Hermes Agent: From Zero to Your Own Personal AI Assistant

This guide summarizes the key concepts, models, and steps from the video.

By: Nate Herk

What is Hermes Agent

Hermes Agent is an open source AI agent from **Noose Research**, MIT licensed and one of the fastest growing open source projects on GitHub. It runs on your own infrastructure and grows with you over time through a self-improving loop with skills and memory.

Where It Runs

- Mac Mini, laptop, or VPS (virtual private server).
- Inside a Docker container.
- On Android via Termux.

Messaging Platforms

Hermes can run through Telegram, Discord, Slack, WhatsApp, or even iMessage. This guide focuses on Telegram setup.

What It Comes With Out of the Box

- 91 built-in skills installed by default.
- Access to a skills hub with over 520 community skills, including 16 official Anthropic skills.
- Vision, browser automation, image generation, text-to-speech, terminal commands, task planning.
- Voice note responses (Hermes can reply with audio, not just text).

Mindset Shift

If you are confused about anything Hermes-related, ask Hermes. It can read its own documentation and figure out how to install or use new things. Drop in an X post link or doc URL and tell it to implement what you want.

Hermes vs Claude Code vs OpenCode

Each tool has a clear role in the workflow. Hermes does not replace **Claude Code**, it complements it.

Claude Code

- Anthropic's coding assistant, lives in your terminal next to your code.
- Used when sitting down at the desk or laptop doing knowledge work.
- Nate's daily driver for 90% of knowledge work.

OpenCode

- Open source, created by Peter Steinberger (now at OpenAI), 350,000+ GitHub stars.
- NVIDIA built Nemo Code on top of OpenCode as a separate enterprise stack.
- Best for on-the-go work and quick crons through Telegram.

Hermes

- Lighter, faster, focused on self-improvement.
- Built for tinkering with open source models like Qwen and LLaMA.
- More stable than OpenCode (fewer breaking updates in Nate's experience).
- Best for on-the-go automation, scheduled crons, and Telegram-based work.

How to Use Them Together

All these agents work in a directory (file structure) that syncs to GitHub. Keep one GitHub repo with your business context and skills, then point any agent (Claude Code, Hermes, **OpenCode**, Codex) at that repo. Each agent understands its own naming convention (claude.md vs **agents.md**), so just ask it to adapt the repo.

The Five Pillars of Hermes

These are the core concepts that make Hermes work. Understanding them upfront makes the setup process click much faster.

Pillar 1: Memory

Memory is the small, durable context Hermes carries across sessions. Two main files load at session start:

- user.md: who you are, your style, your preferences, what you don't like.
- memory.md: environments, projects you're working on, business context.

AI agents wake up stateless (think of the movie Memento). Your job is to make sure context files are holistic so the agent doesn't repeat itself. Hermes will automatically update these files as you work, but you should still nudge it: "Chuck that in memory" or "Don't ever do this again." Beginner nuance: save durable preferences and facts to memory, use session search for old conversations (Hermes stores sessions in a SQLite database), do not store secrets or temporary task status in memory.

Pillar 2: Skills

Skills are procedural memory, reusable playbooks for how to do a task well. Think of a skill as a recipe for chocolate chip pancakes. Following the recipe gets you consistent results every time. Going off memory gets inconsistent pancakes.

- All skills live in a skill.md file with YAML front matter at the top.
- YAML front matter tells the agent what the skill does and when to use it.
- This is called progressive disclosure: full skill content only loads when needed, preventing context bloat.
- Hermes can analyze your workflow and turn repeated tasks into skills automatically.
- Skills update with feedback as you use them.

Memory equals what to remember. Skills equal how to do it again.

Pillar 3: Soul

The **soul.md** file shapes the personality of the assistant. Six different Hermes agents can each have a different vibe: concise, sarcastic, formal, whatever you want. **Soul** also evolves over time based on feedback. The soul file is just markdown, no YAML front matter required.

