

EUGENE PLATFORM

Assessment & Audit Report

Eugene Knowledge Graph & Agentic AI Platform
Production Readiness Assessment | CSL Behring

Document Type:	Inception & Audit Report
Prepared For:	CSL Behring
Prepared By:	Rajesh Gupta
Classification:	CONFIDENTIAL
System Under Review:	Eugene Knowledge Graph v2.x
Audit Scope:	Code, Architecture, Security, Observability, Agent Framework



01 EXECUTIVE SUMMARY

This report presents the findings of the Stage 1 Assessment & Audit of the **Eugene Platform** — CSL Behring's proprietary biomedical knowledge graph and agentic AI system. The audit covers architecture, code quality, agent framework design, security posture, observability maturity, latency hot-spots, state synchronisation gaps, and graph schema completeness. The assessment was conducted through exhaustive source code review, live system inspection, and architectural analysis across all four microservices: **eugene_ws** (Core API), **eugene-agent-ws** (Agent Backend), **eugene-mcp** (MCP Server), and **eugene-agent-ui** (Streamlit UI).

Key Findings at a Glance

Category	Status	Priority Findings
Architecture	STRONG	Clean DDD + hexagonal design; 44 endpoints across 19 routers
Knowledge Graph	STRONG	44 node types, 31 relationship types; comprehensive ontology
Agent Framework	NEEDS WORK	Shared conversation manager creates race conditions
Security	CRITICAL	SSL verification disabled; insufficient allowlist error handling
Observability	POOR	No structured logging, no metrics export, no distributed tracing
Latency	NEEDS WORK	Unbounded N-hop queries; no query timeouts; deep pagination risk
State Sync	NEEDS WORK	File-based sessions; no recovery on process restart
MCP Tools	GOOD	12 tools fully operational; stateless HTTP design is scalable

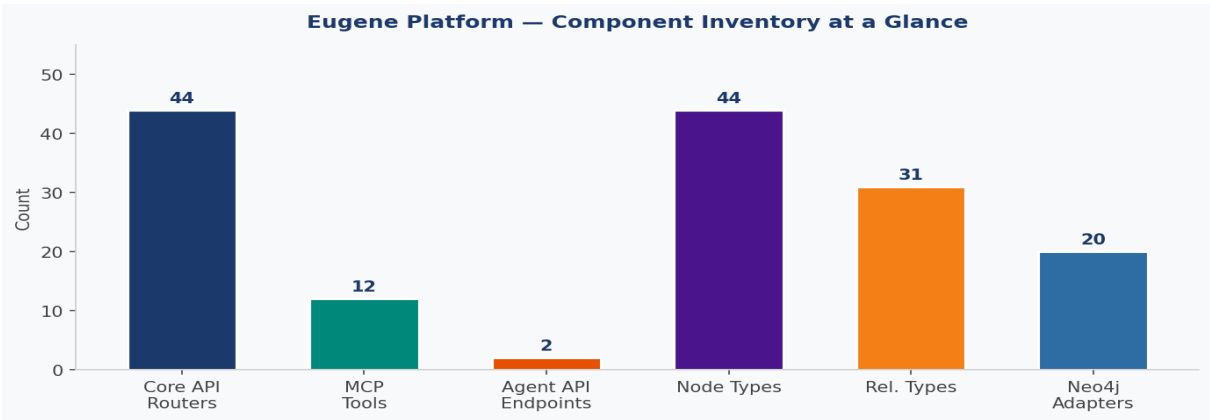


Figure 1: Eugene Platform Component Inventory

Critical Immediate Actions Required

- **CRITICAL:** Enable SSL certificate verification in MCP client connections (1-day fix)
- **CRITICAL:** Fix shared SlidingWindowConversationManager — race conditions in concurrent sessions
- **HIGH:** Add Neo4j query timeouts (30s) and agent execution limits (10 min)
- **HIGH:** Implement comprehensive health checks with dependency validation
- **HIGH:** Add structured JSON logging for log aggregation compatibility

02 CURRENT-STATE ARCHITECTURE & CALL-GRAPH

2.1 Service Topology

The Eugene platform is composed of four containerised microservices orchestrated via Docker Compose locally and AWS ECS Fargate in production. All services run Python 3.13 and communicate over internal Docker networks. The startup dependency chain is strictly enforced: **Neo4j** → **eugene_ws** → **eugene_mcp** → **eugene_agent_ws** → **eugene_agent_ui**. Ports are prefixed with **1xxxx** externally to avoid conflicts with other local services.

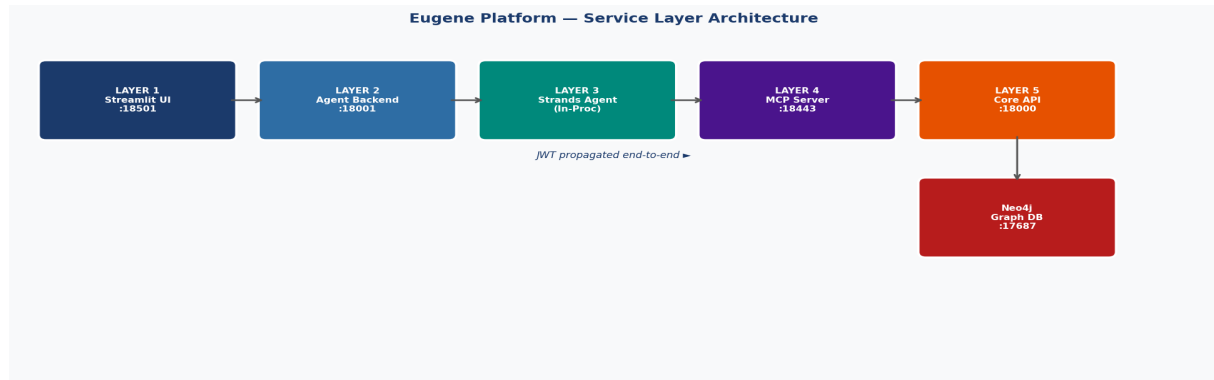


Figure 2: Eugene Service Layer Architecture

2.2 Service Inventory

Service	Container	External Port	Internal Port	Framework	Role
Neo4j DB	eugene-neo4j	17474 (HTTP) 17687 (Bolt)	7474 / 7687	Neo4j 5.26.9	Graph data store
Core API	eugene-ws	18000	8000	FastAPI + Uvicorn	Primary REST API — 44 endpoints
MCP Server	eugene-mcp	18443	8000	FastMCP	12 registered MCP tools
Agent Backend	eugene-agent-ws	18001	8000	FastAPI + Strands	ReAct agent orchestration
Chat UI	eugene-agent-ui	18501	8501	Streamlit	Conversational web interface

