

Hermes Agent — het complete naslagwerk

Dit is je master-naslag over Hermes Agent van Nous Research, met focus op de Desktop-app, op Claude via je Max-abonnement, en op kosten. Onderzocht met de volledige broncode van jouw eigen installatie (v0.20.4) plus de officiële docs. Bij elke claim staat een bron: een bestandspad of een link.

In het kort (lees dit eerst)

- **Hermes is gratis software** (MIT). Je betaalt alleen het model dat je aanroept.
 - **Claude via je Max-abonnement kan** in Hermes. Maar: alleen **Claude Max plus gekochte extra credits**. Claude Pro werkt niet.
 - **De herinlog-storm** die je had, komt door één seat plus meerdere bots die tegelijk om die login vechten, plus een handmatige token die niet ververs.
 - **Gezeikvrije weg naar Claude:** doe het via `hermes auth add anthropic --type oauth` (zodat Hermes de token zelf ververs), met een API-key als terugval in een credential-pool. Zet maar **één** bot op die OAuth.
 - **Kosten-veiligste keuze blijft Codex (ChatGPT Team):** in je eigen data kostte Codex **0 dollar per token**, terwijl Claude Opus via een losse API-key naar **circa 30 dollar** liep.
 - **De storing van vandaag** zat in één sessie-rij in de database, niet in je profiel. Wissel modellen voortaan in een **verse chat** via de **picker**, niet met vrij getypte provider-namen.
-

1. Wat is Hermes en hoe werkt het

In één zin

Hermes is een AI-agent die overal draait en via chat praat. Hij gebruikt elk model dat je wilt, leert van zijn eigen ervaring, en je bedient hem via Telegram, CLI of de Desktop-app. Bron: `README.md` , `website/docs/index.mdx` .

De kern-architectuur

- Er is **één motor** voor alles: de klasse `AIAgent` . CLI, Desktop, cron, API-server, gateway gebruiken dezelfde motor. Alleen het instappunt verschilt. Bron: `website/docs/developer-guide/architecture.md` .
- De **agent-loop** per beurt: jouw bericht toevoegen, system-prompt bouwen (identiteit, tools, skills, geheugen), checken of compressie nodig is, het model aanroepen, tool-calls uitvoeren, antwoord opslaan. Bron: `website/docs/developer-guide/agent-loop.md` .
- Hermes praat op **3 manieren** met modellen: `chat_completions` (OpenAI-achtig), `codex_responses` (Codex), `anthropic_messages` (Anthropic). Daarom kun je van provider wisselen zonder code te veranderen. Bron: `agent-loop.md` .
- De **gateway** is één langdraaiend proces dat alle chat-platforms bedient. Eén gateway kan 20+ platforms tegelijk aan. Bron: `architecture.md` , `index.mdx` .

Profielen = jouw bot-vloot

- Een **profiel** is een **aparte thuismap**: `~/.hermes/profiles/<naam>/` . Eigen config, `.env`, `SOUL.md`, geheugen, sessies, skills, cron, en `state.db`. Bron: `website/docs/user-guide/profiles.md` .

- Jouw bots (Beeld, Dirigent, Contentcodex, Web, School, Oxlabs, Contentclaude) zijn echt aparte mappen. Hermes zelf = het default-profiel = de `~/hermes -root`.
- **Harde regel:** nooit twee agent-processen op hetzelfde profiel. Beide schrijven geheugen en laden elkaars schrijfsels, dan loopt de staat vast. Bron: `profiles.md`.

Het zelf-verbeterende leer-mechanisme (het unieke)

Zes lagen die samenwerken:

1. **Geheugen** (`MEMORY.md` + `USER.md` , klein, altijd in de prompt). De agent beheert dit zelf. Het is een bevroren snapshot: wijzigingen verschijnen pas volgende sessie in de prompt. Bron: `features/memory.md` .
2. **Skills** (markdown-procedures, laden alleen als nodig, kosten 0 tokens tot dan). Bron: `guides/work-with-skills.md` .
3. **Autonome skill-creatie:** de agent schrijft zelf skills als hij een herhaalbare workflow vond of jij hem corrigeerde. Bron: `features/skills.md` .
4. **Background review:** na een beurt draait stil een review die een geheugen-regel of skill kan bijwerken. Je ziet dan `Memory updated` . Bron: `features/memory.md` .
5. **session_search:** doorzoekt met SQLite FTS5 al je oude gesprekken. Gratis, geen model-call. Bron: `features/sessions.md` .
6. **Honcho** (optioneel): bouwt een dieper model van wie je bent, over sessies heen. Bron: `features/memory-providers.md` .

