

The AI Executive Team

Build an AI C-suite for your business. 5 chief agents + specialists, each with a charter, dispatched in parallel from one prompt. The system that runs the other seven without you holding it all in your head.

Built by [@brycenwood.ai](#) · October 2026

THE PROBLEM

One AI is good. A team of them is unreasonable.

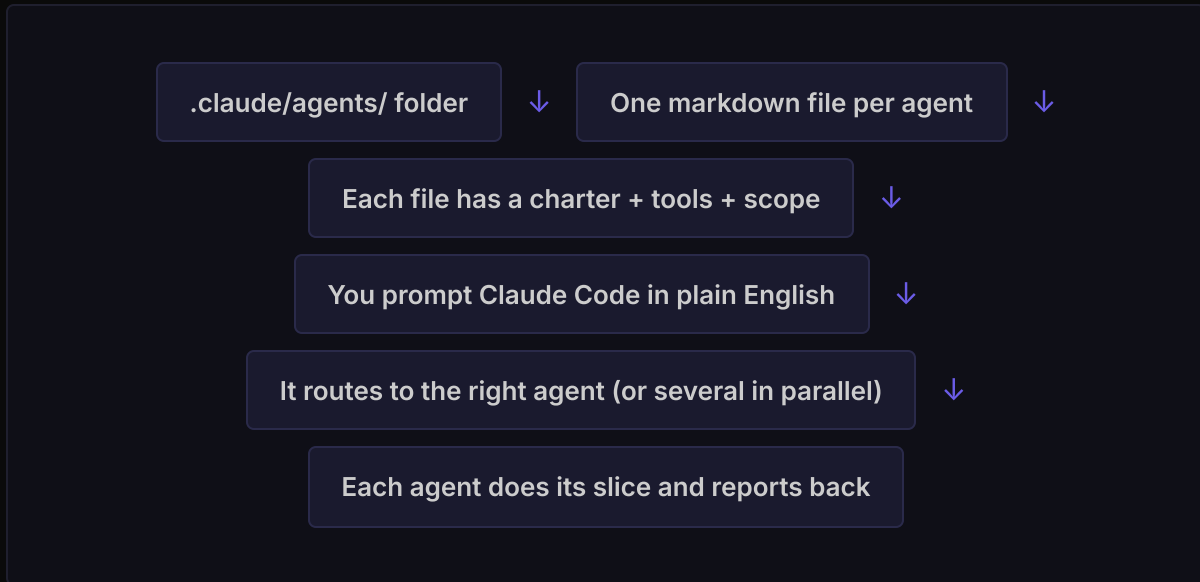
By System 7, you have Claude Code talking to your CRM. You can pull stale leads in one sentence, audit your whole pipeline, build workflows. It is wild — until you try to do all of it on the same day. Then you remember why you wanted help in the first place. There are too many parts of a business for one prompt to hold at once.

Real companies solve this with an executive team. A CMO who only thinks about marketing. A CFO who only thinks about money. A COO who keeps operations running. Each one has a defined scope. Each one knows when to escalate. None of them have to hold the whole business in their head — they just hold their slice and hand off the rest.

System 8 builds that team for you, in Claude Code. Five chief agents, each with their own charter. Specialists under them for the work the chiefs don't do directly. You dispatch one of them with a sentence and they go to work. You dispatch all of them and they go to work in parallel. While you sleep. While you film. While you go to the gym.

What You're Building

An `.claude/agents/` folder inside your project. Each agent is a single markdown file with a name, a description of when to use it, the tools it can access, and a system prompt that defines its role. Claude Code reads the folder automatically and routes your prompts to the right agent — or spawns several of them at once.



HOW THIS STACKS WITH THE REST OF THE PLAYBOOK

Systems 1-5 build your visibility. Systems 6 + 7 connect Claude Code to your CRM so it can act. System 8 turns that one Claude Code into an organized team. Your CRO agent watches the leads from System 6. Your CMO agent owns the SEO from Systems 1-3. Your COO agent makes sure the website (System 5) and YouTube (System 4) keep running. They each report to your Chief of Staff agent — and your Chief of Staff reports to you.

