

OPENCLAW

WHAT PEOPLE ARE
ACTUALLY DOING
WITH IT



FORWARD**FUTURE**

OpenClaw: What People Are Actually Doing With It

Real automations. Real results. No BS.

Let's Get Real

OpenClaw isn't your typical chatbot.

Those AI assistants that just sit in one window? OpenClaw isn't that. It's in your email, your calendar, your Slack, your dev environment. Everywhere.

And people are doing wild things with it.

This guide shows you exactly what they're building. Not theory. Not "you could maybe..." — this is what's working right now.

Every tutorial is conversational. You talk to OpenClaw and it builds what you need.

Ready? Let's go.

Table of Contents

[Running Your Whole Business](#)

[Coding From Anywhere](#)

[Content & Marketing](#)

[Personal Productivity](#)

[Communication Management](#)

[Home Automation](#)

[Research & Analysis](#)

[Health & Fitness](#)

[Shopping & Errands](#)

[Finance & Accounting](#)

[Travel & Transport](#)

[Learning & Knowledge](#)

[Job Search & Career](#)

[Infrastructure](#)

[Security](#)

Running Your Whole Business

The Full Stack

People are running entire businesses through OpenClaw.

Not exaggerating.

What they're automating: - Lead qualification and management - Email across multiple accounts - Daily exec briefings - Sales pipeline management - Social media posting and engagement - Ad campaigns

Why this matters: Your business runs 24/7. While you sleep, your agent works.

Want to try it? You need: - Gmail or Google Workspace - Asana account (free works) - Asana API token - Telegram or Slack - HubSpot for CRM (optional)

Let's build it:

Step 1: Hook up email

Say: "Help me set up Gmail integration. I want to connect my email account your-email@gmail.com"

OpenClaw walks you through it.

Step 2: Morning briefing

Say: “Create a cron job that runs every weekday at 7am. It should check my email, pull my calendar events for today, and send me a morning briefing via Telegram”

Done.

Step 3: Add Asana

“Add my Asana Personal Access Token to the .env file”

Give it your token when it asks.

Then: “Install the HubSpot skill for CRM”

Step 4: Document everything

“Add my Asana workspace and project IDs to TOOLS.md so you can reference them”

This helps OpenClaw remember your setup.

CEO Dashboard Mode

Multi-agent dashboards for serious oversight.

One user: “I setup a multiagent dashboard with OpenClaw. Not for coding. For CEO work. It’s unlocking a part of my brain I didn’t know existed.”

That’s the vibe.

You need: - Asana workspace and project IDs - Asana API token - Google Sheets - Slack (optional)

Build it:

Step 1: Get the Asana skill

“Check if I have the Asana skill installed. If not, install it”

Step 2: Configure workspace

“Add my Asana workspace details to TOOLS.md. My workspace is ‘company-name’ with ID 1234567890. I have two projects: Strategy and Operations”

Step 3: Create queries

“Show me all high-priority tasks across my Strategy and Operations projects”

Or automate it: “Set up a daily summary at 9am showing my top priority tasks”

Step 4: Weekly reports

“Every Monday at 9am, generate a weekly summary of completed tasks from Asana and send it to our Slack channel”

The Autonomous Business Manager

This is where it gets crazy.

Agents that code. Deploy. Manage everything.

Real example: Someone created an agent called “John Wick” to help them hit \$20K MRR. The agent built its own team to make it happen.

Seriously.

You need: - Telegram or Slack - GitHub account (optional) - Clear business goals

Here's how:

Step 1: Spawn sub-agents

“Spawn a sub-agent to research our top 5 competitors and compile detailed findings into a report”

OpenClaw creates and manages it.

Step 2: Document goals

“Add to TOOLS.md that my business goal is \$20K MRR within 6 months. Break this down into quarterly milestones”

Step 3: Weekly reviews

“Create a weekly review where you check progress on all my business goals, identify blockers, and suggest next steps”

Step 4: Stay informed

“Whenever a sub-agent completes a task, send me a notification via Telegram with a summary”

Coding From Anywhere

Mobile Development

Build features from your phone.

One dev: “Parked in the car, chatting with my AI agent through OpenClaw via Telegram flawlessly 1-shotting an app feature. Sending me a screenshot of the result is my new workflow.”

The future is here.