Pillar 4: Crons

Crons turn Hermes from reactive into proactive scheduled automation while keeping the full agentic loop. Just say in natural language: "Every morning at 6 AM do X, Y, Z." Hermes creates the cron and runs a fresh isolated session at that time, then sends results back to the original chat.

- CONTEXTFROM: pass one job's output into another.
- WORKDIR: run tools from a specific project folder.
- NOAGENT flag: run a script without the agentic harness loop (just the script, no agent reasoning).
- Cron sessions cannot recursively create more cron jobs, so prompts need to be self-contained.

This maps to the **WAT framework**: workflow, agent, tools. NOAGENT mode deploys just the workflow, like deploying a Python script on Modal without the agent layer.

Pillar 5: Self-Improving Loop

Hermes improves when useful experience gets persisted as memory, skills, and searchable history. The loop: do the work, agent learns, save things to memory, turn repeatable steps into skills, agent searches past sessions when context matters, repeat.
Automatic does not mean magic. The loop works best when you correct Hermes, ask it to save things to memory, and let it create and update skills after complex work.

Honorable Mention: Context File (agents.md)

agents.md is the project-level context file (equivalent to claude.md in Claude Code or AGENTS.md in Codex). It describes the overall project goal and structure for a contained working environment. More relevant when using Hermes in the terminal for coding work.

Setting Up Your VPS on Hostinger

A **VPS** (virtual private server) is a computer in the cloud you rent from a provider. You get an IP address and password and SSH in to manage files and install things. Nate uses **Hostinger** for Hermes, n8n, OpenCode, and Claude Code.

Choosing a Plan

- KVM 1, 2, 4, or 8 (CPU, RAM, and bandwidth tiers).
- Start with KVM 2 and scale up later if needed.
- Annual plan recommended (around $100/year). Multiple agents can run on one VPS.
- Use code NATEHURK on annual plans for 10% off.

Configuration

- OS: Ubuntu 24.04 LTS.
- Hostinger has a one-click Hermes auto-deploy option.
- Add the free malware scanner.
- Set a memorable hostname (e.g., youtube-hermes.vps).
- Enable daily auto-backups.

Two Install Methods

- Root install: Hermes lives directly on the VPS at the root level. Run the install command in the Hostinger terminal.
- Docker container install: Hermes lives in a containerized environment inside the VPS. One-click install via the Docker Manager.

Docker is easier and recommended for the demo. It lets you run multiple separated agents on one VPS, each with its own keys, memory, and tools.

Managing Your Agents With a Claude Code Project

Nate maintains a Claude Code project called uppit-agents to track every VPS agent: passwords, environment variables, IP addresses, Docker setup, security notes. This is the secret to staying organized.

Why Set This Up

- Centralized place for all VPS configs and credentials.
- If a Hermes agent shuts down, Claude Code can root in and help you boot it back up.
- You don't have to memorize VPS commands or terminal syntax.

- Helps you communicate clearly when things break (just describe what you see).

What to Track Per Agent

- Admin username and admin password (saved to .env file in the project).
- IP address and hostname.
- Whether it's Docker or root install.
- Tool availability, skills, file paths.
- Security notes and integrations.

Onboarding Your Hermes Agent

Once Hermes is deployed, click Open in Hostinger to access the chat. Use the admin username and password you saved earlier.

Step 1: Choose an Inference Provider

Many providers available, but the cheapest non-open-source option is **OpenAI Codex**. This lets you use your existing ChatGPT subscription ($20, $100, or $200/month) instead of paying per API call.
Sign in via the URL Hermes provides, authorize access, copy the 9-digit code, paste it into the terminal.

Step 2: Choose a Model

Pick **GPT-5.5** for the best balance.

Step 3: Set Up Telegram

- Hit space to select Telegram, then enter to confirm.
- In Telegram, open BotFather and send /newbot.
- Give the bot a name and unique username.
- Copy the bot token and paste it into the VPS terminal.

