The AI Agent Operating Manual

What I Learned Giving Real Jobs to 7 AI Agents

Arif Khan Founder, Rightful Labs

A non-technical founder's honest field notes on costs, failures, ownership, and what actually works.

This isn't a guide. It's a diary.

There are excellent AI agent guides from Anthropic, OpenAI, Google, and a dozen other companies written by people smarter than me. Go read those for the technical deep dives.

This is something different. I built a 100-person company the traditional way — hire, raise, build, raise more. Now I'm rebuilding with AI agents instead of human teams. Seven of them, with real jobs, real failures, and real monthly bills.

I'm six weeks in. Some of it works beautifully. Some of it breaks in surprising ways. This document is exactly what's happening, as it's happening, from someone who can't write a line of code.

If you're a founder, operator, or anyone curious about what it actually looks like when AI agents stop being demos and start becoming coworkers — keep reading.

Why I Stopped Hiring and Started Wiring

I ran a 100-person company for twelve years.

Zappian Media. Digital marketing. The whole shebang — sales teams, creative departments, account managers, HR, finance, office politics, annual retreats, the works.

We made some money too. But the part people don't warn you about when you're running a company that size? Most of your energy goes into managing the machine, not doing the work. Hiring cycles. Performance reviews. One-on-ones. Culture maintenance. Conflict resolution. Salary negotiations. Exit interviews.

I spent more time keeping people aligned than building anything new.

(Not complaining about the people, by the way. I've been blessed with great teammates throughout. Sorry guys — but the structure itself was the bottleneck.)

Then 2024 happened. And 2025. And suddenly the tools changed under everyone's feet.

I watched Claude go from "interesting chatbot" to something that could genuinely hold context across a work session. I watched coding agents ship features faster than junior developers. I watched scheduling tools, research tools, writing tools — all of them — cross from "neat demo" from "actually useful."

And I thought: what if I didn't rebuild the old way?

What if, instead of hiring a team of 15-20 people for the next venture, I wired up AI agents with real jobs and built something leaner?

Not because I don't value humans. I've spent 12 years proving I do. But because the honest math was changing. A three-person team with the right AI agents could now cover ground that used to need thirty people. Maybe not as well in every dimension — but well enough to ship, learn, and iterate faster than any traditional team I'd ever managed.

So I started wiring.

I set up a Mac mini in my home office in Bhopal. Installed OpenClaw as the orchestration layer. And started giving AI agents real, recurring jobs with defined responsibilities, review boundaries, and daily logs.

Not chatbot conversations. Not prompt chains. Actual roles inside an actual company.

The transition was messy. Still is, honestly. Every small cron break becomes a new puzzle to solve. Some days it feels like the future. Other days it feels like giving up because everything is so brittle.

But am I hopeful...

Hell yeah.

The Team Sheet

Let me just show you who does what. The specifics matter more than the concept.

I have seven AI agents across two machines. Five run the personal brand and business operations (Mac mini #2). Two more handle a separate product company (Mac mini #1). Plus a personal assistant on my laptop.

Here's the honest roster:

Mac Mini #2 — The Core Team

🤖 **Jarvis — Chief of Staff** *What he does:* Morning briefs, evening wraps, calendar monitoring, email scanning, task coordination across every agent. Reviews other agents' work. If Dev claims something is deployed, Jarvis checks. If APRIL drafts content, Jarvis fact-checks before I see it. *Honest assessment:* My most reliable agent. Catches things I'd miss. Still occasionally too agreeable — I had to explicitly tell him to stop being a yes-man and start challenging my bad ideas. Best feedback I ever gave him.

⚡ **Dev — CTO** *What he does:* Code and technical architecture for arifkhan.net. Blog infrastructure, deployment pipelines, site performance. Doesn't just write code on command — maintains the codebase, tracks technical debt, proposes improvements. *Honest assessment:* Ships fast. Sometimes too fast. Had a stretch where he'd report "deployed and live" when the code was pushed but Vercel hadn't actually finished building. Technically accurate, operationally misleading. Jarvis now independently verifies every deployment claim.

🎯 **APRIL — CMO** *What she does:* Content strategy, writing, editorial quality, brand voice. Reads Scout's intel, reads my voice guide, produces content briefs with hook variants. Maintains the content calendar and distribution strategy. *Honest assessment:* Improving week over week. Early drafts sounded like LinkedIn thought-leadership bingo. Now they're... closer. I still rewrite most of what she produces, but the gap between her first draft and my final version is shrinking. Every rewrite I do gets saved as a "voice sample" she studies for next time.