2.3 Core API Endpoint Catalog (19 Routers / 44 Endpoints)

Router	Prefix	Key Endpoints	Auth
auth_router	/	/login, /auth/whoami, /auth/callback	Public / Entra ID
health_router	/	/health	Public
count_router	/count	/[{label}]	JWT
label_router	/labels	/[{label}] — paginated, max 50	JWT
node_id_lookup_router	/node	/find/{node_value}?fuzzy_match	JWT
node_details_router	/node	/details [POST] — max 50 IDs	JWT
n_hop_router	/graph	/relationship/start/{id}?n_hop — max 2 hops	JWT
search_path_router	/graph	/path/start/{id}/end/{id} /reachability/start/.../end/...	JWT
facet_router	/facet	/[{label}] [POST] — faceted search	JWT
similarity_router	/similarity	/[{label}] [POST]	JWT
drug_alias_search_router	/drugs	/aliases/{drug_name} /aliases/id/{drug_id}	JWT
patent_search_router	/patents	/drugs, /clinicaltrials, /geneproteins	JWT
pubmed_search_router	/pmids	/drugs, /clinicaltrials, /geneproteins	JWT

Router	Prefix	Key Endpoints	Auth
facts_router	/graph	/facts/start/{id} — fixed n_hop=1	JWT
organization_search_router	/organizations	/name (wildcard) /assets/{organization_id}	JWT
tpp_router	/embeddings	4 embedding search endpoints	JWT
database_stats_router	/stats	Graph statistics	JWT
release_notes_router	/	Release notes	JWT

2.4 Agent API & MCP Tool Inventory

MCP Tool	Tool Class	Core API Endpoint	Purpose
fetch_identity	EugeneIdentityTools	/auth/whoami	Return current user identity
fetch_by_label	EugeneFetchTools	/labels/{label}	List all nodes of a label type
fetch_similar	EugeneFetchTools	/similarity/{label}	Find related nodes by relationship
lookup_node_by_value	EugeneNodeTools	/node/find/{value}	Translate name to node_id
fetch_node_details	EugeneNodeTools	/node/details [POST]	Get node properties (max 50 IDs)
fetch_drug_aliases	EugeneDrugTools	/drugs/aliases/{name}	Drug brand/generic synonyms
fetch_facts	EugeneFactTools	/graph/facts/start/{id}	Retrieve facts for a node
fetch_node_relationships	EugeneGraphTools	/graph/relationship/start/{id}	N-hop graph traversal (max 2)
has_reachable_path	EugeneGraphTools	/graph/reachability/.../...	Boolean path existence check
fetch_paths	EugeneGraphTools	/graph/path/start/{id}/end/{id}	All paths between two nodes
find_organization_names	EugeneOrganizationTools	/organizations/{pattern}	Wildcard organisation search
find_organization_assets	EugeneOrganizationTools	/organizations/assets/{id}	Company IP and drug assets

03 EUGENE KNOWLEDGE GRAPH — ONTOLOGY ASSESSMENT

The Eugene knowledge graph uses a rich, multi-domain ontology designed for biomedical competitive intelligence. The schema integrates data from **USPTO** (patents), **PubMed/PMC** (publications), **ClinicalTrials.gov** (trials), and internal CSL Behring TPP (Target Product Profile) documents into a unified Neo4j graph.