Step 4: Authorize Your Telegram Account

- Search for the USERINFO bot in Telegram and message it to get your user ID.

- Paste your user ID into the terminal as the home channel user.
- Confirm yes when asked if this is the home channel.

Step 5: Save Setup Info

Hermes shows you your tool availability, settings paths, API keys location, and configuration paths. Copy all of this and paste it into your Claude Code project. You'll need it later when troubleshooting.

Connecting Hermes to a GitHub Repo

This is the first thing every new Hermes agent should do. If the VPS gets corrupted, you still have all your skills, memory, and config in GitHub. Just spin up a new Hermes and sync to the repo.

Setup Workflow

- Tell Hermes: "Set this up as a private GitHub repo. Do research and figure out how it works."
- Hermes already has GitHub repo management and GitHub Auth skills built in.
- It will ask for your GitHub username, repo name, commit identity, and personal access token.
- It will create a .gitignore so secrets don't get pushed even on private repos.

Personal Access Token Setup

- In GitHub, go to Settings, Developer settings, Personal access tokens.
- For first attempt: try a fine-grained token with contents read/write scope.
- If that fails (no permission to create repos), generate a classic token with full repo scope.
- Set expiration based on use (30 days for testing, longer for permanent setups).

How to Save the API Key Safely

Don't paste API keys directly into the chat. They live in conversation history forever. Use the Hermes config command instead.

- In the VPS, click Open to enter the chat, then hit Ctrl+C to exit Hermes chat into the Docker shell.
- Run: hermes config set GITHUB_TOKEN <paste-token-here>

- This saves the token to /opt/data/.env inside the Docker container, never the conversation log.
- Tell Hermes: "The new token is in the .env file as GITHUB_TOKEN." It will pick it up.

Editing or Deleting Keys

If you need to delete or update a key, ask Hermes for the nano command to open the .env file. Make sure you're editing the .env inside the Docker container, not the root VPS .env. Use Ctrl+O then Enter to save, Ctrl+X to exit.
If nano isn't installed, ask Hermes to install it or give you an alternative editing command.

Building Your First Skill and Cron

Once GitHub is connected, set up a daily backup cron. Just say: "Every night at midnight Central Time, push changes to this GitHub repo. Build a skill around this."

What Hermes Does Behind the Scenes

- Views relevant existing skills.
- Runs terminal commands to set up the cron.
- Creates a new skill (e.g., nightly-github-sync).
- Updates memory.md with the new context.
- Handles timezone differences automatically (container runs in UTC, but it self-checks Central Time so daylight savings doesn't break it).

Permission Levels in Telegram

When Hermes wants to run a command, it asks: allow once, allow for the session, or always allow. For repetitive trusted actions like committing to a repo, choose Always Allow so the cron runs without prompts.

Two Paths to Build a Skill

- Describe the outcome in natural language and let Hermes build it.
- Install one from the skills library: copy the skill URL, paste it to Hermes, say "Install this skill and run it."

CLI vs Telegram: Which to Use When

Functionally it's the same agent in both interfaces. Telegram does not run a weaker version. The difference is visibility and control.

CLI Is the Cockpit

- Best for deep work, building, coding, living inside the agent.
- Better context window visibility.
- All slash commands available.
- Use when sitting at your computer doing high-risk work.

Telegram Is the Remote Control

- Best for scheduled tasks, quick check-ins, on-the-go work.
- Less visibility into context window and session state.
- Auto-compaction happens under the hood as you approach token limits.
- Don't vibe-code hardcore apps from Telegram. Use it for low-risk tasks.

Important Note on Context

Context is token-based, not message-based. The model always sees your system prompt, **user.md**, soul, etc. As you near the context window limit, Hermes runs auto-compaction. In Telegram you can't see exactly when this happens, so high-stakes work is riskier there.

API Key and Security Best Practices

Treat Hermes like a new employee or intern. You wouldn't hand a new hire your credit card. Same rule applies here.