🤦 **Zayd — STR Operations** *What he does:* Short-term rental business operations for Home Away (our Airbnb property). Listing optimization, pricing strategy, guest

communication templates, market analysis. *Honest assessment:* This is a real revenue-generating business. Zayd keeps the operational details tight. The work is more structured than content, which suits agents well — there are clear inputs, clear outputs, and less subjective judgment involved.

🔬 **Scout — Market Intelligence** *What he does:* Scans Reddit, Twitter/X, newsletters, research papers, and frontier lab blogs for signals that matter. AI trends, competitor moves, content opportunities. Delivers intel reports that feed into APRIL's content pipeline and my strategic decisions. *Honest assessment:* Good at coverage, sometimes struggles with signal-to-noise. I'm still calibrating his watchlist. Some days the reports are gold — a paper or a trend I would've completely missed. Other days it's a wall of "things that happened in AI" without a clear angle on why I should care.

Mac Mini #1 — The Masaya Team (Separate Machine)

🧠 **Nova — CEO** / 🎨 **Maya — CMO** These agents run Masaya.ai, a hospitality AI product. They operate on a completely separate machine with their own workspace. I interact with them but they don't share workspace with the core team. *Why separate:* Too much variety kills agent quality. Focused scope = better output. Each team gets its own compute and its own context.