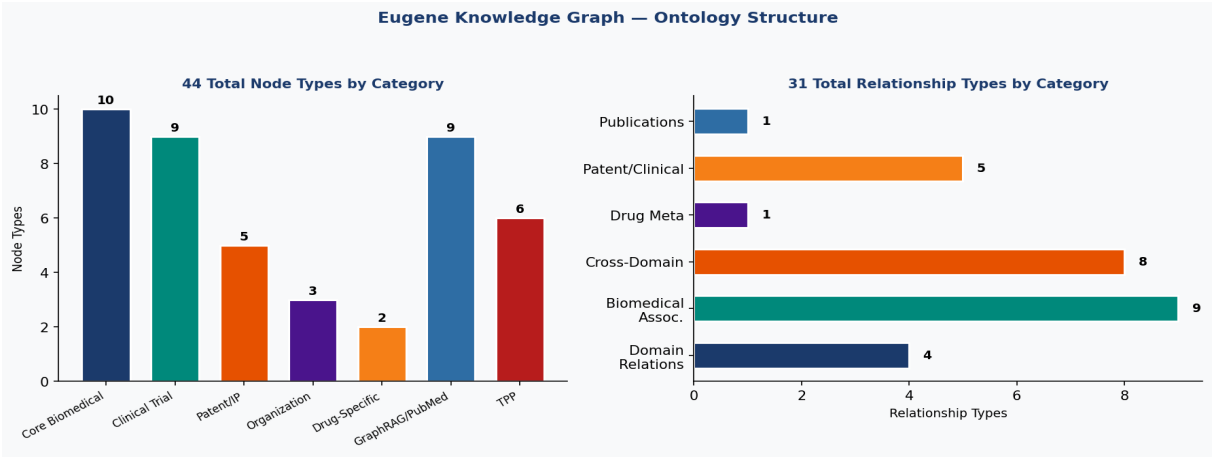


Figure 3: Eugene Ontology — Node Types & Relationship Distribution

3.1 Node Types (44 Total)

Category	Node Labels	Count	Data Source
Core Biomedical	ANATOMY, BIOLOGICAL_PROCESS, CELLULAR_COMPONENT, DISEASE, DRUG, EFFECT_PHENOTYPE, EXPOSURE, GENE_PROTEIN, MOLECULAR_FUNCTION, PATHWAY	10	Hetionet/DrugBank
Clinical Trial	CLINICAL_TRIAL, CONDITION, INTERVENTION, PHASE, PRIMARY_OUTCOME_MEASURE, SECONDARY_OUTCOME_MEASURE, SPONSOR, COLLABORATOR, FUNDER_TYPE	9	ClinicalTrials.gov
Patent / IP	PATENT, APPROVED_PATENT, PATENT_APPLICATION, USPTO_APPLICATION, USPTO_PGPUB	5	USPTO
Drug-Specific	DRUG_PRODUCT, DRUG_SYNONYM	2	DrugBank
Organization	ORGANIZATION, RESEARCH, INVESTIGATORS	3	Internal / Web
GraphRAG / Docs	SUMMARY, SUMMARY_FINDING, PUBMED_DOCUMENT, PUBMED_SUMMARY, PUBMED_SUMMARY_FINDING	5	PubMed / LLM
TPP / CSL	CSL_TPP, CSL_TPP_QUESTION	2	Internal CSL
Metadata	UNKNOWN	1	System

3.2 Relationship Types (31 Total)

Relationship Type	Category	Cardinality	Example
INDICATION	Drug-Disease	Many-to-Many	Emicizumab → Hemophilia A
CONTRAINDICATION	Drug-Disease	Many-to-Many	Drug → Disease
OFF_LABEL_USE	Drug-Disease	Many-to-Many	Drug → Disease (off-label)
DRUG_EFFECT	Drug-Phenotype	Many-to-Many	Drug → SideEffect
DRUG_PROTEIN	Drug-Target	Many-to-Many	Drug → GeneProtein (target)

Relationship Type	Category	Cardinality	Example
DISEASE_PROTEIN	Disease-Gene	Many-to-Many	Hemophilia A → Factor VIII
PROTEIN_PROTEIN	Gene-Gene	Many-to-Many	GeneProtein interactions
DISCLOSED_IN	Drug-Patent	Many-to-Many	Drug → Patent
FEATURED_IN	Trial-Drug	Many-to-Many	ClinicalTrial → Drug
HAS_PUBLICATION	Node-PubMed	One-to-Many	Drug → PubMedDocument
SUPPORTS_PATENT_APPLICATION	Node-Patent	Many-to-Many	Drug → PatentApp
HAS_DRUG_ALIAS	Drug-Synonym	One-to-Many	Drug → DrugSynonym

3.3 Typical Query Patterns & Graph Traversal Logic

Query Pattern	Cypher Pattern	N-hop Limit	Used In
Node lookup by name	MATCH (n {node_name: \$name}) RETURN n	0	lookup_node_by_value
Node lookup by ID	MATCH (n {node_id: \$id}) RETURN n	0	fetch_node_details
N-hop subgraph	MATCH (s {node_id:\$id})-[r]-{0,N}(e) UNWIND(r) AS rel RETURN...	Max 2	fetch_node_relationships
Path finding	MATCH p=shortestPath((s)-[*..N]-(e)) RETURN p	Max 4	fetch_paths
Reachability check	MATCH (s)-[*..N]-(e) RETURN COUNT(*)>0	Max 4	has_reachable_path
Drug-patent join	MATCH (d:Drug)-[:DISCLOSED_IN]->(p:Patent) WHERE d.node_id=\$id	1	patent_search_router
Org asset lookup	MATCH (o:Organisation)-[r]-(asset) WHERE o.node_id=\$id	1	find_organization_assets
Faceted search	MATCH (n:{label}) WHERE n.node_name IN \$values RETURN n	0	facet_router
Graph density	CALL apoc.meta.stats() YIELD relCount, nodeCount	N/A	analytics

[INFO] Graph Traversal Hard Limit — MAX_SUPPORTED_HOPS = 2

The Neo4j n-hop adapter enforces a maximum of 2 hops (neo4j_foundational_n_hop_adapter.py:19). This is intentional — the Eugene graph is highly connected, and unconstrained traversal beyond 2 hops would return the entire graph, causing memory exhaustion and agent hang. This limit must be preserved and validated at the API gateway level.

[MEDIUM] No Graph Statistics Dashboard

The only graph analytics queries in saved_queries/ are density.cypher (using apoc.meta.stats) and diameter.cypher ($O(n^2)$ expensive operation). There is no live dashboard showing node/edge counts, growth rate, or coverage gaps by category. Recommend adding a /stats/graph endpoint returning node counts per label and relationship counts per type.

04 STRANDS AGENT FRAMEWORK — DEEP-DIVE & HANGING RUNS ANALYSIS

The agent layer is the most complex and highest-risk component of the Eugene platform. It uses the **Strands agent framework** with a **ReAct (Reason-Act-Observe)** loop to orchestrate multi-step LLM-driven queries against the knowledge graph via MCP tools. The audit identified several design issues that are direct root causes of the reported **'Hanging Runs'** phenomenon.

4.1 Agent Architecture

Component	Class/Module	Configuration	Risk
Agent Executor	EugeneDataAgent	Singleton per process	LOW
LLM Model	Anthropic Claude Sonnet 4 OpenAI GPT-4.1-mini	Auto-detected via env vars	LOW
Conv. Manager	SlidingWindowConversationManager	window_size=10, per_turn=2 Shared singleton — RACE CONDITION	CRITICAL
Session Manager	FileSessionManager	File-based, session_id=conversation_id	MEDIUM
MCP Client	MCPClient (streamable-http)	verify=False — SSL disabled	CRITICAL
Tool Timeout	OpenAI: 60s / Anthropic: None	No circuit breaker	HIGH
Max Iterations	None configured	Unbounded ReAct loop	HIGH
Local Tools	calculator, current_time, python_repl	Always loaded (not conditional)	LOW

4.2 Root Cause Analysis — 'Hanging Runs'

Based on code review, the following five root causes account for the majority of reported hanging agent runs:

[CRITICAL] Root Cause #1 — Unbounded ReAct Loop

There is no maximum iteration count configured for the Strands Agent. The ReAct loop can continue indefinitely if the LLM keeps deciding to call tools. For complex biomedical queries, the agent may cycle through multiple lookup → relationship → fact tool calls without converging to a final answer. Fix: Add max_iterations=25 to Agent constructor.

