort to a proper product.

01 A one-click launcher

YOUR OWN MACHINE

HOW

- On a Mac, Automator or Shortcuts can wrap a small script as an app with its own icon.
- On Windows, a desktop shortcut to a .bat file does the same job.
- The script starts your local version and opens it in the browser.

This is what I use. One lesson: make the launcher check whether the code has changed and rebuild if so. Otherwise it will happily keep serving yesterday's version, and you will chase bugs you already fixed.

02 Install your live site as an app

ZERO CODE

HOW

- Safari on macOS Sonoma or later: File, then Add to Dock.
- Chrome and Edge: the install option in the browser menu.
- It opens in its own window, with its own icon, like any other app.

Best for your team and for you on the go. **It works when your computer is off**, because it is really your live site in a clean window.

03 A packaged desktop app

TO SHARE OR SELL

HOW

- Tauri (lightweight) or Electron (the framework behind VS Code) bundle your project into an installer.
- To share a Mac app without security warnings, sign and notarise it through the Apple Developer Program (\$99 a year).
- Build in automatic updates from day one.

Remember the rule on page 10: your subscription cannot power other people's copies. Each user signs in with their own plan, or the app calls your server, which pays for the API.

Eight things to keep *in mind*

Most of these came from real mornings building my own engine. Plan for all eight from the start and your engine can run for months untouched.

1 Software updates change things

The Claude command-line tool updates itself. Test your job after an update. One wrong setting can stop a job using your subscription login: I hit exactly that while building mine.

2 Stay signed in

Scheduled jobs use your desktop login. If it expires, jobs fail quietly. Make the morning report say so loudly.

3 Sleep is the enemy

Lid closed means nothing runs. Use a scheduler that catches up on wake (the Mac's launchd does; plain cron does not), or keep the computer awake.

4 New computer, new problems

Moving to a new Mac can leave behind old tools built for a different chip. Check your runtime (Node, Python) actually runs before relying on it.

5 One source of truth

Keep data in one cloud database. My only confusing moment came from a test copy pointing at a test database.

6 Limits are shared

Automated jobs draw from the same usage allowance as your chats. Run heavy jobs early, so they do not eat into your working day.

7 The rules can change

In May 2026 Anthropic announced moving automated use to a separate credit, then paused it on the day it was due. Build a switch that moves any job to the API with one setting.

8 Secrets stay secret

Keys belong in settings files or your keychain. Never in prompts, code, screenshots or chat messages.

Test

Could you go on holiday for a week and trust it? **If a failure would go unnoticed until you got back, add the morning report before you add another feature.**

Your subscription is *for you*

This is the part most "free AI hack" videos skip. Get it right and you are on solid ground. Get it wrong and you risk the account your business depends on.

FINE (AS OF SEPTEMBER 2026)

- **Running Claude Code on your own computer**, including scheduled headless jobs (`claude -p`). It is Anthropic's own tool, and its help centre confirms these jobs draw from your plan's usage limits.
- **Cloud routines and desktop scheduled tasks**: built-in features of your plan.
- **Automating your own work**: your content, your research, your admin, your reports.

NOT FINE

- ✗ **Sharing your login** or making your account available to anyone else.
- ✗ **Reselling access** to what your plan gives you.
- ✗ **Letting customers trigger work** that runs on your personal plan.
- ✗ **Driving the claude.ai website with bots**. The consumer terms ban automated access except through an API key or where Anthropic explicitly permits it.

The test

If a customer presses the button, it runs on the API. If you press it, or your own schedule does, your subscription can cover it. Build that line into your product from day one and you never have to rebuild it later.

On a Team or Enterprise plan, your business agreement applies instead and your admin controls which features are switched on. Rules in this space move quickly, so before you build anything client-facing, read the current terms at anthropic.com/legal and the Claude help centre. This guide is practical advice, not legal advice.

12 projects that pay for *the subscription*

01 On-demand content writer

SUBSCRIPTION

NIGHTLY + ON DEMAND

Drafts posts, scripts and newsletters in your voice from the day's news and your own client wins. You review over breakfast instead of staring at a blank page.

A week of content before your first coffee.

02 Image studio

HYBRID

ON DEMAND

Claude writes the creative brief and the image prompts on your plan. An image API renders them for pennies (about \$0.02 to \$0.24 per image on Google's models). Claude itself does not make images.

On-brand graphics in batches, not one at a time.

03 Document factory

SUBSCRIPTION

ON DEMAND

Proposals, statements of work, reports and policies built from your templates and your best past work, then exported as a finished PDF.

