Berikut adalah detail pekerjaan harian untuk memperbaiki dan melengkapi arsitektur NLP/AI agar sesuai dengan best practice Agentic AI modern dan siap diintegrasikan dengan backend microservices:

🗓️ Rencana Pekerjaan Harian (8 Hari)

✅ Day 1 – Review & Setup

Review ulang arsitektur NLP/AI yang sudah ada.

Setup repo khusus untuk NLP/AI service.

Buat struktur folder modular: llm_engine, embedding_model, vector_db, agent_executor, tools.

Tambahkan dokumentasi arsitektur awal di README.

✅ Day 2 – Fallback Multi-Provider LLM

Implementasikan wrapper untuk LLM (misalnya LangChain + OpenAI + Claude + LLaMA lokal).

Buat mekanisme fallback otomatis jika provider gagal.

Buat konfigurasi model di .env agar bisa diganti tanpa ubah kode.

✅ Day 3 – Hybrid Retrieval (BM25 + Embedding Search)

Tambahkan komponen keyword-based retrieval (BM25/ElasticSearch) selain embedding search di Vector DB.

Buat service retriever yang bisa menggabungkan hasil semantic + keyword search.

Uji kualitas pencarian untuk soal/materi.

✅ Day 4 – RBAC & SSO Integration

Tambahkan Role-based Access Control (RBAC) untuk API (admin, guru, siswa).

Integrasikan dengan Auth Service dari backend utama menggunakan JWT/SSO.

Tambahkan middleware autentikasi API key.

✅ Day 5 – Scalability & Auto-Scaling

Buat konfigurasi Docker + Kubernetes deployment untuk AI service.

Implementasi Horizontal Pod Autoscaler (HPA) untuk skenario traffic tinggi.

Tambahkan caching layer (Redis) untuk mempercepat response.

✅ Day 6 – Integrasi ke Backend Microservices

Buat diagram komunikasi antara Agentic AI Service ↔ User Service, Question Service, API Gateway.

Implementasikan API/gRPC call ke backend utama.

Buat service discovery (misalnya pakai konsul/istio).

✅ Day 7 – Observability & Security

Tambahkan logging (ELK Stack / OpenTelemetry).

Implementasikan rate-limiting untuk API.

Tambahkan monitoring LangSmith + Prometheus/Grafana.

✅ Day 8 – Final Testing & Dokumentasi

Lakukan end-to-end testing dengan dummy data SAT/UTBK.

Buat dokumentasi API lengkap (OpenAPI/Swagger).

Finalisasi README dengan arsitektur baru, cara deploy, dan cara integrasi.