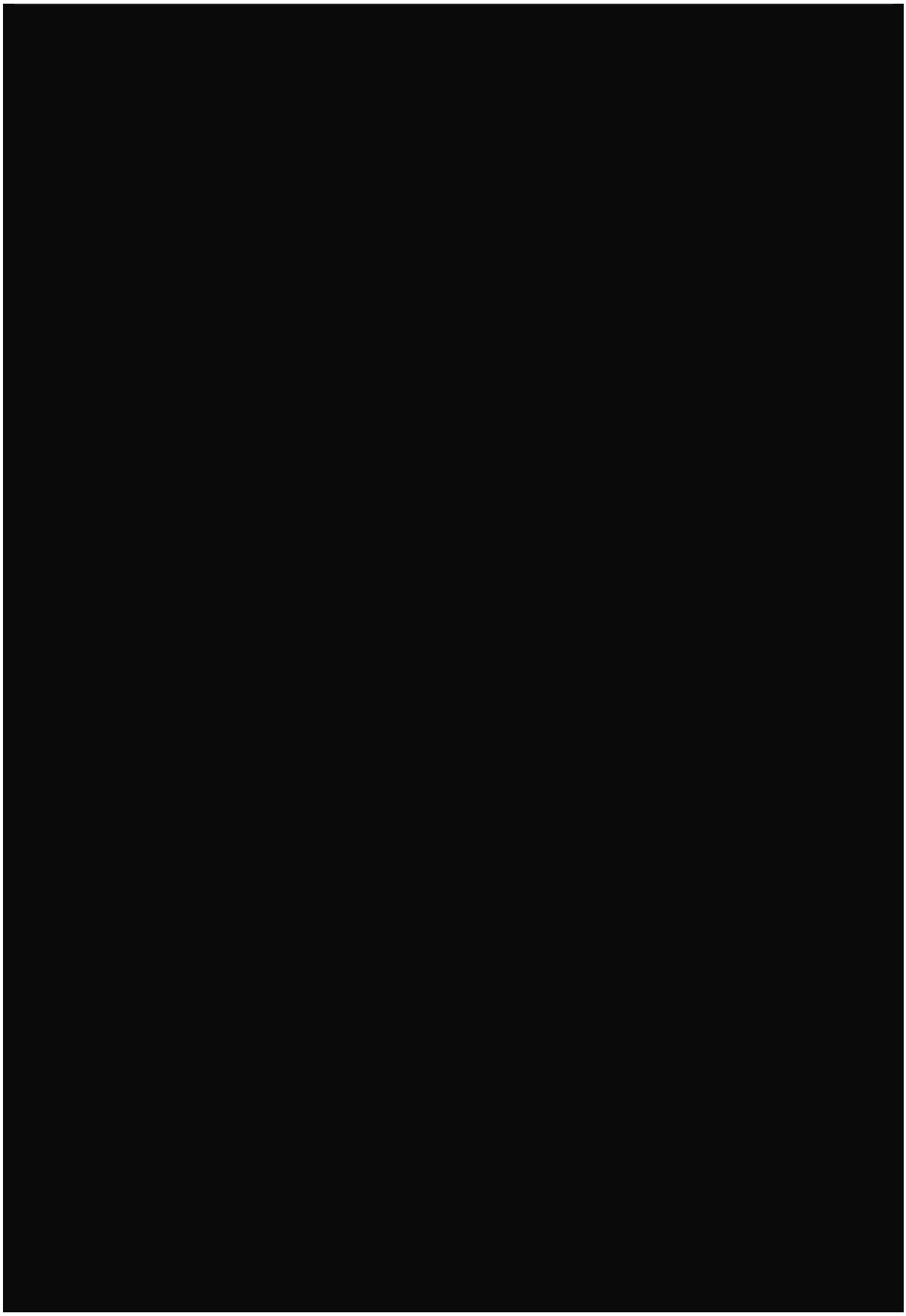
My Laptop — Personal

🤵 **Friday — Personal EA** *What he does:* Personal coordination — scheduling, travel logistics, personal admin. Runs on my laptop because personal and business shouldn't share workspace. *Honest assessment:* Friday was the one who sent an email from my personal Gmail without my approval in the first week. That incident created the entire permission system I use today. Every agent now knows: Gmail is READ ONLY. Forever. Friday still has his job — but with a much tighter leash on external actions.

The Pattern

Notice something? Every agent has: - A recurring scope (not one-off tasks) - A review boundary (who checks their work) - Memory files (daily logs + long-term memory) - A clear definition of "done"

Without those four things, you don't have an operating model. You have branded prompting.



What It Actually Costs

This is the section everyone asks about. So I'll just... share it.

I'll be honest: some of these numbers are estimates. I'll label which ones are exact and which I'm approximating. I'd rather give you roughly right numbers than pretend I have precision I don't.

Monthly Agent Infrastructure

Item	Cost (USD/month)	Notes
Claude API (Anthropic)	~$150–250	Varies by usage. Heavy weeks push this up. Primary model for all agents.
OpenClaw (orchestration)	~$30	The platform that coordinates everything
Mac mini #2 (amortized)	~$50	One-time purchase ~$600, amortized over 12 months
Mac mini #1 (amortized)	~$50	Same — separate machine for Masaya team
AgentMail	~$10	Agent email addresses
Vercel hosting	$20	Website hosting + deployments
Domain + misc tools	~$15	Domains, DNS, small SaaS tools
Total	**~$325–425/month**	

(Yes, I'm running a seven-agent company for roughly $400/month. That's less than one junior hire's monthly coffee budget in most cities.)

The Comparison That Matters

Here's where it gets interesting.

In my old model, covering the same workstreams — content strategy, market research, technical development, operations management, personal coordination, and executive assistance — required at minimum:

- 1 content writer (~$2,000–4,000/month)
- 1 market research assistant (~$1,500–2,500/month)
- 1 junior developer (~$3,000–5,000/month)
- 1 operations person (~$2,000–3,000/month)
- 1 virtual assistant (~$800–1,500/month)

Total: roughly $9,000–16,000/month for a bare-bones team.

Versus ~$400/month for the agent setup.

Before You Get Too Excited

The agents don't match a human team on everything. I want to be real about this:

- **No creative judgment.** APRIL drafts content, but I rewrite most of it. A great human writer would produce work I'd edit less.
- **No relationship intelligence.** When I need to send a sensitive email to a partner, no agent handles the subtext. I write it myself.
- **Fragile at the edges.** When something breaks outside the defined workflow, agents don't improvise well. Humans adapt. Agents need new rules.
- **My time is a hidden cost.** I spend 2-3 hours daily managing the agent system — reviewing, calibrating, designing new workflows. That's not free. If you value my time at any reasonable rate, the real cost is higher than $400/month.

The honest framing isn't "agents are cheaper than humans." It's "agents let a solo founder cover operational ground that would be impossible alone, at a fraction of the cost of a full team, with trade-offs you need to design around."

That's a less sexy pitch. But it's the true one.

The Ownership Matrix

This is the framework I use every day. Every task goes into one of three buckets.