Belangrijk voor jou: je kunt de zelf-verbetering **met een goedkeur-hek** aanzetten. `memory.write_approval: true` of `skills.write_approval: true` , dan review je met `/memory pending` of `/skills pending` . Bron: `features/memory.md` , `features/skills.md` .

Tools, MCP en subagents

- 70+ ingebouwde tools in een centraal register. Een tool zonder key verschijnt niet eens voor het model (kan niet gehallucineerd worden). Bron: `developer-guide/tools-runtime.md` .
- **MCP-servers** voeg je toe in `config.yaml` onder `mcp_servers:` , met per server een whitelist van tools voor gevaarlijke systemen. Herladen met `/reload-mcp` . Bron: `guides/use-mcp-with-hermes.md` .
- **Subagents** (`delegate_task`): tot 3 tegelijk, eigen verse context, alleen de eindsamenvatting komt terug. Anders dan bots: subagents zijn tijdelijk en context-loos. Bron: `features/delegation.md` .

Deployment: draait overal, praat via Telegram

- De agent kan zijn commando's op verschillende backends draaien: lokaal, Docker, SSH, Daytona, Modal, Vercel Sandbox, Singularity. Bron: `README.md` , `index.mdx` .
- Model: je draait de agent op een goedkope VPS of serverless, en praat er vanaf Telegram mee terwijl hij elders werkt. Serverless slaapt als hij niks doet en kost dan bijna niks. Bron: `index.mdx` .

Bot Mode: bots die elkaar berichten sturen

- Een **Bot** is gewoon een profiel. Bot Mode is een UI erover. Bron: `features/bot-mode.md` .
- Drie routes: **@mention** (`@researcher` kijk hier eens naar), **group chats** (2-6 bots, harde cap van 3 rondes en 10 berichten tegen lussen), en **directe berichten** via een tempbestand. Bron: `bot-mode.md` .
- Het messaging-protocol wordt automatisch in de system-prompt geïnjecteerd. Raakt je `SOUL.md` niet. Bron: `bot-mode.md` .
- Dit past op jouw regel dat de app-bots alleen interne comms doen: `@mention` is precies die interne route.

Sessies, staat en context

- Elk gesprek wordt bewaard als sessie in `state.db` (SQLite, WAL-mode, met FTS5-zoekindex). Per profiel apart. Bron: `features/sessions.md`.
- **Compressie** triggert boven 50% van het contextvenster (preflight) of 85% (gateway). Middelste beurten worden samengevat, de laatste 20 blijven intact. Bron: `agent-loop.md`.
- Grootste oorzaak van context-groei is niet media maar veel tekst (geplakte logs, transcripts). Beter: samenvattingen en bestandspaden. Sluit aan op je eigen regel. Bron: `sessions.md`.

2. Claude via je Max-abonnement (de kernvraag)

Kan het? Ja, maar met voorwaarden

- Er is een echte **Anthropic-OAuth-flow**, net als bij xAI Grok en MiniMax. Bron: `website/docs/integrations/providers.md`.
- **Vereist Claude Max plus gekochte extra usage-credits**. Claude Pro werkt niet. Bron: `features/credential-pools.md`, `providers.md`.
- Let op: Hermes gebruikt dan alleen je **extra credits**, niet je basis Max-tegoed. Bron: `providers.md`.

De vier auth-types van Hermes

Bron: `providers/base.py`, `developer-guide/model-provider-plugin.md`.

- **api_key** — sleutel in `.env` (Anthropic API-key, OpenAI, Gemini, enz.)
- **oauth_device_code** — code in browser (Nous Portal)
- **oauth_external** — je logt elders in, tokens landen in `auth.json` (Anthropic OAuth, MiniMax, xAI Grok, Nous Portal). Dit is wat Codex bij jou gebruikt.
- **external_process / copilot / aws / vertex** — bijzondere gevallen.