BEFORE YOU START

What You Need

- **Claude Code** installed (the CLI tool, not `claude.ai`)
- **An existing project folder** where you already use Claude Code (the same folder where you built System 6 or 7 works)
- **10-20 minutes** to build your first chief agent and dispatch it
- That is it. No new accounts, no new APIs, no new costs. Subagents are built into Claude Code.

TIME + COST

End to end: about 30 minutes the first time to set up your first 2 chief agents. After that, every new agent takes 5-10 minutes. Cost: roughly \$1/month extra in Anthropic API usage from running the team — most of your cost is still you talking to Claude Code directly.

STEP 1

Stand up the agents folder

Open Claude Code in your project. The first move is creating the folder Claude Code looks at to find your team.

PASTE THIS INTO CLAUDE CODE

```
I want to build an AI executive team inside this project using Claude Code subagents.
```

```
Create a .claude/agents/ folder in this project. Then create a README inside that folder explaining:
```

- That each .md file in this folder is one agent
- The required frontmatter for an agent file (name, description, tools)
- The pattern for writing the system prompt as the body of the file
- A note that subagents run in their own context window, so they don't pollute the main conversation

```
Show me the README when done.
```

WHY A FOLDER OF MARKDOWN FILES

This is the actual format Claude Code uses to register subagents. Each .md file becomes a callable agent. You can edit them in any text editor, version them in git, and copy them between projects. No platform lock-in.

STEP 2

Build your first chief agent — Chief of Staff

The Chief of Staff is the agent you start with because everything else routes through it. Its job is to hold your strategic priorities across every part of your business and tell you (or other agents) what to work on right now. It is the closest thing to having a real human chief of staff who actually keeps the map in their head for you.

PASTE THIS INTO CLAUDE CODE

Create my first subagent inside `.claude/agents/`.

File: `.claude/agents/chief-of-staff.md`

Required frontmatter:

- name: chief-of-staff
- description: Strategic router and weekly review owner. Holds priorities across all parts of my business. Use for "what should I work on right now" decisions, weekly reviews, and cross-priority calls.
- tools: Read, Edit, Write, Bash, Glob, Grep, WebFetch

System prompt body should establish:

1. The agent's role (chief of staff for my business)
2. The scope of what it owns (strategic priorities, weekly reviews, cross-priority decisions)
3. The scope of what it does NOT own (it does not run marketing campaigns, it does not touch the CRM directly, it does not write code – it routes those to specialist agents)
4. The format it should respond in (always lead with the recommended next action, then the reasoning, never bury the answer)
5. When it should escalate back to me (anything involving money over \$X, anything legal, anything personal)

Also: the system prompt should make this agent talk to me like a homie who has my best interest in mind, not like corporate consultant. Direct, casual, no fluff.

Build the file. Then read it back to me so I can confirm.

THE CHARTER IS THE MAGIC

An agent's charter is the system prompt. It defines what the agent owns, what it doesn't, how it should respond, and when to escalate. If you spend an extra 5 minutes making the charter sharp, the agent gets dramatically better. If the charter is vague, the agent will be vague back. Treat it like a job description for a real hire.

Add your other four chiefs

Now repeat the pattern for your other four executive agents. CMO, CRO, COO, CFO. Each one gets their own .md file in `.claude/agents/` with a frontmatter block and a charter.

PASTE THIS INTO CLAUDE CODE

Create the other four chief agents in `.claude/agents/`. Same format as `chief-of-staff.md`. For each one, write a sharp charter that defines scope, format, and escalation rules.