The proposal goes out the same afternoon as the call.

04 Morning briefing

SUBSCRIPTION

06:00 DAILY

Calendar, the emails that matter, market news and your top three priorities on one page. A cloud routine sends it even when your laptop is shut.

You start the day decided, not reacting.

05 Lead research desk

SUBSCRIPTION

OVERNIGHT

Every new lead researched while you sleep: company, recent news, likely pain, and a tailored opening line, written into your CRM notes.

Every call starts warm.

06 Inbox triage

SUBSCRIPTION

HOURLY

Sorts, labels and drafts replies to everything that lands. It never sends a word until you have watched it long enough to trust it.

Inbox zero without the hour it usually costs.

And six more *for the back office*

07 Meeting follow-up

SUBSCRIPTION

AFTER EVERY CALL

When a call transcript lands in a folder, it writes the summary, the action list with owners, and the follow-up email draft.

Follow-ups out before the other side has written theirs.

08 Client reporting

SUBSCRIPTION

MONDAY 07:00

Pulls each client's weekly numbers and writes them up as a branded PDF with plain-English commentary, ready for you to check and send.

Reporting day becomes a ten-minute review.

09 Competitor and price watch

SUBSCRIPTION

WEEKLY

Checks competitor websites, pricing and offers, and reports only what changed since last week, with a note on what it means for you.

You hear about their price change before your customers do.

10 Website health check

SUBSCRIPTION

NIGHTLY

Finds broken links, slow pages, forms that quietly stopped working and search gaps, then hands you a prioritised fix list.

No more finding out from a lost lead.

11 Cash chaser

SUBSCRIPTION

DAILY

Spots overdue invoices and drafts polite reminders that get firmer over time, each one waiting for your approval.

Paid faster, without the awkward emails.

12 Customer-facing assistant

API

24/7

Anything your customers use directly: a quote builder, a support helper, a document generator on your website. This one belongs on the pay-per-use API.

Price it so each use earns more than it costs.

How to pick your first one

Choose the job that **happens most often**, **follows the same steps each time**, and **would hurt least if the first draft were wrong**. That is almost always content, research or reporting. Save anything customer-facing for later.

THE NUMBERS

What it actually *costs*

ITEM	PRICE	WHAT YOU GET	UPGRADE WHEN
YOUR SUBSCRIPTION			
Claude Pro	\$20/mo, or \$17/mo billed yearly	Claude Code, desktop scheduled tasks and cloud routines	You hit your limits every week
Claude Max 5x	From \$100/mo	5x Pro's usage per session	Automations and daily work start to collide
Claude Max 20x	\$200/mo	20x Pro's usage per session	You run several engines every day
PAY PER USE			
Claude API	Per million tokens (in / out): Haiku 4.5 \$1 / \$5, Sonnet 5 \$2 / \$10, Opus 5.5 \$4 / \$20	Live sites and anything customers trigger	Use the Batch API for 50% off work that can wait
Web search (API)	\$10 per 1,000 searches	Fresh facts for jobs running on the API	Only needed on the API side
Image APIs	About \$0.02 to \$0.24 per image (Google's models)	Rendering the prompts Claude writes	Use the cheaper models for drafts
HOSTING (ONLY IF YOU WANT A LIVE SITE)			
Vercel	Hobby: free. Pro: \$20/mo per user	Hobby runs cron at most once a day; Pro can run it every minute	You need more than one run a day
Neon database	Free: 0.5 GB per project. Paid from \$5/mo	The one shared database your engine and site both use	Your data outgrows the free tier
Apple Developer	\$99 a year	Signing and notarising a Mac app	Only if you share a packaged app

Your plan + £0

That is the honest starting cost. Desktop tasks, routines and a local engine all run inside the subscription you already pay for. **Add \$20 a month only when you want a live site with frequent schedules, and pay per use only for work other people trigger.**

All prices in US dollars as published by each provider. UK prices include VAT and can differ slightly.

The one-weekend *challenge*

You do not need the whole system. You need one job running on a schedule by Monday morning. Here is the order.

1 List five repeat tasks

Everything you or your team did more than twice last week. Be honest about the boring ones.

2 Pick the daily one

Highest frequency, same steps each time, lowest risk if the draft is wrong.

3 Write the brief

What to read, what to produce, your voice, your rules, and what good looks like. One page.

4 Run it by hand three times

Fix the brief each time. Stop when you would happily use the output.