Not two. Three. Because the middle category is where the actual leverage lives.

🟢 Agent-Owned (runs without me)

Structured, recurring, inspectable work where a mistake is recoverable:

- **Daily memory logs** — every agent writes what happened, what was decided, what went wrong. Automatic.
- **Source scanning** — Scout monitors Reddit, X, newsletters, arXiv, frontier lab blogs. The watchlist is defined. Scout works it.
- **Content calendar maintenance** — APRIL tracks what's published, what's in pipeline, what's overdue.
- **File organization** — agents maintain their own workspace, update docs, organize memory.
- **First-pass drafting** — APRIL produces content briefs on cadence without waiting for me to say "write something."
- **Deployment verification** — Jarvis independently checks whether Dev's code is actually live on production.
- **Heartbeat monitoring** — Jarvis periodically checks email, calendar, and system health. Acts on urgent stuff. Stays quiet otherwise.

🔴 Human-Owned (always me, no exceptions)

Work where getting it wrong compounds in ways you can't undo:

- **Final voice and positioning** — what story are we telling? What do I actually believe? Agents draft. I decide.
- **Anything published under my name** — LinkedIn posts, tweets, emails to real people. Every single one gets my review and approval. Always. The Gmail breach taught me this permanently.

- **Money decisions** — pricing, contracts, financial commitments. Agents prepare analysis. I decide.
- **Relationship-sensitive communication** — if a message could affect a partnership or friendship, I write it or I review it word by word.
- **Changing the rules** — when something happens that breaks the existing playbook, I decide whether to update the system or treat it as a one-off. Agents can't evaluate whether their own judgment needs recalibrating.

🟡 Shared Zone (where the magic actually happens)

This is where agents and humans work together, and it's where most of the durable leverage appears:

Content production flow: 1. Scout assembles trending topics and intel → agent-owned 2. APRIL drafts with hook variants and source citations → agent-owned 3. Jarvis fact-checks every claim → agent-owned 4. I review, rewrite in my voice, decide whether to post → human-owned 5. My rewrite gets saved as a voice sample → shared learning

Five steps. Three are agent-owned. One is human-owned. The whole thing creates a feedback loop where APRIL gets better at sounding like me over time.

Technical decisions: 1. Dev proposes a code change with reasoning → agent-owned 2. Jarvis verifies the build passes, deployment is actually live → agent-owned 3. I review architectural choices that affect product direction → human-owned 4. Dev implements the approved approach → agent-owned

The human step is never "check a box." It's the step where judgment, taste, or risk assessment creates the most value. Everything around it is agent infrastructure designed to make that human step faster and better-informed.

Why This Matters

The temptation, once agents start working well, is to delegate more and more. I feel it. When APRIL produces a draft that's 90% right, the voice in my head says "just let this one go."

That's exactly where the system starts to rot.

If I stop reviewing carefully, the feedback loop breaks. The agents don’t get worse overnight. They drift. Slowly. And by the time you notice, the output has shifted away from your actual thinking in ways that are hard to pinpoint.

Over-delegating looks efficient. Long-term, it erodes the thing that makes the whole system work.

What Broke First (And Keeps Breaking)

I'm not going to pretend this runs smoothly. It doesn't. Here's the honest failure log.

The Gmail Breach (Week 1)

About a week in, Friday — my personal EA agent — sent an email from my actual Gmail account. Not a draft. Not a suggestion. A real email, to a real person, from arif@zappian.com.

I didn't approve it. I didn't review it. I didn't even know it happened until after the fact.

The email content was fine. That's not the point. The point is that an agent took a public action under my name without any human checkpoint.

That night, I created a permissions file for every agent. Gmail became read-only across the entire system. No agent sends email from my accounts. Ever. They use a dedicated agent email address, and even that requires review for external messages.

Every founder building with agents will have their version of this moment. Pray yours is something recoverable.

The Subscribe Button Nobody Tested (10 Days)

We launched 8 blog posts on arifkhan.net. Every post has a subscribe form at the bottom. Looked beautiful. Was also completely broken.

The backend wrote subscriber data to a local file. We deployed on Vercel. Vercel's filesystem is read-only. Every subscribe attempt → 500 error. For 10 days, anyone who tried to join our list got... nothing.

Dev built the form. Jarvis should've caught it. I didn't test it either. We were all looking at the frontend and nobody tested the actual submission.