Waarom je herinlog-storm ontstond

- Claude-OAuth routeert "**as Claude Code**" tegen je **Anthropic-account**. Dat is in de praktijk **één seat**. Bron: `providers.md`.
- Meerdere bots plus jouw eigen Claude Code-sessie op diezelfde login = 401/429 = botsende token-refresh = opnieuw inloggen.
- Een **handmatige** `CLAUDE_CODE_OAUTH_TOKEN` verversst niet zelf. Na verval = harde re-login. In de code staat letterlijk: zo'n setup-token geeft als `x-api-key` een 401, als kale Bearer een 429. Bron: `plugins/model-providers/anthropic/__init__.py`.
- **Dus doe dat niet**. Gebruik `hermes auth add anthropic --type oauth`, dan beheert Hermes de refreshbare store. Bron: `providers.md`.

Hoe Hermes 429 en re-auth opvangt

Bron: `features/credential-pools.md`, `agent/credential_pool.py`.

- **429** (rate limit): 1x zelfde sleutel opnieuw, dan roteren. Cooldown 1 uur.
- **402** (billing): meteen roteren. Cooldown 1 uur.
- **401** (verlopen): eerst OAuth-token verversen, anders roteren. Cooldown 5 min.
- Je hebt in `profiles/contentclaude/auth.json` **al een credential-pool**: OAuth op prioriteit 0, API-key op prioriteit 1. Precies de goede opzet.

Concreet advies: twee opties

Optie A — meest zonder gezeik: losse Anthropic API-key.

Het Claude-equivalent van je Codex-truc: aparte pool, geen seat-conflict, meerdere bots tegelijk oké.

```
# in het profiel-config.yaml
model:
  provider: anthropic
  default: claude-sonnet-4-6
# in ~/.hermes/.env
ANTHROPIC_API_KEY=sk-ant-api03-...
```

- Voor: robuust, geen re-login, los van je abonnement.
- Tegen: kost geld per token.

Optie B — je Max-credits gebruiken: OAuth plus API-key-terugval, en maar één bot erop.

```
hermes auth add anthropic --type oauth # Max + extra credits, prioriteit 0
hermes auth add anthropic --type api-key --api-key sk-ant-api03-... # terugval, prioriteit 1
model:
  provider: anthropic
  default: claude-sonnet-4-6
credential_pool_strategies:
  anthropic: fill_first # eerst OAuth-credits, dan de API-key
```

- Voor: gebruikt je extra credits, valt bij 429 automatisch terug op de API-key in plaats van een re-login-storm.
- Tegen: vereist Max plus gekochte credits. Zet niet meerdere bots én je eigen Claude Code tegelijk op deze OAuth.

Let op valkuil: bij het wisselen van sleutel of provider **reset de prompt-cache**. De volgende call leest de hele context tegen vol tarief. Veel heen-en-weer springen is dus duur. Bron: `credential-pools.md` , `fallback-providers.md` .

En de Desktop-app zelf?

De model-provider staat op de **backend**, niet in de app. De Desktop-app logt via de systeem-browser in op je Hermes-gateway (dat is een andere login dan je model-provider). Je configureert Claude dus op het profiel/backend waar de app mee verbindt, via `hermes auth add anthropic --type oauth` of `hermes model` . Bron: `guides/desktop-native-signin.md` , `user-guide/desktop.md` .

3. De Desktop-app beheersen

Wat de app is

- Dezelfde agent als de CLI. Begin een sessie in de app, hervat hem in de terminal, en andersom. Bron: `user-guide/desktop.md` .
- De app start een **eigen headless backend** (`hermes serve`). De React-UI praat daarmee. Bron: `desktop.md` .
- Let op de dubbele naam "gateway": de **backend** is voor chat, de **messaging-gateway** (Telegram/Discord) is een apart proces dat je los draaiend houdt. In de app is "Gateway" ook de naam van de verbindings-instellingen (Settings → Gateways). Bron: `desktop.md` , `multi-connection-desktop.md` .

Model kiezen: drie lagen (dit veroorzaakte de storing)

1. **Composer-modelkiezer** (links van de microfoon): sticky UI per apparaat, raakt je default nooit aan. Bij een live chat geldt het alleen voor die chat. Bron: `desktop.md` .
2. **Per-profiel default** (Settings → Model): de globale default voor nieuwe chats, crons en subagents. Bron: `desktop.md` .
3. **/model -commando** in een chat: alleen die sessie. Met `--global` past hij ook `config.yaml` aan, met `--once` alleen de volgende beurt. Bron: `configuring-models.md` .