You need: - Telegram account and bot token - Cursor installed on your dev machine - GitHub account

Let's set it up:

Step 1: Telegram

“Help me set up Telegram so I can message you from my phone”

Follow the steps.

Step 2: Cursor agent

“Install the cursor-agent skill so you can write code for me”

Step 3: Build from mobile

From your phone: “Build a login form component with email and password validation. Use React and Tailwind CSS”

OpenClaw writes it. Sends you screenshots.

Step 4: Review and ship

“Show me the code you wrote”

“Run the tests”

“If everything looks good, commit this to GitHub and deploy to production”

Done from your phone. While waiting for coffee.

The Full Dev Pipeline

End-to-end automation. Zero friction.

The flow: 1. Monitor site performance 2. Auto-create tickets 3. Implement fixes with coding agents 4. Deploy changes 5. Notify team

All automatic.

You need: - GitHub account and token - Sentry for monitoring (optional) - Deployment platform like Fly.io or DigitalOcean - Jira or just use GitHub Issues

Build it:

Step 1: Install skills

“Install the GitHub skill”

“Install the Sentry skill if available”

Step 2: Monitoring

“Every hour, check the metrics at example.com/metrics. If the bounce rate is over 50%, create a GitHub issue labeled ‘auto-fix’”

Step 3: Auto-fix workflow

“Create a skill that watches for GitHub issues labeled ‘auto-fix’. When you see one: analyze the issue, spawn a coding agent to implement a fix, create a PR, run tests, and if tests pass, merge and deploy. Then notify the team in Slack”

Step 4: Deployment

“When I say ‘deploy to production’, run these commands: [your deployment steps]”

Or: “Help me set up automated deployment to Fly.io”

Background Tasks That Actually Work

Persistent automation with notifications.

Real example: Oracle server migration script runs every 5 minutes, pings Telegram when successful.

You need: - VPS or always-on machine - Telegram bot token - SSH access to servers

The setup:

Step 1: Create the script

“Write a script that checks if the Oracle server at oracle.example.com is available and responding”

Step 2: Cron it

“Create a cron job that runs this Oracle check script every 5 minutes”

Step 3: Notifications

“When the Oracle server becomes available, send me a Telegram message saying ‘Server migration complete!’”

Step 4: Monitor

“Show me all my active background jobs”

“What’s the status of the Oracle migration checker?”

Content & Marketing

Video Production Automation

From idea to production pipeline.

The workflow: 1. Monitor Slack for “potential video idea” 2. Search X/Twitter for trends 3. Create Asana card with research 4. Generate storyboards 5. Automate production

You need: - Slack workspace with API token - X/Twitter API via Grok/xAI key - Asana with Video Pipeline project - WhatsApp for storyboards (optional)

Let's build:

Step 1: Slack integration

“Install the Slack skill and help me connect my workspace”

Step 2: The workflow

“Monitor our Slack channel #content-ideas. Whenever someone says ‘potential video idea’, extract the topic, search X/Twitter for related discussions, collect the top 10 relevant tweets, and create an Asana card in my Video Pipeline project with a summary, the tweets, and a suggested outline”

Step 3: X/Twitter search

“Add my xAI API key to .env so you can search X/Twitter”

Step 4: Configure Asana

“My Video Pipeline project ID is [ID]. Save this to TOOLS.md”

Step 5: Test it

Post in Slack: “potential video idea: how to automate video production with AI”

Watch it work.

YouTube Analytics

Track everything automatically.

Real deployment: 922 videos tracked, 17,915 analytics rows, 30 days of data.

You need: - YouTube channel - Google Cloud project with YouTube Analytics API - OAuth credentials - Python 3

Build it:

Step 1: API setup

“Help me set up YouTube Analytics API access. I want to track performance data for my channel”

Step 2: Database

“Create a database to track my YouTube videos. Store video ID, title, published date, and daily stats for views, likes, and comments”

Step 3: Collection script

“Write a Python script that fetches analytics data for all my videos and stores it in the database”

Step 4: Daily automation

“Every day at 6am, update my YouTube analytics data”

Step 5: Query interface

“Show me my top 10 performing videos this month”

“What’s my total view count for the past week?”

“Which videos have the highest engagement rate?”

Brand Voice Consistency

Keep your voice unified across platforms.