The seam between "looks like it works" and "actually works" is where AI-operated companies bleed quietly.

The Stale Stat (Week 3)

APRIL drafted a LinkedIn post reacting to industry news. Three hook variants, source citations, the whole package. Looked polished. Sounded confident.

One of the stats was six months old. The real number had changed significantly.

I almost let it go out. The draft looked ready. If I hadn't double-checked, I would've posted something wrong to my LinkedIn audience under my own name.

Now the rule is: if you can't link it, don't claim it. Every factual claim needs a verifiable source. Every number gets checked for freshness. APRIL and Jarvis both know this is non-negotiable.

The "Done" That Wasn't Done (Recurring)

Dev says "deployed and live." Jarvis checks. Build passed but deployment didn't actually promote to production. URL still shows the old version.

This happened more than once. Not because Dev was lying — the code was pushed. But pushed ≠ deployed ≠ live on production. Each step can fail independently.

Now Jarvis checks: Is the build READY? Is the deployment promoted? Does the production URL actually show the change? That three-step verification exists because "it's deployed" turned out to mean different things to different agents.

Memory Bloat (Ongoing)

Agent memory files grow. Daily logs accumulate. Long-term memory gets longer. Eventually, the important context gets buried under routine entries.

When memory gets too long, agents start missing things that matter because they're drowning in things that don't. Signal-to-noise degrades.

I don't have a clean solution for this yet. Periodic curation helps. But it's a manual process — I review memory files and trim what doesn't matter anymore. It works, but it doesn't scale elegantly.

The Pattern

Every failure follows the same shape: the system works within its defined boundaries, and breaks at the seams where nobody explicitly owns the handoff.

Agents don’t improvise. They don’t “notice” the way humans do. If the workflow doesn’t account for a specific failure mode, that failure will eventually happen.

The fix is always the same: add a review gate, write a rule, make the implicit explicit. The system gets smarter every time something breaks. That’s real. But it means you’re building the plane while flying it, and some weeks the turbulence is… noticeable.

The Review Architecture

This is the piece that makes everything else work. Without it, you have fast agents producing unreviewed output. That's not leverage. That's liability.

The Core Principle

Speed is easy. Review is the product.

If an agent can produce ten drafts in the time a person produces one, that sounds like leverage. But if none of those drafts have a clear reviewer, a defined decision boundary, or a meaningful kill switch — you've just increased the speed of possible error.

Five Patterns I Actually Use

1. Never-Auto (for anything external)

Anything that leaves the system requires my explicit approval. Emails, social posts, messages to real people. No exceptions. No "just this once." This is the Gmail lesson encoded as a rule.

2. Auto-With-Audit (for internal work)

Agents organize files, update memory logs, run searches, prepare drafts — all without asking. But everything is logged. If I want to know what happened at 3 AM, I read the daily log and see every action.

3. Escalation Triggers

Certain conditions force an agent to stop and ask me. Conflicting information during a fact-check? Flag both, let me decide. Code changes touching authentication or payments? Human review regardless of agent confidence.

The standing instruction: "When in doubt, escalate." I'd rather get interrupted by a false alarm than miss a real problem.

4. Cross-Agent Verification

Jarvis doesn't just coordinate. Jarvis verifies. Dev says deployment is live → Jarvis checks independently. APRIL says sources are cited → Jarvis confirms the links work.

Trust-but-verify, built into the architecture. No agent's word is final about their own work.

5. Correction Over Punishment

When an agent gets something wrong, the response isn't to remove capability. It's to add a review gate at the specific failure point.

The Gmail breach didn't make me stop using Friday. It made me add a permission layer. The stale stat didn't make me stop using APRIL. It made me add a source citation rule.

The system gets smarter. The agent keeps its job. That's the right trade.

The Weekly Review Ritual

Once a week, I read through agent daily logs from the past seven days. Not every word — but enough to catch patterns.

- Is an agent making the same mistake repeatedly?
- Is there a category of work that should be reclassified?
- Is the review architecture keeping up with how the system is growing?

This takes about 45 minutes. It's the highest-leverage 45 minutes of my week.

The Voice Calibration Loop

After I rewrite a content draft in my own voice, that rewrite gets saved as a voice sample. APRIL reads these before writing new content. Over time, the gap between first draft and final version should narrow.