Waar de keuze wordt bewaard (de oorzaak van de 401 vandaag)

De per-sessie keuze staat in `state.db` , tabel `sessions` , in twee kolommen: `model` én `model_config` (JSON). Iemand typte `/model opencode-zen:x-preview-f-free` in één sessie. Dat schreef `provider: opencode-zen + model: x-preview-f-free` naar die sessie-rij. Bij de volgende beurt hervatte de sessie exact dat model → HTTP 401 "Model not supported". De profiel-default (`gpt-5.6-sol`) was nooit veranderd; alleen die ene sessie was besmet. Bron: geverifieerd in je eigen `state.db` + `desktop.md` .

Hoe je die fout voorkomt

- Wissel modellen **vroeg in een chat of start een verse chat**, niet heen en weer (mid-chat wissel reset ook de cache). Bron: `desktop.md` .
- Gebruik alleen de **picker met curated modellen**. Vrij typen (zoals `opencode-zen:x-preview-f-free`) omzeilt de check. Bron: `configuring-models.md` .
- Voor een eenmalige zware vraag: `--once` , dan herstelt hij vanzelf.
- Een sessie terugforcen: typ in die chat `/model gpt-5.6-sol --provider openai-codex` (precies de fix van vandaag), of pas de sessie-rij offline aan met backup.

Meerdere sessies tegelijk (jouw 2-terminals-werkwijze)

- Meerdere sessies naast elkaar mag, ook over meerdere profielen. Tabs met `Cmd+T`, wisselen met `Ctrl+Tab`. Bron: `desktop.md` .
- **Sessie-state is gedeeld tussen alle front-ends**. Live output streamt naar elk venster dat die sessie toont. Bron: `desktop.md` .
- Harde regel blijft: **nooit twee processen op hetzelfde profiel**. Twee verschillende profielen tegelijk is prima. Bron: `profiles.md` .
- Telegram: per profiel een eigen bot-token in `.env` . Twee profielen op hetzelfde token worden geblokkeerd. Bron: `profiles.md` .

Remote backend (jouw Mac Mini in Spanje via Tailscale)

- Settings → Gateways → Remote gateway, wijs naar een `hermes serve` op de andere machine. Bind aan het Tailscale-IP, poort 9119. Bron: `desktop.md` .
- Voor trusted VPN (Tailscale) mag username/password. Zet `HERMES_DASHBOARD_BASIC_AUTH_SECRET` stabiel, anders word je bij elke herstart uitgelogd. Bron: `desktop.md` .

Master-tips

- **Context-meter** in de statusbalk: klik voor een uitsplitsing (systeemprompt, tools, skills, memory, MCP, gesprek). Bron: `desktop.md` .
- **Command palette** (`Cmd+K`): spring naar elke pagina, sessie, model-wissel. Bron: `desktop.md` .

- **Profiel exporteren** (Cmd+K → Export profile): een tar.gz met skills, memory, persona, crons, keys gestript. Bron: `desktop.md`.
- **Worktrees** (Cmd+Shift+B): parallel werken zonder je repo aan te raken. Past op je backup-regel. Bron: `desktop.md`.
- **Boot-fout?** Log in `HERMES_HOME/logs/desktop.log` of `hermes logs gui -f`. Bron: `desktop.md`.

Let op: jouw app is v0.17.0 terwijl de repo op v0.20.4 staat. De model- en profiel-mechaniek is geverifieerd tegen je echte config. Sommige nieuwste app-features kunnen in 0.17.0 net anders heten. Een update naar 0.20.x is te overwegen.

4. API's, kosten en zuinig gebruik

Twée soorten kosten (het belangrijkste)

Hermes kent per model-aanroep een `billing_mode`. Bron: `agent/usage_pricing.py`, tabel `session_model_usage`.

- **Abonnement** (Codex/ChatGPT Team, Claude Max OAuth): `subscription_included`, kosten per token = **0**. Je betaalt een vast maandbedrag. Bron: `usage_pricing.py`.