5 Schedule it

A desktop task if it needs your files, a cloud routine if it must run with the lid shut.

6 Send it where you already look

Your inbox, a shared folder or your phone. A result you have to go looking for is a result you ignore.

7 Add the morning report

One line: ran or did not, how long, anything that failed.

8 Review on Friday, add one more

Tighten the brief with what you edited that week. Then pick job number two.

The challenge

By Monday, **one job runs without you**. By the end of the month, five. That is when the subscription stops being a monthly cost and starts being **the cheapest member of your team**.

Copy-paste setup: *the engine*

For readers who want to build their own engine. Everything here runs on your own computer using the Claude Code command-line tool, signed in with your subscription.

A · RUN CLAUDE ON A JOB WITHOUT OPENING A CHAT

`-p` prints the result and exits, which is what a scheduler needs.

```
claude -p "Read notes/today.md and write three LinkedIn post drafts in my voice. Save them to
drafts/linkedin.md." \
  --model sonnet --allowedTools "Read,Write" --output-format json
```

B · MAC: RUN IT AT 05:30 EVERY DAY, AND CATCH UP AFTER SLEEP

Save as `~/Library/LaunchAgents/com.you.morning-job.plist`. The Mac's launchd runs a missed calendar job when the computer wakes; plain cron would skip it.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0"><dict>
  <key>Label</key><string>com.you.morning-job</string>
  <key>ProgramArguments</key>
  <array><string>/bin/zsh</string><string>-lc</string><string>$HOME/jobs/morning.sh</string></array>
  <key>StartCalendarInterval</key>
  <dict><key>Hour</key><integer>5</integer><key>Minute</key><integer>30</integer></dict>
  <key>StandardOutPath</key><string>/tmp/morning-job.log</string>
  <key>StandardErrorPath</key><string>/tmp/morning-job.log</string>
</dict></plist>
```

```
# switch it on (and again after any edit)
launchctl bootstrap gui/${id -u} ~/Library/LaunchAgents/com.you.morning-job.plist
```

Tip: schedulers start with a bare environment. The `-lc` above loads your normal shell setup, so `claude` and `node` are found.

C · WINDOWS: TASK SCHEDULER

```
schtasks /create /tn "MorningJob" /tr "C:\jobs\morning.bat" /sc daily /st 05:30
```

Then open Task Scheduler, find MorningJob, and on the Settings tab tick "Run task as soon as possible after a scheduled start is missed".

Copy-paste setup: *the site*

D · LINUX SERVER: CLASSIC CRON

```
# crontab -e    (minute hour day month weekday)
30 5 * * * /home/you/jobs/morning.sh >> /home/you/jobs/morning.log 2>&1
```

Reliable on an always-on server. On a laptop, cron skips anything scheduled while it sleeps.

E · A HOSTED SCHEDULE ON VERCEL

Add to `vercel.json`. The route it calls runs on the pay-per-use API, and should check a secret so strangers cannot trigger it.

```
{ "crons": [ { "path": "/api/cron/daily", "schedule": "30 5 * * *" } ] }
```

F · THE "WAIT OR PAY" PATTERN

The site never spends money unless the user chooses to. Written as plain steps so it fits any language.

```
when the user asks for a job:
  ask: "Wait for my computer (free)" or "Run it now on the API (about $X)"
  if wait:  save the job to the database with status QUEUED
  if now:   call the Claude API with the key from settings, save the result

on your computer, every 15 minutes and on wake:
  take every QUEUED job from the database
  run it with:  claude -p "<the job's brief>"
  save the result, mark the job DONE (or FAILED, with the reason)
```

G · BEFORE YOU TRUST IT

- **Log every run** to a file, and have the morning report read it back to you.
- **Make runs safe to repeat.** If the job starts twice, the second run should notice and stop.
- **Parse the AI's output defensively** and retry once. It will occasionally answer in the wrong shape.
- **Put the model name in a setting**, not the code. Models change; your code should not have to.
- **Keep one switch** that moves every job from subscription to API, in case the rules change.

LAST WORD

Stop asking. Start *scheduling*.

You already pay for the most capable assistant your business has ever had. Right now it waits for you to type. The people getting the most from AI gave it a job, a schedule and a place to put the results, then went to sleep.

Pick one job this weekend. Write the brief, run it three times, put it on a clock. On Monday morning, the work will already be done.

Want help building your own engine, or deciding what to automate first? That is what we do.

[Learn more at b2blcc.com](https://b2blcc.com) →