



id: SI\_009

model: google/gemma-4-26b-a4b-it

seeds:

04\_risiko: sårbarhet

03\_relasjon: tillit

05\_handling: forhandle

01\_perspektiv: strategisk

02\_tidhorisont: kortsiktig

## Instans SI\_009

**Seeds:** strategisk | kortsiktig | tillit | sårbarhet | forhandle

### 1. Arkitektur: Strategisk splitting vs. Taktisk låsning

Glem Copilot Studio for Case 1/2; det er for rigid for 100 parallelle agenter. **Strategisk valg:** Azure OpenAI Service (AOAI) + Python (LangGraph/Semantic Kernel) kjørende på Azure Container Apps.

- **Trade-off:** Copilot Studio er “safe” (høy **tillit**), men mangler orkestreringsdybde. Ved å bruke AOAI beholder du data residency i din Azure-tenant (sikrer **tillit**), men du må selv håndtere auth via Managed Identities.
- **Forhandling:** Du forhandler med kompleksitet for å vinne kontroll over latency.

### 2. Spesialisering: Fra duplisering til divergens

For Case 1 må du unngå “mean opinion” kollaps.

- **Implementering:** Bruk en Hierarchical Swarm. En “Orchestrator” dekomponerer problemet til et Task Schema. Hver agent får ikke bare en prompt, men en unik `System_Persona` + `Knowledge_Slice` (RAG-subset).
- **Syntese:** Bruk en “Synthesizer Agent” som ikke oppsummerer, men kategoriserer output i en Conflict Matrix (hvor er agentene uenige?). Dette eksponerer **sårbarhet** i beslutningsgrunnlaget fremfor falsk konsensus.

### 3. Micro-case Processing: Inbox-modell

- **Arkitektur:** Azure Service Bus (Queue) → Function App (Dispatcher) → Agent Swarm (Workers).



- **Consistency:** Bruk en “Base Template” med strenge JSON-skjemaer. Hver agent må returnere strukturerte data. Variabel kompleksitet håndteres ved at Dispatcher måler “token-estimate” og ruter til enten GPT-4o (kompleks) eller GPT-4o-mini (enkel).

## 4. Data & Sikkerhet: Den kritiske sårbarheten

I helsevesenet er “data leakage” din største **sårbarhet**.

- **Løsning:** Ingen data ut av Azure. Bruk Azure AI Search med OneLake integrasjon. Bruk Entra ID (tidligere Azure AD) for “User-level security trimming” i RAG-prosessen. Agenten ser kun det brukeren har tilgang til.

## 5. Operasjonell overvåking & ROI

- **Kortsiktig kontroll:** Implementer Azure Monitor og Application Insights. Logg hver agent-kall med `CorrelationID`.
- **Cost-optimering:** Implementer “Token Budgeting” per case. Hvis en agent bruker >X tokens uten fremgang, må den termineres (kill-switch).

## 6. Transition Roadmap

1. **Phase 1 (0-3 mnd):** “Shadow Swarm” – Batch-kjøring av Case 2 (distribuert prosessering) for å validere nøyaktighet mot eksisterende manuelle prosesser.
2. **Phase 2 (3-9 mnd):** Integrasjon av Case 1 i beslutningsstøtte for ledelse.

### Neste steg:

1. Sett opp en Proof of Concept (PoC) i en isolert Azure Sandbox med LangGraph.
2. Definer JSON-skjemaet for Case 2 for å sikre datakonsistens.
3. Kjør en risikoanalyse på “Agent Hallucination” i kontekst av helsedata.