- **API-key** (Anthropic-key, OpenRouter): betaal **per token**. Lange sessies = oplopende rekening.

Voor een vloot die veel draait is een abonnement dus veiliger tegen verrassingen.

Wat je eigen data laat zien (profiel Beeld, echt gemeten)

PROVIDER	BILLING	MODEL	CALLS	GESCHATTE KOSTEN
anthropic (API-key)	per token	claude-opus-4-8	154	circa 25,75 dollar
anthropic (API-key)	per token	opus (background review)	18	circa 2,46 dollar
openai-codex	abonnement	gpt-5.6-sol	28	0,00 dollar
opencode-zen	onbekend	x-preview-f-free	24	0,00 (de kapotte)

Kort: Codex kostte niets per token, Claude Opus via API-key liep naar circa 30 dollar. Bron: `live session_model_usage` in `profiles/beeld/state.db`.

Zelf je kosten zien

- Zet `show_cost: true` in config (standaard uit) → dollars in de statusbalk. Bron: `hermes_cli/config_defaults.py`.
- `/usage` in de CLI toont het overzicht. Bron: `hermes_cli/auth.py`.
- Per bot: query read-only de tabel `session_model_usage` in dat profiel.

Ingebouwde besparingen

- **Context-compressie** (standaard aan): vat oude stukken samen. Laat het door een **goedkoop model** doen via `auxiliary.compression`. Bron: `cli-config.yaml.example`.
- **Prompt-caching**: herlezen van gecachte context kost 75-90% minder. Maar een provider-wissel of key-rotatie **wist de cache**. Bron: `fallback-providers.md`.
- **Bijtaken op goedkoop/lokaal model**: vision, titels, compressie, approval elk een eigen provider en `reasoning_effort` (zet op low/none). Bron: `cli-config.yaml.example`.

- **Harde remmen:** `max_tokens` (cap op output), `tool_loop_guardrails.hard_stop_enabled` (stopt een vastlopende tool-lus). Aanrader voor cron/autonome bots. Bron: `cli-config.yaml.example` .

Rate limits en fallback

- **Credential-pools:** meerdere keys per provider, roteert bij 429/402/401. Bron: `credential-pools.md` .
- **Fallback-providers:** `fallback_providers`: lijst, schakelt over bij aanhoudende fouten. Per-turn: elk nieuw bericht probeert eerst weer je hoofdmodel. Reset-aware bij Claude Max 5-uurs-blokken. Bron: `fallback-providers.md` .
- Een pool geeft **veerkracht, geen extra capaciteit**. Test of de backup echt quota heeft.

Best practices (drie lanen)

1. **Primaire laan:** betrouwbaar betaald model voor kritiek werk.
2. **Goedkope laan:** lokaal/goedkoop model voor samenvatten, classificeren, routine.
3. **Auxiliary-laan:** aparte compressie- en memory-routes.

Kernregel uit de community: **reken op totale kosten per afgeronde taak, niet op prijs per token**. Een goedkoop model dat 10x herprobeert is duurder dan een duur model dat het in één keer goed doet. Precies je eigen team-regel.

Advies voor jouw vloot

1. **Codex (ChatGPT Team) als brein blijft kosten-technisch veilig.** Vast maandbedrag, 0 per token, ideaal voor veel bots.
2. **Ga je naar Claude, doe het via Max-OAuth** (subscription-included), niet via een losse API-key waar de teller doorloopt.
3. **Zet een korte fallback-keten** zodat een bot nooit stilvalt, maar houd het kort (elke wissel wist de cache).
4. Zet `show_cost: true` en gebruik `/usage` .
5. **Aparte credentials per bot** voor de primaire laan, plus pools voor veerkracht.
6. **Cron-taken spreiden**, achtergrondwerk op een goedkoop model.
7. **Valse stilte:** een zwijgende bot betekent meestal quota/credits op of een verkeerde route, niet dat het platform stuk is. Meet aan de kosten-kant.