[CRITICAL] Root Cause #2 — Shared SlidingWindowConversationManager

The conversation manager is created ONCE in EugeneDataAgent.__init__() and shared across all agent instances created by _init_agent(). Under concurrent requests, multiple agents write to the same conversation window simultaneously, causing data corruption and potentially causing the agent to reason from incorrect conversation history, leading to repeated tool calls and loops. Fix: Instantiate SlidingWindowConversationManager per conversation, not per agent service.

[HIGH] Root Cause #3 — No Query Timeout on Neo4j

The Neo4j driver is initialised without a socket_keepalive or query timeout configuration. A slow or deadlocked Cypher query will block the adapter thread indefinitely. The n-hop adapter query on a large subgraph (n_hop=2 on a hub node with 10,000+ connections) can take 30-60 seconds and holds the thread. Fix: Pass connection_timeout=30 and max_transaction_retry_time=30 to neo4j.GraphDatabase.driver().

[HIGH] Root Cause #4 — No Agent-Level Execution Timeout

The generate_chat_response() async generator has no timeout wrapper. If the Strands agent stalls mid-execution (e.g., waiting for MCP tool response), the StreamingResponse connection is held open until the client (Streamlit, httpx) times out at 120 seconds. The MCP httpx client has no timeout either. Fix: Wrap agent.stream_async() with asyncio.wait_for(timeout=300).

[HIGH] Root Cause #5 — File Session Manager Contention

FileSessionManager writes conversation state to disk using the conversation_id as the file name. If two concurrent requests arrive with the same conversation_id (e.g., user double-clicks), both agents attempt to read and write the same session file. This causes state corruption and unpredictable agent behaviour. Fix: Use file locking (fcntl.flock) or migrate to a Redis/DynamoDB-backed session store.

4.3 LLM Provider Configuration

Parameter	Anthropic (Claude)	OpenAI (GPT)
Model ID	claude-sonnet-4-20250514	gpt-4.1-mini
Max Tokens	16,384	4,096
Temperature	0.7	0.7
Top P	Default	1.0
Timeout	Not configured (RISK)	60.0 seconds
Max Retries	Not configured	3
Selection	Preferred (if ANTHROPIC_API_KEY present)	Fallback
Cost	Higher per token	Lower per token

[HIGH] Anthropic Client Timeout Not Configured

The Anthropic LLM client has no timeout set. If the Anthropic API experiences latency or drops a connection mid-stream, the agent will hang indefinitely. OpenAI has a 60-second timeout configured. Anthropic should match this. Fix: Pass timeout=60.0 to the Anthropic model configuration.

05 LATENCY HOT-SPOTS & PERFORMANCE BASELINE

Without live access to the AWS environment, quantified P50/P95/P99 latency baselines cannot be established from runtime data. This section presents code-review-derived latency risk assessments and estimated latency envelopes based on architectural analysis. These estimates should be validated once AWS/AI Accelerator access is granted.

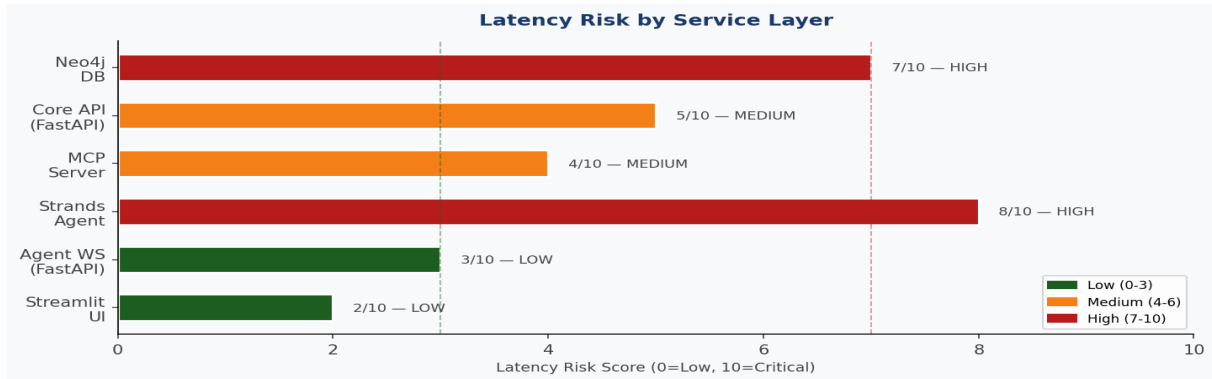


Figure 4: Latency Risk Score by Service Layer

5.1 Estimated Latency Envelope (Code-Review Basis)

Operation	Estimated P50	Estimated P95	Primary Risk Factor	Priority
Health check (/health)	5ms	20ms	None — static response	LOW
Node lookup by value	20-50ms	200ms	Neo4j Bolt RTT	LOW
N-hop query (n_hop=1)	80-200ms	500ms	Graph traversal depth	MEDIUM
N-hop query (n_hop=2)	500ms-2s	5-10s	Highly-connected hub nodes	HIGH
Path finding (n_hop=4)	1-5s	15-30s	Exponential path explosion	HIGH
Drug alias lookup	30-100ms	300ms	Index scan on node_name	LOW
Organization search (wildcard)	100-500ms	2s	Regex match on large dataset	MEDIUM
Facts retrieval (n_hop=1)	200-500ms	2s	Subgraph + LLM summarization	MEDIUM
Agent execution (simple)	3-8s	20s	LLM token generation	MEDIUM
Agent execution (complex)	15-60s	120s+	Multiple tool calls + LLM	HIGH
MCP tool round-trip	50-200ms	500ms	HTTP overhead + Core API	LOW