You need: - Sample content (50+ tweets, posts, articles) - X/Twitter API (optional)

Here’s how:

Step 1: Collect samples

“Search my X/Twitter profile and export my last 100 tweets”

Or: “Here are 50 of my blog posts and tweets [paste content]”

Step 2: Analyze

“Analyze these writing samples and extract my writing style. Create a style guide covering tone, vocabulary, sentence structure, and common phrases”

Step 3: Save it

“Save this style guide to ~/clawd/reference/brand-voice.md”

Step 4: Use it

“Review this draft tweet for brand voice consistency: [paste draft]”

“Rewrite this paragraph to match my writing style: [paste paragraph]”

Personal Productivity

Morning Briefings

Automated intelligence every morning.

Weather, calendar, health metrics, social trends. All in one place.

Users save 30+ minutes every morning.

You need: - Google Calendar - Gmail - Telegram or Slack - Weather skill

Build it:

Step 1: Weather skill

“Install the weather skill”

Step 2: Briefing workflow

“Create a skill called ‘morning-briefing’ that generates a daily summary with: weather for San Francisco, my calendar events for today and tomorrow, any urgent emails from the last 12 hours, and top 3 trending topics in AI and tech”

Step 3: Daily cron

“Create a cron job that runs my morning briefing every weekday at 7am and sends it to me via Telegram”

Step 4: Customize

“Update TOOLS.md with my morning briefing preferences: location San Francisco, check work and personal calendars, trending topics in AI, tech, and startups”

Calendar Optimization

Auto time-blocking with conflict resolution.

Example: Startup founder delegated complete calendar management.

Zero manual work.

You need: - Google Calendar - Task management like Todoist, Asana, or Notion - Calendar permissions

The setup:

Step 1: Calendar access

“Connect to my Google Calendar with read and write permissions”

Step 2: Priority system

“Help me tag my Todoist tasks with priority levels: high, medium, low”

Step 3: Auto-scheduler

“Every Monday at 6am, pull all my tasks from Todoist, calculate how much time each needs, and block time on my calendar based on priority. High priority tasks get morning slots. Leave 2-hour buffer blocks open for meetings. Never schedule over events marked ‘client meeting’”

Step 4: Conflict resolution

“If you detect a calendar conflict, automatically move the lower priority task to a different time slot and notify me”

Weekly Reviews

Extract decisions from meetings. Generate briefings.

You need: - Meeting transcripts or Notion workspace - Slack access
(optional) - GitHub (optional)

Build it:

Step 1: Review cron

“Every Friday at 5pm, create my weekly review”

Step 2: Pull sources

“For the weekly review, pull from: my Notion pages tagged ‘meeting notes’, Slack threads I participated in, and GitHub PRs I merged this week”

Step 3: Generate report

“The weekly review should summarize: key decisions made, action items for next week, project status updates, and any blockers. Send it to me via email”

Step 4: Add context

“For my weekly 1-on-1s, also include: what we discussed last week and follow-up items”

Communication Management

Multi-Account Email

Query across all your accounts at once.

Sales teams save 2+ hours daily with this.

You need: - Multiple Gmail accounts - OAuth credentials for each

Let's do it:

Step 1: Connect accounts

"Help me connect three email accounts: personal@gmail.com, work@company.com, and youtube@gmail.com"

Step 2: Document

"Add these to TOOLS.md: personal@gmail.com is for personal stuff, work@company.com is for work, youtube@gmail.com is for YouTube"

Step 3: Search all

"Search all my email accounts for messages about 'blinds installer' from the last 3 months"

"Find any unread emails across all accounts from today"

Step 4: Auto-triage

"Every hour, check all my inboxes. Archive obvious spam, flag urgent emails from my boss or clients, and send me a Telegram message with a summary of new important emails"

Voice Message Processing

Transcribe and extract action items.

You need: - Whisper.cpp or similar - Telegram, WhatsApp, or Signal

Setup:

“When I send you a voice message via Telegram, automatically transcribe it using Whisper, extract the key points and action items, and send me back a formatted text summary”

Test it. Send a voice message. Get back formatted text with action items.

Home Automation

Smart Home Control

Lights. Thermostat. Coffee maker. All from your phone.

