10 systems · 1 build prompt

How I automated 90% of my tech startup.

The ten boring systems I built with Claude to run my startup, plus the exact prompt I use to build each one.

One job each, every day, without me.

Fadé Adeniyi

THE PREMISE

How did I automate 90% of my startup?

I didn't automate my startup with one big clever agent. As co-founder and CEO of Techlerator, the company behind Xeruit, I built ten small, boring systems with Claude. Each one does a single job every day. Together they run about 90% of the work.

Every one of them has the same five parts. Once you can see the parts, you can build your own, one at a time.

Every system has five parts

Reads	One place it looks: an inbox, a calendar, a list of companies, a meeting.
Decides	One judgement Claude makes each time, like which emails need you.
Sends	One place the result lands, where you already look.
Runs	A fixed time, or a command you type. It doesn't wait to be remembered.
Never	The line it must not cross, written down before you build.

STEP 1

The prompt I build every system with

Paste this into Claude Code and fill in the brackets. It's the brief I start every system with. Claude asks its questions first, builds a dry run and only then switches it on.

```
I want you to build me one small system that runs on its own. It does one job, every
time, without me. Read my answers below, then follow the build rules.

SYSTEM: [A short name, e.g. Morning brief]

JOB: [The one job it does, in one sentence]

READS: [What it looks at: my inbox, my calendar, a folder, a spreadsheet, a website]

DECIDES: [What you should judge each time, e.g. which emails need me]

SENDS TO: [Where the result goes: Google Chat, Slack, an email to me, a Google Doc]

RUNS: [When: every day at 7am, every hour, or only when I type a command]

NEVER: [What it must never do, e.g. send an email without my approval]

BUILD RULES:

1. Ask me up to five questions before you write any code. Don't guess the parts you
can't see.
2. Use the simplest setup that works on my computer. Explain every tool you install
and why.
3. Keep passwords and keys in a .env file. Never write them into the code.
4. Build a dry run first that shows me what it would send without sending it. Wait
for my OK.
5. Anything that goes to another person needs an on/off switch I control.
6. Then schedule it to run on its own and show me how to stop it.
7. Finish with a short README: what it does, how to run it, how to switch it off.

Start with your questions.
```

Fill in NEVER before anything else. It's the line that keeps a system from doing damage while you sleep.

THE TEN

What each of my ten systems does

1. Product brain	Holds 16 years of my product experience. Writes requirements, tests what we build and runs the roadmap with me.
2. Client finder	Finds companies that fit who we sell to, checks the contact and works out why we should write.
3. Outreach email	Writes to those companies from its own address. Their emails stop when they reply.
4. Meeting agent	Listens to my Google Meet calls, catches every ask and drafts the work in Drive.
5. Morning brief	At 7am, reads my inboxes and calendar and sends one message.
6. Inbox watch	Every hour, messages me only about emails that need me.
7. Market analyst	At 8am, reads the AI news and what competitors did, then says what it means for us.
8. Hiring manager finder	Finds the person behind each new job post and drafts an email. I approve each one.
9. Control room	One page to switch outreach on or off and read, edit or stop any email.
10. Co-founder agent	Argues with my ideas before I commit to them.

STEPS 2 AND 3

Start small and keep it safe

Start with the morning brief. It only reads and only messages you, so nothing can go wrong for anyone else. Here it is filled in.

```
SYSTEM: Morning brief

JOB: Tell me my day before I open my laptop.

READS: My Gmail inbox and today's Google Calendar.

DECIDES: Which emails need a reply from me today, what's on my calendar and the one
thing I should do first.

SENDS TO: One message in Google Chat.

RUNS: Every day at 7am.

NEVER: Reply to, delete or archive any email.
```

1. **Build one system at a time.** Get it running for a week before you start the next.
2. **Run the dry run until it's boring.** If you'd be embarrassed by what it would have sent, fix the brief, not the output.
3. **Give anything outward-facing a switch.** My outreach has an on/off switch and one page where I can stop any email.
4. **Send results where you already look.** I use Google Chat, so I read them without opening anything new.

> **THE TRAP**
>
> The trap is building one agent that does everything. It's hard to test, hard to trust and you can't tell which part broke. Ten boring systems with one job each are easier to check, easier to fix and easier to switch off.

THE TAKEAWAY

Build the boring one first.

Paste the build prompt into Claude Code today and fill it in for a morning brief. When it has run quietly for a week, pick the next job you're tired of doing.

How I use it. All ten run from my own computer. The scheduled ones start on a timer each morning or every hour, read my email and calendar and message me with what they found. My outreach only sends while its switch is on and I can stop any email from one page. My hiring manager finder only drafts. I approve every one.

What this won't do: it won't run while your computer is off, unless you move it to a server. Google sign-ins can expire and need redoing. Agents make mistakes, so keep a human check on anything that reaches another person. If you send cold email, check the rules where you are. In the UK that's PECR.

One job each. Every single day.

Built from my own systems as they run in November 2026. Tool names and menus change often, so follow what Claude Code asks for on your machine.

Building something of your own? **The Audacity to Build** is my community for people building alongside me, with the systems I actually run. It is opening soon. Join the waitlist at fadeadeniyi.com/waitlist.

Fadé Adeniyi

Fadé & The Machines