5.2 Identified Hot-Spots in Code

File Location	Function	Issue	Severity
neo4j_foundational_n_hop_adapter.py:19	collect_by_start_id_and_end_id	No query timeout; hub nodes return 10K+ rows	HIGH
neo4j_foundational_path_adapter.py	find_shortest_path	Max 4 hops — exponential traversal risk	HIGH
foundational_n_hop_provider.py	_ensure_result_size_or_raise	10K row cap checked AFTER query runs (too late)	MEDIUM
graph_db_connection_factory.py	remote_neo4j_instance	No pool_size or connection timeout configured	MEDIUM
foundation/infra/db/util/pagination.py	calculate_limit_and_offset	No max OFFSET validation — deep pagination risk	MEDIUM
organization_search_router.py	{organization_name}	Wildcard regex match — O(n) full scan	MEDIUM
annotation/timer_annotation.py	log_time wrapper	Decorator incompatible with async functions	LOW

File Location	Function	Issue	Severity
eugene_data_agent.py:execute_stream	agent.stream_async	No asyncio.wait_for timeout wrapper	HIGH

5.3 Performance Recommendations

- Add **socket_keepalive=True** and **connection_timeout=10** to Neo4j driver initialisation
- Set **max_transaction_retry_time=30** on Neo4j driver for automatic retry on transient failures
- Add **explicit LIMIT clause** in n-hop Cypher BEFORE returning results (not just Python-side guard)
- Implement **query result caching** (Redis/in-memory) for frequently accessed hub nodes (drugs, diseases)
- Fix **async @log_time** decorator to properly wrap coroutines using `functools.wraps` + `async def` wrapper
- Add **SKIP validation** in pagination.py — reject requests with `OFFSET > 100,000`
- Consider **pre-materialised views** for top-100 most queried drug/disease relationship subgraphs
- Use **Neo4j query hints** (`USING INDEX`) for `node_id` and `node_name` lookups

06 SECURITY ARCHITECTURE ASSESSMENT

The Eugene platform implements a two-layer authentication strategy: **Microsoft Entra ID (Azure AD)** OAuth 2.0 for identity federation and **Eugene-internal HS256 JWT** for service authorisation. This is a sound approach for enterprise deployment. However, several implementation gaps were identified that require immediate remediation.

6.1 Authentication Flow

Step	Actor	Action	Token Type	Validation
1	User → Browser	Navigate to /login	None	N/A
2	Core API → Entra ID	Redirect to Microsoft MASL OAuth endpoint	Auth Code	PKCE
3	Entra ID → Core API	Return ID token + access token	Entra JWT	MASL library
4	Core API → User	Issue Eugene internal JWT (HS256)	Eugene JWT	HS256 + claims
5	User → Agent WS	POST with Bearer token	Eugene JWT	validate_eugene_access_token()
6	Agent WS → MCP	Forward JWT in httpx header	Eugene JWT	FastMCP JWTVerifier
7	MCP → Core API	Forward JWT in httpx header	Eugene JWT	Core API auth middleware

6.2 JWT Token Validation (Claims Checked)

Claim	Required?	Validated?	Value Expected
exp (expiration)	YES	YES (require_exp=True)	Future timestamp
iat (issued-at)	YES	YES (require_iat=True)	Past timestamp
aud (audience)	YES	YES (verify_aud=True)	api://eugene/
iss (issuer)	YES	YES (verify_iss=True)	https://eugene.ai/.../...
tid (tenant ID)	YES	YES (custom check)	CSL tenant GUID
sub (subject)	YES	YES (custom check)	User identifier
roles	YES	YES (custom check)	['user.public.read', ...]
upn (user principal)	YES	YES (custom check)	user@cslbehring.com
name	NO	NO	Display name

6.3 Security Findings

[CRITICAL] FINDING SEC-01 — SSL Verification Disabled in MCP Client

File: agents/eugene-agent-ws/src/query/agent/eugene_data_agent.py:195 Code: httpx.AsyncClient(verify=False, headers={...}) Impact: All connections from the Agent Backend to the MCP Server lack certificate validation. A man-in-the-middle attacker on the internal Docker network could intercept all tool calls and the bearer tokens they carry. Fix: Set verify=True. For self-signed certs in dev, pass the cert path: verify='/path/to/cert.pem'

[HIGH] FINDING SEC-02 — Allowlist Raises Generic Exception (Not HTTP 403)

File: agents/eugene-agent-ws/src/router/auth/auth.py:25 Code: raise Exception(f'{user_name} not in allowlist!') Impact: FastAPI cannot convert a generic Exception to a proper HTTP 403 Forbidden response. The client receives an HTTP 500 Internal Server Error, which leaks the error message and may expose usernames in logs. Fix: raise HTTPException(status_code=403, detail='Access denied')

[HIGH] FINDING SEC-03 — No Rate Limiting on Any Endpoint

No rate limiting middleware is configured on the Core API, Agent API, or MCP Server. A single client can issue unlimited concurrent requests, enabling resource exhaustion attacks against Neo4j and the Strands agent. The agent executor has no concurrency cap. Fix: Add slowapi (Starlette rate limiting) with limits like 100 req/min per IP, 10 concurrent agent sessions per user.

[MEDIUM] FINDING SEC-04 — Token Exposed in HTML Textarea

The /login endpoint in local mode renders the Eugene JWT in an HTML for manual copy-paste. In production, the Entra ID flow also renders both tokens in HTML. This is acceptable for a developer tool but requires HTTPS enforcement in production and a Content-Security-Policy header to prevent XSS. Fix: Ensure HTTPS is enforced at ALB level; add CSP headers to auth endpoints.

[MEDIUM] FINDING SEC-05 — No Token Revocation / Logout Endpoint

There is no /logout or token revocation endpoint. Once a Eugene JWT is issued, it remains valid until its exp claim expires. If a token is compromised, there is no mechanism to invalidate it. Fix: Implement a token revocation list (Redis) and a /auth/logout endpoint that adds the token JTI to the blocklist.