One user: “A new era of home automation... LLM for OpenClaw and Grok 4.1”

You need: - Smart home devices - Home Assistant or device API keys - Telegram or WhatsApp

Setup:

Step 1: Install skills

“Search for Home Assistant skills and install one if available”

Step 2: Configure devices

“Add my device names to [TOOLS.md](#): living room lights, coffee maker, thermostat”

Step 3: Automation rules

“When I say ‘goodnight’, turn off all my lights”

“Every morning at 7am, turn on the coffee maker”

Step 4: Voice control

Via Telegram: “Turn on living room lights”

Via WhatsApp: “Set thermostat to 72 degrees”

Tesla & EV Control

Check your charge level. Precondition the cabin. Flash the headlights to find it in a parking lot. All from Telegram.

You need: - Tesla account - Tesla API token - Telegram or WhatsApp

Setup:

Step 1: Install the skill

“Search for Tesla skills and install one if available”

Step 2: Authenticate

“Add my Tesla API token to .env”

Step 3: Control it

“What’s my Tesla’s current charge level?”

“Set the cabin temperature to 70 degrees and start preconditioning”

“Open the trunk”

Step 4: Automate it

“Every weekday at 6:30am, precondition my car and send me the charge level via Telegram”

Energy Monitoring

Track your power usage. Get alerts when something spikes. Know exactly what's costing you money.

You need: - Smart meter or energy monitor API - Home Assistant (optional)
- Telegram

Setup:

Step 1: Connect your data

“Add my energy monitor API key to .env”

Step 2: Set alerts

“If my energy usage goes over 5kW, send me a Telegram alert immediately”

“Send me a daily energy summary at 9pm with total kWh and estimated cost”

DNS & Network Filtering

Block ads and trackers across your whole network. Manage your AdGuard Home rules by talking to it.

You need: - AdGuard Home instance - API access

Setup:

Step 1: Connect

“Add my AdGuard Home URL and credentials to .env”

Step 2: Manage rules

“Block all TikTok domains on the kids’ devices”

“Show me the top 10 most blocked domains this week”

“Temporarily disable filtering for my laptop for the next hour”

Robot Vacuum Control

Tell your Roborock to clean the kitchen. From your couch. Via WhatsApp.

You need: - Roborock or similar robot vacuum - Device API or Home Assistant

Setup:

Step 1: Install plugin

“Search for Roborock vacuum skills”

Step 2: Use it

“Vacuum the kitchen”

“Send the robot back to its dock”

“Every Tuesday and Friday at 10am, vacuum the whole house”

Recycling Reminders

Never miss collection day.

Example: Saturday morning reminders with calculated collection type (paper vs. container weeks).


You need: - Local recycling schedule - Telegram

Here's how:

Step 1: Document schedule

“Add this to my reference files: Mill Valley recycling alternates weekly. Paper week is mixed paper and cardboard. Container week is bottles, cans, plastic. The week of Jan 25, 2026 was a paper week. Calculate from there: even weeks from that reference are paper, odd weeks are container”

Step 2: Create reminder

“Every Saturday at 10am, calculate which recycling week it is based on my reference, and send me a Telegram reminder: ‘ Recycling reminder: [paper or container] week”

Step 3: Test

“What type of recycling week is this Saturday?”

Research & Analysis

Market Research

Pre-launch validation. Keyword research. Competitor analysis.

You need: - Brave Search API key OR Exa API key - Data4SEO API (optional)

Build it:

Step 1: API setup

“Add my Brave Search API key to .env”

Step 2: Research workflow

“Research the market for ‘AI automation tools’. Find competitors, analyze their pricing models, identify market gaps, and compile everything into a markdown report”

Step 3: Save and review

“Save the research report to ~/clawd/research/ai-automation-market.md”

“Create an Asana task with the key findings”

YouTube Competitor Tracking

Track competing channels. Upload frequency. Topics. Engagement.

You need: - YouTube Data API - OAuth credentials - Competitor channel IDs

Setup:

Step 1: Competitor list

“Add these YouTube competitors to TOOLS.md: @channel1, @channel2, @channel3”

Step 2: Tracking script

“Write a Python script that fetches recent upload data for my competitor channels: titles, view counts, like counts, and publish dates”

Step 3: Weekly analysis

“Every Monday, analyze my competitor channels and send me a summary: how many videos they uploaded this week, what topics they covered, and which videos performed best”

Health & Fitness

Wearable Health Tracking

Your Oura Ring data meets your calendar meets your AI. Personal health assistant that actually knows your body.