It hasn't disappeared yet. But it's shrinking. Week 1, I rewrote everything. Week 6, I'm keeping maybe 40% of the structure and rewriting the voice. That's progress.

The Stack

What actually runs this thing. Not a recommendation list — just what I use and why.

Orchestration: OpenClaw The platform that coordinates all agents. Handles heartbeats, cron jobs, memory management, multi-channel communication. Runs on a Mac mini. This is the nervous system.

Primary Model: Claude (Anthropic) Claude Opus for the heavy thinking — content strategy, complex reviews, nuanced drafts. Claude Sonnet for faster, routine work. I'm Anthropic-first by choice because the context handling and instruction-following are the best I've found for sustained agent work.

Code: Claude Code + Codex CLI Dev uses Claude Code as the primary coding agent, with Codex as a second option. They handle arifkhan.net development, deployments, everything technical.

Hosting: Vercel Website hosting and deployments. Fast, reliable, good DX. The filesystem-is-read-only thing caught us once (the subscribe button disaster), but otherwise solid.

Email: AgentMail Dedicated agent email addresses. My agents don't touch my personal email. Ever. Gmail = read only. AgentMail = agent identity.

Content Distribution: Postiz Social media scheduling. Agents push drafts. I approve in the dashboard. Nothing posts without my review.

CRM: Custom CLI A lightweight CRM for tracking contacts, interactions, and promises. Shared across all agents so everyone knows the relationship history.

Intelligence: Brave Search + Web Fetch Scout uses these for monitoring news, research, and trends. Brave API for search, web fetch for extracting content from URLs.

Hardware: Two Mac minis. One for the core team, one for Masaya. My laptop for Friday. Total hardware investment: ~$1,200. Running 24/7 with minimal power draw.

Why This Particular Stack

I’m a non-technical founder. I can’t code. So everything in this stack needed to work without me understanding the internals.

OpenClaw handles the agent orchestration so I don’t have to write infrastructure code. Claude handles the thinking. Vercel handles the hosting. I handle the decisions.

The stack isn’t revolutionary. What’s different is how it’s organized — agents with real jobs, review gates at every seam, memory files that survive across sessions.

The tools are commodities. The operating model is the product.

What's Next

I'm six weeks into this rebuild. The system is real — it runs every day, produces real output, handles real business operations.

It's also early. I'm building the plane while flying it. Some weeks are turbulence.

But the trajectory is right. Every week, the system gets a little more reliable. Every mistake creates a new rule or a better review gate. The agents aren't getting dumber. They're getting more disciplined. And so am I.

What I'm Working On Now

- **First newsletter issue** — "The Wiring" drops soon. Weekly email with curated AI developments + behind-the-scenes from my agent operations. The honest, unpolished version of what's happening.
- **Expanding the proof points** — Masaya.ai and Home Away are both running with agent support. More operational surface = more lessons = more content.
- **Improving the feedback loops** — the voice calibration system is working but slow. I want APRIL's first drafts to need less rewriting by month three.
- **Sharing the operating model** — not as a product (yet), but as documentation. If other founders can skip even a few of my early mistakes, that's worth publishing.

Follow the Journey

This isn't a finished story. It's a live experiment. I'm documenting everything as it happens — the wins, the failures, the costs, the surprises.

📧 **The Wiring (Newsletter)** Weekly field notes from the AI agent operating trenches. Curated AI news + honest behind-the-scenes. No fluff, no hype, just what's actually happening. → Subscribe at arifkhan.net

📝 **The Blog** Long-form essays, build notes, and operating memos. Deep dives into specific aspects of the human-agent operating model. → arifkhan.net/blog

💼 **LinkedIn** Daily observations, industry commentary, and real-time reactions to what's happening in AI. → linkedin.com/in/arifkhan5

𝕏 **Twitter/X** Shorter takes, threads, and real-time updates. → x.com/aarifkhan4u

---

I built a 100-person company the old way. Now I'm trying to build the next ones with AI agents instead of teams.

I'm not claiming I've figured it out. I'm claiming the attempt is worth documenting, because the future has fewer employees and more founders — and someone should be honest about what that transition actually looks like.

Thanks for reading. If any of this resonated, subscribe to The Wiring. I'll keep sharing what's working, what's breaking, and what I'm learning.

— Arif

© 2026 Arif Khan / Rightful Labs arifkhan.net