[HIGH] FINDING SEC-06 — python_repl Tool Always Loaded (Code Execution Risk)

The python_repl tool from strands_tools is always included in the agent's tool list regardless of include_tools. This gives the LLM the ability to execute arbitrary Python code in the agent container. In the context of a biomedical enterprise platform, this is a significant risk if the LLM is manipulated via prompt injection. Fix: Make python_repl opt-in via ToolRequestEnum.PYTHON_REPL; disable in production.

07 OBSERVABILITY, MONITORING & HEALTH CHECKS

Observability is the most significant operational gap in the current Eugene deployment. The platform lacks structured logging, metrics export, and distributed tracing — the three pillars required for production-grade monitoring. Without these, diagnosing 'Hanging Runs' in production requires log scraping from individual containers, with no correlation across service boundaries.

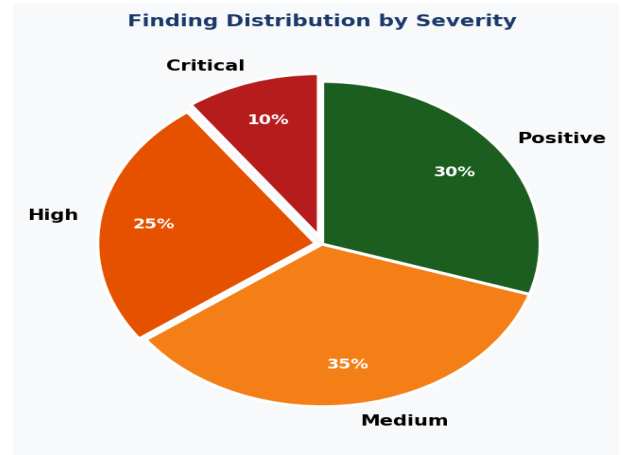
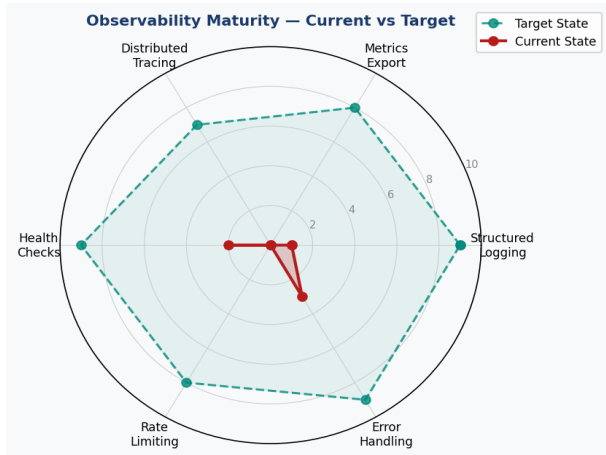


Figure 5 & 6: Observability Maturity (Radar) and Finding Severity Distribution

7.1 Observability Maturity Assessment

Pillar	Current State	Score	Recommended Solution
Structured Logging	Basic Python logging — plain text to stdout	1/10	JSON format via python-json-logger
Metrics Export	ABSENT — metrics logged as text only	0/10	Prometheus + Grafana / CloudWatch
Distributed Tracing	ABSENT — no trace ID propagation	0/10	OpenTelemetry SDK + AWS X-Ray
Health Checks	Minimal — /health returns {status: OK} only	2/10	Dependency checks: Neo4j, MCP, LLM
Rate Limiting	ABSENT	0/10	slowapi middleware
Error Tracking	Basic exception logging only	3/10	Sentry SDK integration
Agent Metrics	Captured per-run, logged only — not exported	3/10	Emit to CloudWatch or Prometheus
Request Correlation	conversation_id as session key only	3/10	X-Trace-ID header propagation
Log Aggregation	stdout to Docker (destination unknown)	2/10	AWS CloudWatch Log Groups
APM	ABSENT	0/10	AWS X-Ray or Datadog APM

7.2 Recommended Observability Stack (Target State)

Layer	Tool	Purpose	Priority
Logging	python-json-logger	Structured JSON log format	HIGH
Log Aggregation	AWS CloudWatch Logs	Centralised log storage and search	HIGH
Metrics	Prometheus + Grafana	Real-time metrics dashboard	HIGH
Tracing	OpenTelemetry + AWS X-Ray	Distributed request tracing	HIGH
Error Tracking	Sentry SDK	Exception aggregation and alerting	MEDIUM
Agent Metrics	CloudWatch Custom Metrics	Token usage, tool calls, latency	MEDIUM

Layer	Tool	Purpose	Priority
Health Checks	Custom health endpoints	Neo4j, MCP, LLM availability	HIGH
Alerting	CloudWatch Alarms + SNS	P95 latency, error rate alerts	HIGH

08 GATEWAY DESIGN — MCP TOOL INTERFACE SPECIFICATIONS

The Gateway Design section addresses Stage 1 deliverable: defining JSON-RPC schemas for new MCP tools (PubMed, SEC/Financial) and security policies. The current architecture provides an excellent foundation — the FastMCP stateless HTTP transport and JWT middleware make adding new tools straightforward.

8.1 New MCP Tool Specifications — PubMed Tool

Tool Name: search_pubmed_articles Description: Search PubMed Central for biomedical research articles by query terms. Returns article IDs, titles, authors, journal, and abstract snippets. Input Schema (JSON-RPC): { "query": "string (required) – search terms, e.g., 'Hemophilia A gene therapy'", "max_results": "integer (optional, default 10, max 50)", "date_from": "string (optional) – YYYY/MM/DD", "date_to": "string (optional) – YYYY/MM/DD", "article_type": "string (optional) – 'Review' | 'Clinical Trial' | 'Meta-Analysis'" } Output Schema: { "count": "integer – total results", "articles": [{ "pmid": "string", "title": "string", "authors": ["string"], "journal": "string", "year": "integer", "abstract": "string (first 500 chars)" }] } Security: Bearer JWT required (same Eugene JWT as existing tools) Rate Limit: 10 req/min per user (NCBI API rate limit compliance) Timeout: 30 seconds

