Hermes-geheugenarchitectuur voor Haarvisie

Kort: Jamal vroeg uit te zoeken of je GBrain en Mnemosyne kunt combineren voor een beter geheugen, en stuurde een architectuur-visie door (Hermes als Conductor, Mixture-of-Agents, email-gateway). Dit verslag legt het uit, checkt het tegen wat je AL hebt, en geeft een concreet stappenplan. **Conclusie: de architectuur klopt, past verrassend goed op je bestaande stack, en we kunnen 'm gefaseerd bouwen — cherry-pick, geen heel nieuw framework.**

---

1. De gelaagde geheugenlaag: GBrain vs Mnemosyne

Twee geheugens met een compleet andere, aanvullende rol:

- **GBrain = het langetermijngeheugen** (de "compiled truth"). Een **markdown-brein dat jij bezit**, met een Postgres-index eroverheen zodat het semantisch doorzoekbaar is. Diep, analytisch, leesbaar, version-controlled. Bevat wereld-feiten over mensen, klanten, projecten. Te zwaar/traag voor real-time loops.
- **Mnemosyne = het kortetermijn-/werkgeheugen** (de "Cortex/Cache"). Ultrasnel (sub-milliseconde), zero-dependency (SQLite, 1 bestand). Onthoudt wat de agent één stap geleden deed — cruciaal tijdens snelle tool-executies.

Samen: snelle cache + duurzaam doorzoekbaar langetermijngeheugen. Dit is precies de aanbevolen Hermes-geheugenstack.

---

2. De Conductor: Hermes Agent

In plaats van een los orkestratie-framework kan **Hermes** de regisseur (Conductor) zijn:

- Stuurt de LLM-cycles aan, hakt knopen door, regisseert beide geheugens (Mnemosyne voor snelle updates, GBrain via MCP voor diepe achtergrondkennis).
- Heeft een **closed learning loop** (bouwt autonoom "skills" op).

- Draait als achtergrond-service, aanstuurbaar via Telegram/Discord — precies zoals jouw bots nu.

---

3. De denkkracht: Mixture of Agents (MoA)

Om de beslissingskwaliteit te maximaliseren:

- **Specialisten (reference models):** meerdere modellen (bijv. GPT-5 + DeepSeek) schrijven parallel en onafhankelijk een concept-antwoord.
- **Voorzitter (aggregator):** een zwaargewicht (bijv. Claude Opus) leest alle concepten, combineert met de diepe context uit GBrain, corrigeert fouten en schrijft het definitieve antwoord.
- Zit ingebouwd in nieuwe Hermes-versies (`hermes moa configure`).
- ⚠ **Kost fors meer tokens.** Advies: alleen inzetten voor complexe codeer-/analysetaken; voor regulier werk terug naar één model.

---

4. Use-case: Email-gateway (IMAP/SMTP) + MoA

Hermes heeft een ingebouwde email-gateway (IMAP + SMTP). Met MoA erachter wordt je mailbox een intelligente backoffice:

- **Autonome klant-triage:** leest inbox, MoA analyseert, aggregator checkt GBrain voor klanthistorie/documentatie, stelt een perfect antwoord op, mailt in-thread terug.
- **Inbox sorteren/labelen** + acties uitzetten (ticket, taak).
- **Data-extractie** uit facturen/rapportages naar database/Sheet.

⚠ **VEILIGHEID (kritiek):**
- NOOIT je persoonlijke hoofdmailbox — maak een aparte inbox (bijv. `agent@...`).
- Gebruik een **app-wachtwoord**, niet je echte wachtwoord.
- Zet een **whitelist** (`EMAIL_ALLOWED_USERS`) zodat de agent alleen op vertrouwde afzenders reageert.
- Credentials in een afgeschermd `.env` , nooit in plaintext/queue (jouw eigen regel).

5. Waarom dit bij Haarvisie past — je hebt de bouwstenen AL

LAAG IN DE VISIE	WAT JE NU AL HEBT
Markdown-brein (GBrain)	**Obsidian** — 1600+ notities, onder git, jij bent eigenaar
Snel semantisch geheugen (Mnemosyne)	**Mnemosyne** draait al (auto-vastleggen + betekenis-zoeken)
Conductor/regisseur	**Conductor** (deze MacBook) + de bot-vloot
Hermes-agent	**Hermes** draait al (Kore-stem, retrieval, "Bespreek met Hermes"-knop)

Wat GBrain concreet TOEVOEGT: een semantische Postgres-index over je markdown. Obsidian zoekt nu op trefwoord; GBrain zoekt op **betekenis** ("rijwiel" vindt "fiets"). En je blijft eigenaar van de data — past bij je regel *geheugen onder git*.

---

6. Concreet stappenplan (gefaseerd, cherry-pick)

1. **Fase 1 — GBrain-test op Conductor.** Installeer de gbrain-CLI, start een lokaal brein (PGLite, geen cloud nodig), richt het op je bestaande geheugen/Obsidian-markdown, koppel via MCP. Meet: vindt het dingen die Obsidian-trefwoordzoeken mist? *(Skill `setup-gbrain` staat al klaar.)*
2. **Fase 2 — Gelaagd koppelen.** GBrain (langetermijn) naast Mnemosyne (snel). Conductor/Hermes gebruikt beide.
3. **Fase 3 — MoA gericht.** Alleen voor zware analyse/code (kostenbewust); regulier werk op één model.
4. **Fase 4 (optioneel) — Email-gateway.** Aparte inbox + app-wachtwoord + whitelist. Eerst read-only triage/labelen, pas na akkoord automatisch antwoorden.

---

7. Advies

- □ De architectuur is legitiem en cutting-edge (aanbevolen Hermes-stack).
- □ Sluit naadloos aan op wat je hebt — geen sloop, alleen een extra semantische laag.
- ⚠ Let op tokenkosten bij MoA en op veiligheid bij de email-gateway.
- □ **Start klein:** GBrain-test op Conductor (fase 1). Meet het echte voordeel voordat we de rest uitrollen.

Bronnen: vectorize.io/articles/what-is-gbrain · github.com/AxDSan/mnemosyne · hermesatlas.com/guide/memory

Verslag door Conductor, 3 juli 2026.