Account and Key Hygiene

- Give each Hermes agent its own Gmail or AgentMail account, not your personal one.
- If you must share a key, use one with very strict scopes.

- Use named API keys per agent (separate OpenRouter or Perplexity keys) so you can track spending per agent.
- Apply the principle of least privilege: only the credentials and tools needed for the job.

VPS Hardening

- Set up a firewall in Hostinger (Nate uses one called Upguard).
- Lock down to your IP and block unused ports.
- Ask Hermes or Claude Code to research your environment and recommend a firewall config.
- Build a nightly or weekly security audit skill that checks for vulnerabilities.

Maintaining Hermes Over Time

Hermes is not a tool you finish setting up. It's a teammate you keep training.

Maintenance Rules

- When the agent gets something wrong twice, correct it on the spot and tell it to update the relevant skill or memory.
- When you give the same instruction twice, ask Hermes to write a skill for it.
- When the agent is too verbose or off-tone, edit the soul.md.
- When you want a new scheduled task, build a skill and ask Hermes to schedule the cron.
- When something breaks, check memory.md first. Stale memory is the number one cause of weird agent behavior.
- At any time, ask: "Read me your memory file" or "Read me your soul file" to see what's actually in there.

Scaling: Multiple Hermes Agents

As you grow, you may want dedicated agents for different roles. Don't force it. Use the main one until you naturally hit the criteria below.

How to Separate Agents on One VPS

- Each agent runs in its own Docker container.
- Each container has its own memory, tools, and private API keys.
- Containers don't share .env files, so keys don't clash across agents.
- Office building analogy: VPS is the building, each agent is a separate office with its own keys.

Decision Tree for a New Agent

- Does it need different permissions, secrets, or tools? If yes, create a new agent.
- Does it need separate long-term memory? If yes, create a new agent.
- Is it ongoing or repeated work (not a one-off)? If yes, create a new agent.
- If none of these apply, keep it inside your main personal Hermes.

Bad Pattern vs Good Pattern

- Bad: one mega-agent with all API keys, all skills, every cron. High confusion, high risk if it breaks.
- Good: split agents by vertical (marketing, finance, ops) or by platform/role. Cleaner memory, easier debugging, lower risk.
- Migration is easy because skills, memory, and crons are just markdown files. Move them between agents as needed.

Hermes Dashboard

Hermes ships with a built-in dashboard for recent sessions, connected platforms, a **Kanban board**, keys, configs, skills, and plugins.

How to Open It

Tell Hermes you want to open the dashboard. Give it your VPS route and Docker container setup info. Hermes will open the gateway and tunnel so you can access the local dashboard. The first attempt may feel rough. Once it works, save it as a skill so future opens are: "Open dashboard" then run three commands.

When It's Useful

- Multiple agents working on different projects (Kanban view).
- Visualizing tasks across agents.
- Quick way to manage crons, plugins, and configs.
- Nate rarely uses it because his Hermes work happens on the go via Telegram.

Key Takeaways

1. Hermes is best for on-the-go automation through Telegram. Claude Code stays the daily driver for desk-bound knowledge work.
2. The five pillars (Memory, Skills, Soul, Crons, Self-Improving Loop) are the mental model for everything Hermes does.
3. Use the Hostinger one-click Docker install on Ubuntu 24.04 LTS. Code NATEHURK gets 10% off annual plans.
4. Always set up a Claude Code project to track your VPS agents (passwords, IPs, configs). It pays off the first time something breaks.
5. First thing after onboarding: connect Hermes to a private GitHub repo and set a daily backup cron.
6. Never paste API keys into chat. Use hermes config set KEY_NAME <value> from inside the Docker container.
7. Treat each Hermes like a new hire: principle of least privilege, named API keys per agent, separate accounts where possible.
8. Stale memory.md is the number one cause of weird agent behavior. Check it first when things go sideways.