1. `.claude/agents/chief-marketing-officer.md`
 - Owns: brand strategy, content calendar, ad performance, audience growth, partnerships
 - Does NOT own: closing deals, billing, operations
 - Tools: Read, Edit, Write, Bash, Glob, Grep, WebFetch, WebSearch
2. `.claude/agents/chief-revenue-officer.md`
 - Owns: pipeline integrity, conversion rate, sales cycle time, win/loss analysis, hot inbound
 - Does NOT own: product delivery, brand decisions, capital allocation
 - Tools: Read, Edit, Write, Bash, Glob, Grep, WebFetch
3. `.claude/agents/chief-operating-officer.md`
 - Owns: calendar integrity, vendor uptime, SOPs, dashboard health, integration QA, HR ops
 - Does NOT own: strategic priorities, marketing, finance
 - Tools: Read, Edit, Write, Bash, Glob, Grep, WebFetch
4. `.claude/agents/chief-financial-officer.md`
 - Owns: P&L per business, AR, capital allocation, pricing strategy, owner pay policy, Profit First allocation
 - Does NOT own: marketing spend decisions (CMO partners on those), operational vendors (COO owns)
 - Tools: Read, Edit, Write, Bash, Glob, Grep, WebFetch

Build all four. Then list every agent now in `.claude/agents/` so I can confirm the team is staffed.

YOU CAN CHANGE THE NAMES

If you are not running multiple businesses, you do not need a CRO and a CMO and a COO. You might just need a "marketing agent" and a "sales agent" and a "ops

agent." The names are yours. The pattern is the same. Define the scope sharply, define what it does not do, give it the tools it needs.

STEP 4

Add the specialists

Chiefs do strategic work. Specialists do executional work. Below your CRO, you might have a "closer agent" that drafts hot inbound replies. Below your CMO, a "content agent" that runs your post pipeline. Below your CFO, a "bookkeeper agent" that reconciles your accounting daily.

You do not need 13 specialists tomorrow. You need 1 specialist for whichever lane is most painful today.

PASTE THIS INTO CLAUDE CODE

```
I want to add a specialist agent under one of my chiefs. The most painful unowned work in my business right now is [DESCRIBE THE WORK – e.g. "drafting hot inbound replies", "reconciling QBO transactions daily", "auditing my SEO rankings weekly"].
```

```
Pick a sensible name for this specialist. Create .claude/agents/[specialist-name].md with:
```

- name + description (when to use)
- tools (only what it actually needs – read-only for audit agents, write access for content/messaging agents)
- A sharp charter that explains:
 - The exact scope of what this agent owns
 - Which chief it reports to
 - The format of its outputs (always show me the draft before sending, never auto-send messages, etc.)
- When to escalate to its chief vs back to me directly

```
After you build it, dispatch this new specialist with a small test task to confirm it works.
```

READ-ONLY BY DEFAULT

When you build a new specialist, give it the smallest tool set it needs. An audit agent only needs Read + Glob + Grep. A reporting agent only needs Read + Bash. Only give Write or send-message tools to agents whose entire job is to produce

drafts you approve. This is the same principle as least-privilege access for human employees.

STEP 5

Dispatch one agent

Now you actually use the team. The simplest dispatch is asking Claude Code to run a single agent on a single task.

PASTE THIS INTO CLAUDE CODE

Spawn the chief-of-staff agent. Give it this brief:

GOAL: Tell me what I should focus on this week.

CONTEXT: Read my latest dossier in `_data/reports/daily/`, look at my open tasks in `_data/personal/tasks.json`, and check git log for what shipped in the last 7 days.

CONSTRAINTS: Maximum 3 priorities. Each one needs a "why now" reason. Skip vanity work.

DEFINITION OF DONE: A 3-bullet list with rationale, sent to me in this conversation.

Run it. Show me the report when it comes back.

THE DISPATCH CONTRACT

GOAL / CONTEXT / CONSTRAINTS / DEFINITION OF DONE is the format that makes agent dispatches reliable. Every time you brief an agent, give it those four things. No agent does its best work from a one-line prompt. Treat them like sharp humans who need a written brief.

STEP 6

Spawn the team in parallel

This is the moment the system gets unreasonable. You ask Claude Code one question. It dispatches multiple agents at the same time. They each do their slice in their own context window. They report back. You read the synthesis.