You need: - Oura Ring (or similar wearable with API) - Google Calendar - Telegram

Build it:

Step 1: Connect your ring

“Add my Oura Ring API token to .env”

Step 2: Daily health brief

“Every morning at 7am, pull my Oura data and send me a health summary: sleep score, readiness, HRV, and whether I should push hard or take it easy today”

Step 3: Smart scheduling

“If my readiness score is below 70, reschedule my gym session to tomorrow and block the slot for recovery”

Shopping & Errands

Grocery Autopilot

Weekly meal plan to grocery order. Delivery slot booked. No APIs needed — just browser automation.

Real community build: Tesco Shop Autopilot. Meal plan, add regulars, book delivery, confirm order. All hands-free.

You need: - Browser automation (desktop-control skill) - Grocery delivery account

Set it up:

Step 1: Install browser control

“Install the desktop-control skill so you can use my browser”

Step 2: Define your order

“Here’s my weekly meal plan. Go to Instacart, add the ingredients I need, pick the earliest delivery slot, and confirm the order”

Step 3: Automate it

“Every Sunday at 9am, place my regular grocery order and send me a confirmation via Telegram”

Price Tracking

Monitor prices across Amazon, eBay, Walmart. Get alerts when something drops. Find arbitrage opportunities.

You need: - Price tracker skill from ClawHub - Telegram

Setup:

Step 1: Install skill

“Install the price-tracker skill”

Step 2: Track items

“Track the price of this Sony WH-1000XM5 on Amazon and alert me if it drops below \$250”

“Compare prices for ‘MacBook Pro M4’ across Amazon, Best Buy, and Walmart”

Finance & Accounting

Accounting Intake

Collect invoices from email. Organize receipts. Prep everything for your accountant. Monthly bookkeeping on autopilot.

You need: - Gmail integration - PDF text extractor skill - File storage

Build it:

Step 1: Email scanning

“Every week, scan my email for invoices and receipts. Download the PDFs and save them to ~/accounting/incoming/”

Step 2: Extract and categorize

“Extract the vendor name, amount, date, and category from each invoice. Create a monthly summary spreadsheet”

Step 3: Monthly prep

“On the 1st of every month, compile last month’s invoices into a package and email the summary to my accountant”

Stock & Market Analysis

Real-time quotes. Chart analysis via TradingView. Market summaries via Telegram. No Bloomberg terminal required.

You need: - Stock quotes skill (Quodd) or browser automation - TradingView account (optional)

Setup:

Step 1: Get quotes

“What’s the current price of AAPL, TSLA, and NVDA?”

Step 2: Chart analysis

“Log into TradingView, pull up the daily chart for TSLA, screenshot it, and give me a technical analysis”

Step 3: Daily digest

“Every trading day at 4:30pm, send me a market summary: how my watchlist performed, any big movers, and notable news”

Travel & Transport

Travel Planning

Multi-destination itineraries. Family logistics. Cost optimization. All from one conversation.

You need: - Travel manager skill from ClawHub - Brave Search API (for flight/hotel research)

Build it:

Step 1: Install the skill

“Install the travel-manager skill”

Step 2: Plan a trip

“Plan a 10-day trip to Japan for two people in April. Budget \$5K total. Find flights from LAX, suggest an itinerary covering Tokyo, Kyoto, and Osaka, and recommend hotels near train stations”

Step 3: Real-time transit

“When’s the next train from Shibuya to Shinjuku?”

Community members have built transit skills for specific cities — Vienna’s Wiener Linien skill gives real-time departures, disruptions, and routing.

Learning & Knowledge

Language Learning

Practice Chinese pronunciation via voice messages. Get real-time corrections. Study flows built into your chat app.

Community build: xuezh — a full Chinese learning engine with pronunciation feedback via OpenClaw.

You need: - Whisper.cpp or similar for transcription - Telegram (voice messages)

Try it:

“I want to practice Mandarin. Give me a conversation prompt, I’ll respond with a voice message, and you correct my pronunciation and grammar”

“Quiz me on the 50 most common Spanish verbs. Send one every hour via Telegram”

Knowledge Vault

Ingest your entire WhatsApp history. Transcribe voice notes.

Cross-reference with your other data. Build a searchable second brain.

Community build: Someone ingested a full WhatsApp export — 1,000+ voice notes transcribed, cross-checked with git logs, output as linked markdown reports.

