Short answer: **MCP isn’t required for GraphRAG**, but it’s a **useful integration layer** in a few places—especially in an Azure stack—if you want a standard, governed way for agents/LLMs to reach internal systems (Cosmos DB, Azure AI Search, SharePoint, etc.). Think of it as a pluggable “USB-C” for tools; your core graph + retrieval can stay exactly as designed.

Here’s how I’d use (or skip) MCP for this UAE legal project:

**Where MCP helps (recommended)**

* **Agent toolbelt, safely:** If you plan an *agentic* reviewer/drafter that must call multiple backends (Cosmos DB graph, Azure AI Search, Gazette store, case-law DBs), MCP gives you a standard tool interface and a growing ecosystem of servers—including community servers for **Azure Cosmos DB**—you can host inside your VNet. Azure now supports connecting **Azure AI Foundry Agent Service** to remote MCP servers (preview).
* **Azure-native integration path:** Microsoft has published guidance and samples for **integrating MCP with Azure OpenAI/Agent Service** and an **Azure MCP Server** to expose Azure services as tools. That means you can keep everything in UAE regions and apply Entra/Key Vault/Monitor controls.
* **Developer & analyst productivity:** MCP already runs in popular clients (e.g., Claude Desktop), and Microsoft is adding MCP support in Windows/Foundry—handy for internal legal analysts who want desktop agents that can securely reach your corp data.

**Where I would not put MCP (keep direct SDKs)**

* **Hot retrieval path:** Your **GraphRAG retrieval** (AI Search → graph traversal → packaging) should use direct SDKs/queries for latency and control. Use MCP **around** the edges (agent tools, admin utilities), not in the tight loop that serves queries to judges or drafters. (Protocol hops add overhead for little gain here.)

**A pragmatic plan (Azure-first)**

1. **Core stays the same:** Cosmos DB (Gremlin) for the bi-temporal legal graph; Azure AI Search hybrid index; Azure OpenAI for generation. (No MCP required.)
2. **Add MCP as a thin “adapter” layer for agents**, where helpful:
   * **Cosmos DB MCP server** for read-only graph queries (there are community and Microsoft samples you can harden).
   * **Azure AI Search MCP server** (or an Agent Service tool) for scoped legal text search.
   * Optional MCP servers for SharePoint/Fabric if you later ingest internal briefs or memos.
3. **Wire to Azure AI Foundry Agent Service (preview)** so your sanctioned agents can call those MCP tools under Entra identities and policy.

**Security checklist (non-negotiable if you adopt MCP)**

* **Private, allow-listed servers only;** keep them inside your VNet with Private Link and Entra app registrations (no public registries).
* **Read-only tools for production** (especially for graph/search); use separate write tools in a gated admin workspace.
* **Bind per-tool scopes** (jurisdiction, tenant, time window) and log every call.
* Follow MCP’s own **security best-practices** (tool list tampering, event streams, auth) and your Azure security baseline (network egress control, key management, RBAC).
* Be aware of **MCP threat models** (tool spoofing/registry abuse); keep a signed internal registry and explicit trust policy.

**When to prioritize MCP**

* You foresee **multiple agents** (drafting assistant, conflict auditor, compliance checker) sharing the same connectors.
* You want a **plugin marketplace** internally (teams can add approved servers without code changes in the agent).
* You plan to let vetted **desktop clients** (e.g., Claude Desktop) reach gov data with least privilege. Ecosystem momentum from Microsoft/Windows suggests this will age well.

**Bottom line**

* **Not required** for the graph/RAG core.
* **Recommended** as a **governed connector layer** for agentic workflows and cross-team reuse—especially in Azure, where Agent Service and sample **Cosmos DB MCP servers** already exist. Start without it in the POC; add it in Phase 2 when you formalize the agent toolbelt and need standardized, auditable connectors.