



id: SI_002

model: qwen/qwen3.6-plus

seeds:

04_risiko: usikkerhet

03_relasjon: konkurranse

05_handling: implementer

01_perspektiv: juridisk

02_tidhorisont: langsiktig

Instans SI_002

Seeds: juridisk | langsiktig | konkurranse | usikkerhet | implementer

Orkestrering blir snart standardinfrastruktur. Din langsiktige konkurransefortrinn bygges ikke på antall agenter, men på juridisk og arkitektonisk robusthet. I et marked preget av usikkerhet rundt API-er, modell-deprekeringer og datasporing, vinner den som designer for feiltoleranse og ren data provenans. Ikke kopier opphavrettslig tvilsomt innhold direkte til din KB. Lagre kun hashes, lenker og strukturerte metadata. Det sikrer deg juridisk og gjør systemet fremtidssikkert.

Implementer en tretrinns-pipeline på din server: **Agent 1 (Ingestion)** henter X via RSS-broer og YouTube via `yt-dlp` + `whisper.cpp`. **Agent 2 (Orchestrator)** ruter via OpenRouter: bruk `claude-haiku` for bulk-transkripsjon, `claude-sonnet` for KB-sammenligning, og kun `gpt-4o` når semantisk tvetydighet > 0.7 . **Agent 3 (Gatekeeper)** filtrerer ved å kun flagge avvik $> 15\%$ fra din eksisterende base eller identifiserte juridiske risk-faktorer.

Mainstream råder deg til å følge alt. Gjør det motsatte: bygg en ignorance-filter. Svermen skal kun levere signaler som reelt endrer din strategiske posisjon. Resten arkiveres i et lokalt `Streamlit`-dashboard med SQLite, som synker til Obsidian via `obsidian-local-rest-api`. E-post til `raymond@sverm.ai` trigges kun ved konsensus-score > 0.85 etter tre agenter.

Neste steg:

1. Sett opp `LangGraph` + OpenRouter med en daglig cron-job som simulerer API-svikt og tester automatisk fallback til alternative modeller.
2. Koble `yt-dlp` / `whisper.cpp` direkte til Obsidian REST API; fjern all manuell e-post-forwarding fra workflowet.



3. Definer 5 juridiske “kill-switch”-kriterier (f.eks. plattform-ToS-brudd, lisensendring, opphavsrettsflagg) som agenter bruker til å blokkere ingest før den når din KB.