PASTE THIS INTO CLAUDE CODE

I want a snapshot of where my business stands right now. Spawn my CMO, CRO, COO, and CFO in parallel. Each one writes a 5-bullet brief from their lane.

CMO brief: state of brand + content + audience growth this week. Top 1 thing to push on.

CRO brief: state of the pipeline. Hot leads. Stalled deals. Top 1 thing to close.

COO brief: state of operations. What is slipping. Top 1 thing to firm up.

CFO brief: state of money. Top 1 thing to chase or pay.

Run them in parallel, not in sequence. After all four return, synthesize the four briefs into a single "what matters most this week" recommendation. Show me each brief AND the synthesis.

WHY PARALLEL MATTERS

If you ran these one at a time, the conversation would burn through tokens carrying every previous brief into the next agent's context. By spawning them in parallel, each agent gets its own clean context, runs its work, and returns a tight summary. You only see the summaries — not the work that produced them. That is how the system stays cheap and fast even with a real team.

STEP 7

Add scope rules + safety

The single biggest risk with an agent team is an agent doing something you did not ask for — sending a message you would not have sent, deleting a file you needed, calling an API you did not authorize. The fix is writing scope rules into every charter and adding hard safety gates for high-blast-radius actions.

PASTE THIS INTO CLAUDE CODE

Audit every agent in `.claude/agents/`. For each one, do two things:

1. Read the system prompt and check that it explicitly states:
 - The exact scope of what the agent OWNS (positive scope)
 - The exact scope of what the agent DOES NOT OWN (negative scope)
 - When the agent must escalate back to me (e.g., any send-to-customer action, any deletion, any spend over \$X)

2. Read the tools list in the frontmatter and check that it follows the least-privilege principle. Audit agents should be Read-only. Drafting agents should NEVER auto-send. Only agents whose entire job is to send a message I approved should have send-message tools.

For any agent missing scope or with overly-broad tools, propose a fix. Do not modify anything yet. Just show me the audit and the proposed fixes. I will approve before any changes.

DEFAULT TO DRAFT-ONLY

Make every agent that sends messages, edits production data, or spends money draft-only by default. The agent prepares the action, hands it to you for review, and you click send. After 30-60 days of trust building, you can let specific agents auto-execute on specific narrow tasks. Earn the auto-pilot, do not assume it.

STARTER PROMPT LIBRARY

Three more dispatches to run today

Once your team is built, you will start asking it to do things you did not realize were possible. Three to start with.

PASTE THIS INTO CLAUDE CODE

Spawn my CRO. Pull every lead in my GHL that has gone cold (no inbound or outbound activity in 14+ days). For each one, write a one-sentence reason it probably went cold and a one-sentence reactivation hook tailored to that lead's original inquiry. Show me the table. I will pick which to send.

PASTE THIS INTO CLAUDE CODE

Spawn my CMO. Audit my last 30 days of content. Group posts by hook style and topic. Tell me the 1 hook style that is over-performing, the 1 topic I have been avoiding, and the 3 specific posts I should make in the next 7 days based on that data.

PASTE THIS INTO CLAUDE CODE

Spawn my CFO. Read every QB0 transaction from the last 30 days. Categorize anything uncategorized. Flag any vendor charge over \$100 that wasn't in last month's pattern. Show me the YTD P&L per business with what changed vs last month. Do not modify QB0. Read-only audit.

REAL RESULT

I run a 5-chief + 13-specialist team across my 3 businesses (vehicle wraps, meal prep, personal brand). Total cost: about \$1/month extra in API usage. They do morning audits, draft hot inbound replies, reconcile my accounting, audit my SEO, and surface what I am avoiding. The biggest unlock is not any single agent — it is that I no longer hold all 3 businesses in my head every day. The Chief of Staff agent does. I just ask "what should I focus on right now" and it tells me. Built every charter from a couch with zero coding experience.

That is the whole system.

Zero ad spend. Zero coding experience. The other playbooks
live at brycenwood.com/playbook, one download each.

Built by Brycen Wood · [@brycenwood.ai](https://twitter.com/brycenwood.ai) · brycenwood.com