Team OS – Detailed Architecture

Agentic, White‑Label API Platform

Date: 2025‑06‑10

# 1 · Core Runtime Layers

|  |  |  |
| --- | --- | --- |
| Layer | Purpose | Primary Tech |
| Interaction | Partner-built UI (web / Slack / Teams) or backend service calls Team OS over HTTPS. | Next.js / React components or REST SDKs |
| Agent | LangChain / LangGraph agent orchestrates prompts, retrieval and tool calls; LLM provider is swappable per tenant. | LangChain, LangServe, LangGraph |
| Tool Façade | MCP (Model Context Protocol) servers expose TMS Global and any partner systems as discoverable tools (`list\_tools()`, `call\_tool()`). | FastAPI / Azure Functions |
| Source Systems | Systems of record accessed ONLY via their MCP server. | TMS Global API, Slack API, HRIS, Stripe, … |

# 2 · Retrieval‑Augmented Generation (RAG)

• Vector Store – Azure Database for PostgreSQL Flexible Server with pgvector (single DB, tenant‑namespaced)  
 – tms\_corpus – static chunks of TMP, QO2, WoWV, LLP‑360 manuals  
 – team-<tenant\_id> – dynamic chunks (each team’s TMP wheel, QO2 scores, manager notes)  
• Blob Storage – Azure Blob (Hot/LRS) holds raw PDFs & exports.  
• Embedding Worker – Timer-triggered Azure Function:  
 1. Detect new blobs or `TMS /profile-pdf/{id}`  
 2. Chunk ≤ 1 k tokens, embed (text‑embedding‑3‑small)  
 3. Insert `{id, tenant\_id, chunk, embedding, metadata.visibility}`  
• Retriever – LangChain SupabaseVectorStore (or AzureAISearchRetriever) filters on both tenant\_id and visibility (blurb vs full).

# 3 · Multi‑Tenant & White‑Label Control Plane

|  |  |
| --- | --- |
| Concern | Implementation |
| API Gateway & Dev Portal | Azure API Management – issues keys, enforces quotas, hosts Swagger docs. |
| Control‑Plane DB | Stores tenant config: brand name, seat cap, allowed diagnostics, preferred LLM, registered MCP endpoints. |
| Dynamic Agent Loader | Injects tenant prompt & brand voice, constrains allowed tool list, selects preferred LLM (Azure OpenAI, Claude, Gemini). |
| Plugin / Tool Registry | Tenants POST their own MCP server URLs; agent discovers tools at runtime via list\_tools(). |
| Usage Metering | LangSmith traces + Azure Monitor logs funnel into per-tenant billing reports or sublicence statements. |

# 4 · Paywall & Data‑Leak Guardrails

|  |  |
| --- | --- |
| Guard | Mechanism |
| Data Gate | Full reports live only in report\_full namespace. Retriever adds subscription.tier == "paid" filter; free users see only report\_blurb. |
| Reasoning Gate | PaywallGate callback inspects the LLM draft—if > N tokens originate from report\_full and user is not entitled, replace with upsell message. |
| MCP Tool Gating | Tools like download\_qo2\_pdf are disabled unless entitlement check passes; MCP returns HTTP 403 otherwise. |
| Defence-in-Depth | Even if the model hallucinates gated content, the callback strips it; direct API calls are still blocked by MCP. |

# 5 · End‑to‑End Flow – “Assess Risk” (QO₂)

1. User → “We need to assess the team’s approach to risk.”  
2. Agent → tmsMCP.create\_qo2\_workflow(teamId)  
3. TMS Global emits progress webhooks (40 % → 70 % → 100 %) → tmsMCP converts to MCP stream events (SSE) → LangGraph updates state.  
4. Analysis Node computes risk spread; if outliers > ±5, drafts nudges.  
5. Agent → slackMCP.send\_nudge(member\_id, message) weekly until spread stabilises.

# 6 · Azure‑Native Service Mapping

|  |  |
| --- | --- |
| Concern | Azure Service |
| Frontend | Azure Static Web Apps |
| Agent Runtime | Azure Functions (Python) or Azure Container Apps |
| Vector Store | Azure Database for PostgreSQL + pgvector or Azure AI Search |
| Blob Storage | Azure Blob Storage (Hot/LRS) |
| Queue / Retry | Azure Service Bus |
| Secrets | Azure Key Vault with Managed Identity |
| Observability | Azure Monitor + Application Insights; LangSmith overlay |
| API Gateway | Azure API Management (multi‑tenant keys, quotas) |

# 7 · Reference Sequence Diagram (textual)

Partner UI ──HTTP──▶ API Gateway ──▶ LangGraph Agent  
 │ │  
 │ ├─ list\_tools() ─▶ tmsMCP  
 │ └─ call\_tool(create\_qo2\_workflow) ─▶ tmsMCP ─▶ TMS Global  
 │ ◀─ progress SSE ◀─────────────────────────────┘  
 │◀───────────────── Streamed analysis & nudges ──────────────────────────────────┘

## Summary

Team OS exposes one headless, agentic API: partners hit /v1/agent/chat, the LangChain agent loads tenant‑scoped prompts, tool lists, and vector context, calls MCP‑wrapped functions, and streams grounded, policy‑safe answers—without ever leaking a full PDF unless the subscription tier allows it. The architecture is unchanged whether the backend runs on Vercel/Supabase or the full Azure stack above, and any LLM (OpenAI, Claude, Gemini) can be swapped simply by updating the tenant’s preferred\_llm flag in the control plane.