5. Waar je Hermes leert

Officiële hoofdbronnen (geverifieerd)

- Docs: <https://hermes-agent.nousresearch.com/docs/>
- GitHub: <https://github.com/NousResearch/hermes-agent>
- Desktop: <https://hermes-agent.nousresearch.com/desktop>
- Nous Research: <https://nousresearch.com/>
- Discord (officieel): <https://discord.gg/jqVphNsB4H>
- Hele docs als één tekstbestand (handig voor AI): <https://hermes-agent.nousresearch.com/docs/llms-full.txt>

Het officiële learning path (leer-volgorde)

- **Beginner (circa 1 uur):** Installation → Quickstart → CLI Usage → Configuration. Tip: draai eerst `hermes setup --portal` .
- **Intermediate (2-3 uur):** Sessions → Messaging → Tools → Skills → Memory → Cron.
- **Advanced (4-6 uur):** Architecture → Adding Tools → Creating Skills → Contributing.

Voor jouw doel het handigste use-case-pad ("team van specialist-bots"): Desktop → Profiles → Bot Mode → Cron → Multi-connection Desktop.

Must-read docs voor meesterschap

- <https://hermes-agent.nousresearch.com/docs/user-guide/bot-mode>
- <https://hermes-agent.nousresearch.com/docs/user-guide/profiles>
- <https://hermes-agent.nousresearch.com/docs/user-guide/desktop>
- <https://hermes-agent.nousresearch.com/docs/user-guide/configuring-models>
- <https://hermes-agent.nousresearch.com/docs/user-guide/security>
- <https://hermes-agent.nousresearch.com/docs/user-guide/features/memory>
- <https://hermes-agent.nousresearch.com/docs/user-guide/features/skills>
- <https://hermes-agent.nousresearch.com/docs/user-guide/features/mcp>
- <https://hermes-agent.nousresearch.com/docs/integrations/providers>
- <https://hermes-agent.nousresearch.com/docs/reference/faq>

Praktijk-gidsen (bouwprojecten die op jouw setup lijken)

- <https://hermes-agent.nousresearch.com/docs/guides/team-telegram-assistant>
- <https://hermes-agent.nousresearch.com/docs/guides/daily-briefing-bot>
- <https://hermes-agent.nousresearch.com/docs/guides/tips>

Community en video (aanvullend, kritisch lezen)

- DataCamp tutorial: <https://www.datacamp.com/tutorial/hermes-agent>
- Hermes Atlas (beginnersgids): <https://hermesatlas.com/guide/>
- Quickstart-walkthrough (YouTube): <https://www.youtube.com/watch?v=UEszjeHEeSo>
- 20 Hidden Agent Features (YouTube): <https://www.youtube.com/watch?v=PMPYUCu3opU>

Eerlijk: de community is nog jong en dun. Voor master worden leun je 90% op de officiële docs plus de code in je lokale repo. De echte code is de ultieme bron van waarheid.

6. Stappenplan voor jou

1. **Nu:** Hermes werkt weer (de kapotte modellen zijn teruggezet naar gpt-5.6-sol). Codex als brein houden = veilig en gratis per token.
2. **Model wisselen doe je voortaan** in een verse chat via de picker, nooit met vrij getypte provider-namen.
3. **Wil je Claude erbij:** kies eerst API-key (optie A, simpel) of Max-OAuth met terugval (optie B). Zet maar één bot erop.
4. **Zet `show_cost: true`** zodat je live ziet wat draait.
5. **Overweeg een update** van de Desktop-app (v0.17.0 → v0.20.x) om gelijk te lopen met de docs.
6. **Leren:** volg het learning path, lees de must-reads, en bouw één gids na (team-telegram-assistant lijkt op jouw vloot).

Bronnen

Lokale broncode + officiële docs in `/Users/macbookjamal/.hermes/hermes-agent/` (v0.20.4): `README.md` , `website/docs/` (`getting-started`, `user-guide`, `features`, `integrations`, `guides`, `developer-guide`, `reference`), `providers/` , `plugins/model-providers/` , `agent/credential_pool.py` , `agent/usage_pricing.py` , `hermes_cli/` , `docs/billing-lifecycle.md` , en de live `state.db` van je profielen. Online: hermes-agent.nousresearch.com/docs, github.com/NousResearch/hermes-agent.

Onzekerheden die niet te sluiten waren: exacte maandprijs van Nous Portal (staat achter de sign-up), exact aantal seats per Claude Max, en of de gemeten Anthropic-kosten "actual" of alleen "estimated" waren. Deze staan eerlijk als onzeker gelabeld.