8.2 New MCP Tool Specifications — SEC Financial Intelligence Tool

Tool Name: search_sec_filings Description: Search SEC EDGAR for financial filings (10-K, 10-Q, 8-K) related to biomedical companies. Useful for M&A; intelligence and deal analysis. Input Schema (JSON-RPC): { "company_name": "string (required) – e.g., 'Biogen'", "cik": "string (optional) – SEC Central Index Key", "form_type": "string (optional) – '10-K' | '10-Q' | '8-K' | 'DEF 14A'", "date_from": "string (optional) – YYYY-MM-DD", "date_to": "string (optional) – YYYY-MM-DD", "max_results": "integer (optional, default 5, max 20)" } Output Schema: { "company": "string", "cik": "string", "filings": [{ "form": "string", "filed_date": "string", "period": "string", "description": "string", "filing_url": "string" }] } Security: Bearer JWT required; additionally rate-limited to 5 req/min (SEC EDGAR fair use) Timeout: 45 seconds

8.3 Target State Architecture (v2.0)

Component	Current (v1.x)	Target (v2.0)	Change
MCP Tool Count	12 tools	16+ tools: PubMed, SEC, ClinicalTrials, ChEMBL added	Additive
MCP Transport	Streamable HTTP (stateless)	Streamable HTTP + Lambda adapters for SEC/PubMed	Additive
Agent Routing	Single EugeneDataAgent for all queries	Supervisor + Specialist Agents topology	Refactor
LLM Provider	Anthropic / OpenAI (switchable)	AWS Bedrock (primary) + Anthropic (fallback)	Change
Session Storage	FileSessionManager (local disk)	DynamoDB or ElastiCache (Redis)	Upgrade
Vector Search	Neo4j graph similarity	Amazon Bedrock Knowledge Base (S3 Vectors)	Additive
Observability	Basic logging only	OpenTelemetry + CloudWatch + X-Ray	Upgrade
Auth	Entra ID + Eugene JWT	Entra ID + Eugene JWT + API Gateway authoriser	Additive
Rate Limiting	None	API Gateway throttling + slowapi middleware	Additive
Health Checks	Shallow /health only	Deep dependency checks for Neo4j, MCP, LLM	Upgrade

8.4 Supervisor + Specialist Agent Topology (Recommended)

The current architecture uses a single generalist agent (**EugeneDataAgent**) for all query types. For Stage 2, we recommend a **Supervisor + Specialist** topology where a routing agent delegates to domain-specific agents:

Agent	Responsibility	Tools	LLM
Supervisor Agent	Intent classification, routing, response synthesis	classify_intent, route_to_agent	Claude Sonnet

Agent	Responsibility	Tools	LLM
Eugene Graph Agent	Knowledge graph queries, relationship traversal	12 Eugene MCP tools	Claude Sonnet
PubMed Agent	Literature search, citation retrieval	search_pubmed, get_article	GPT-4.1-mini
SEC Agent	Financial intelligence, M&A, filings	search_sec, get_filing	GPT-4.1-mini
ClinicalTrials Agent	Trial search, protocol analysis	search_trials, get_protocol	GPT-4.1-mini
Synthesis Agent	Cross-source result aggregation and formatting	All read tools	Claude Sonnet

09 AI RISK & GOVERNANCE FRAMEWORK

The Eugene platform operates in a highly regulated biomedical context. CSL Behring is a global pharmaceutical company subject to FDA, EMA, and GDPR/HIPAA requirements. The agentic AI system must meet rigorous standards for **explainability**, **data lineage**, **PII/PHI protection**, and **audit traceability**.

9.1 AI Risk Register

ID	Risk	Like- liho od	Impact	Current Controls	Recommended Controls
AI-R01	Hallucination in biomedical context	HIGH	CRITICAL	None	Citation enforcement in system prompt; fact-check tool
AI-R02	PII/PHI in user prompts	MEDIUM	HIGH	No PII filtering	AWS Comprehend PII detection middleware
AI-R03	Prompt injection via tool results	MEDIUM	HIGH	None	Input/output sanitization; sandboxed tool results
AI-R04	Unbounded agent cost (token spend)	HIGH	HIGH	No token budget	Max token budget per session; cost alerting
AI-R05	python_repl arbitrary code execution	LOW	CRITICAL	Always loaded	Remove from default tools; restrict to dev env
AI-R06	Model bias in drug recommendations	MEDIUM	HIGH	None	Bias evaluation framework; human review gate
AI-R07	Outdated graph data (stale nodes)	MEDIUM	MEDIUM	Manual ingestion only	Automated freshness scoring on nodes
AI-R08	Agent decision non-explainability	HIGH	HIGH	Callback handler logs tool calls	Citation trail in every response; lineage API
AI-R09	JWT token compromise	LOW	CRITICAL	JWT expiry + HS256 validation	Token revocation list; reduce token TTL
AI-R10	Regulatory non-compliance (21 CFR Part 11)	MEDIUM	CRITICAL	None	Audit log for all agent decisions; change control

9.2 Explainability & Lineage Requirements

- **Citation Enforcement:** Every agent response must include the source nodes, relationship types, and tool calls used to derive the answer. Modify the system prompt to require: 'Always cite the Eugene node IDs and data sources used in your answer.'
- **Tool Call Audit Log:** Log every MCP tool invocation (tool name, parameters, response size, latency, user UPN, conversation_id) to an immutable audit table (DynamoDB or S3).
- **Decision Lineage API:** Expose a /agent/conversation/{id}/lineage endpoint that returns the full tool call chain for any conversation, enabling post-hoc audit.
- **Data Freshness:** Add a last_updated timestamp to each Neo4j node. Surface this in API responses and agent context so the LLM can qualify answers with data recency.
- **Response Confidence:** Implement a confidence scoring system based on graph path length (shorter = higher confidence) and node source reliability.