You need: - Chat exports (WhatsApp, Telegram) - Whisper for transcription
- Semantic search (Qdrant + embeddings, optional)

The idea:

“Here’s my exported WhatsApp chat history. Transcribe all voice notes, extract every decision, action item, and important link, and create a searchable markdown knowledge base”

“What did I discuss with Sarah about the product launch last month?”

Job Search & Career

Automated Job Hunting

Search listings. Match against your resume. Get relevant opportunities delivered daily. Built in 30 minutes.

You need: - JSearch API key (or Brave Search) - Your resume/CV as a reference file

Build it:

Step 1: Add your resume

“Save this as my resume: [paste CV]. Add my JSearch API key to .env”

Step 2: Daily search

“Every morning, search for remote senior developer jobs in AI/ML. Match them against my resume and send me the top 5 most relevant via Telegram with links and match scores”

Infrastructure

Hardware Options

Run OpenClaw anywhere.

You need: - VPS like DigitalOcean, Hetzner, or Fly.io with 2GB RAM minimum OR - Raspberry Pi 4 with 4GB+ OR - Dedicated machine - Cloudflare for Tunnel OR Tailscale

Deploy:

Step 1: Choose infrastructure

“Help me deploy OpenClaw to a DigitalOcean VPS. I want a \$6/month droplet with 2GB RAM”

Step 2: Remote access

“Help me set up a Cloudflare Tunnel so I can access OpenClaw remotely without exposing it to the internet”

Step 3: 24/7 operation

“Set up OpenClaw to run as a system service that starts automatically on boot”

Always-On Agents

24/7 operation with notification-based oversight.

You need: - Always-on infrastructure - Telegram bot token - Process management like systemd or pm2

Setup:

Step 1: Heartbeat monitoring

“Edit HEARTBEAT.md and add: every 6 hours check my email for urgent messages, check my calendar for events in the next 2 hours, and check system health. If anything urgent, send a Telegram alert. Otherwise just reply HEARTBEAT_OK”

Step 2: Notification routing

“For notifications: critical issues send to Telegram immediately, important stuff goes in a daily summary, general info goes in a weekly digest”

Step 3: Auto-restart

“Monitor that OpenClaw is running. If it crashes, automatically restart it and alert me if it’s been down for more than 5 minutes”

Security

Mandatory Stuff

Don't skip this.

You need: - Cloudflare OR Tailscale - Config file access - Strong random secret generator

Do this now:

Step 1: Check exposure

“Check my config file. Make sure OpenClaw is only listening on localhost (127.0.0.1), not exposed to the internet”

Step 2: Secure access

“Help me set up a Cloudflare Tunnel for secure remote access”

Step 3: Authentication

“Enable authentication on the gateway. Generate a secure random token and add it to the config”

Step 4: File permissions

“Check the permissions on my .env file. It should only be readable by me (600)”

Step 5: Auto-updates

“Create a weekly cron job that checks for OpenClaw updates and notifies me when a new version is available”

Your First Automation

Start with a morning briefing.

It's simple, useful, and teaches you the basics.

You need: - OpenClaw installed - Telegram - Weather skill

Let's do it:

Step 1: Telegram

“Help me set up Telegram so I can message you from my phone”

Step 2: Weather

“Install the weather skill”

Step 3: First cron job

“Create a cron job that runs every weekday at 7am. It should get the weather for San Francisco and send it to me via Telegram”

Step 4: Test

“Run that cron job now so I can see what it looks like”

Step 5: Expand

“Add my calendar events to the morning briefing”

“Also check for urgent emails”

“Include trending topics in AI and tech”

Resources

- Docs: <https://docs.openclaw.ai>
- GitHub: <https://github.com/openclaw/openclaw>
- Community: Discord server
- Skills: <https://clawdhub.com> (1,700+ skills)
- Simplified Setup: <https://www.simpleclaw.com>

The Bottom Line

OpenClaw is different.

It's not about chatting with AI. It's about AI actually doing the work.

Every tutorial here uses natural language. No coding required. No complex commands to memorize.

Start with one automation. Master it. Then expand.

The question isn't whether AI agents can improve your workflows.

It's which workflow you'll automate first.

Everything here is real. Real deployments. Real users. Real results.

Compiled from OpenClaw community builds, X/Twitter discussions (Feb 2-4, 2026), GitHub repos, and production workflows.