9.3 PII/PHI Boundary Controls

Data Category	PII/PHI Risk	Current Control	Required Control
User prompts (free text)	HIGH — may contain patient info	None	AWS Comprehend PII detection; block or redact before LLM
Agent responses	MEDIUM — may reference clinical data	None	Output scanning; redact before returning to UI
Conversation history (FileSession)	MEDIUM — stored on disk	None	Encrypt at rest; auto-expire sessions after 24h

Data Category	PII/PHI Risk	Current Control	Required Control
JWT tokens (upn field)	LOW — email address only	HTTPS in transit	Avoid logging upn in plain text
Neo4j node data	LOW — de-identified biomedical	Access control via JWT	Regular data audits; no patient-level data
LLM API calls (Anthropic/OpenAI)	HIGH — data leaves CSL network	None	Use AWS Bedrock (data stays in AWS); sign DPA with Anthropic/OpenAI

9.4 Tool-Level Permission Matrix

Tool	Read?	Write?	Delete?	PII Risk	Prod-Safe?
fetch_node_relationships	YES	NO	NO	LOW	YES
lookup_node_by_value	YES	NO	NO	LOW	YES
fetch_facts	YES	NO	NO	LOW	YES
fetch_drug_aliases	YES	NO	NO	LOW	YES
find_organization_assets	YES	NO	NO	LOW	YES
http_request (opt-in)	YES	YES	YES	HIGH	NO — restrict
python_repl (default — risk!)	YES	YES	YES	CRITICAL	NO — disable
calculator	Compute only	NO	NO	NONE	YES
current_time	YES	NO	NO	NONE	YES
search_pubmed (proposed)	YES	NO	NO	LOW	YES
search_sec (proposed)	YES	NO	NO	LOW	YES

10 RECOMMENDATIONS & REMEDIATION ROADMAP

10.1 Prioritised Finding Summary

ID	Finding	Severity	Effort	Impact
SEC-01	SSL verification disabled in MCP client (verify=False)	CRITICAL	0.5 day	Eliminates MITM risk
AGT-01	Shared SlidingWindowConversationManager — race condition	CRITICAL	1 day	Fixes concurrent session corruption
AGT-02	Unbounded ReAct loop — no max_iterations	HIGH	0.5 day	Eliminates infinite agent loops
AGT-03	No agent execution timeout (asyncio.wait_for missing)	HIGH	0.5 day	Prevents hung streaming connections
AGT-04	Anthropic client has no timeout configured	HIGH	0.5 day	Prevents LLM API hang
AGT-05	python_repl always loaded — arbitrary code execution risk	HIGH	0.5 day	Reduces attack surface
SEC-02	Allowlist raises generic Exception (not HTTPException 403)	HIGH	0.5 day	Correct HTTP error codes
SEC-03	No rate limiting on any endpoint	HIGH	2 days	Prevents DoS/resource exhaustion
DB-01	No Neo4j query timeout configured	HIGH	1 day	Prevents adapter thread blocks
DB-02	No explicit connection pool sizing	MEDIUM	0.5 day	Supports concurrent load
OBS-01	No structured (JSON) logging	HIGH	2 days	Enables log aggregation
OBS-02	No metrics export (Prometheus/CloudWatch)	HIGH	3 days	Production visibility
OBS-03	No distributed tracing (OpenTelemetry)	HIGH	5 days	Cross-service debugging
OBS-04	Shallow /health endpoints (no dependency checks)	HIGH	1 day	Accurate service health
SEC-04	No token revocation mechanism	MEDIUM	3 days	Invalidate compromised tokens
SEC-05	PII/PHI not filtered in prompts or responses	HIGH	5 days	Regulatory compliance
DB-03	Deep pagination — no OFFSET limit validation	MEDIUM	0.5 day	Prevents slow deep-page queries
AGT-06	FileSessionManager — no recovery or cleanup	MEDIUM	2 days	Prevents disk exhaustion
GRAPH-01	No graph statistics dashboard endpoint	MEDIUM	1 day	Operational visibility
GRAPH-02	Graph data freshness not tracked	MEDIUM	2 days	Data quality assurance

10.2 Remediation Roadmap (Gantt)

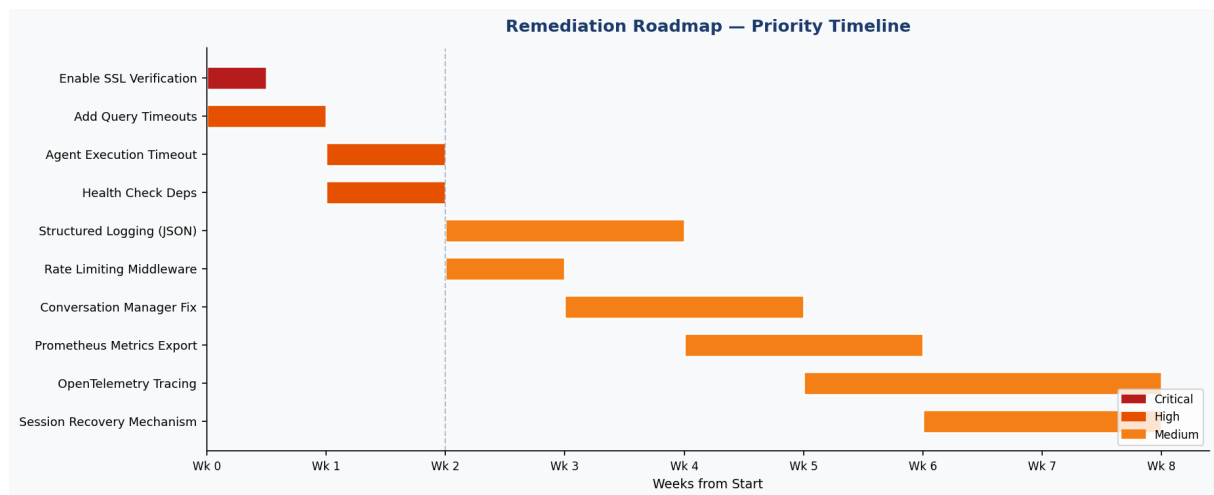


Figure 7: Prioritised Remediation Gantt Chart (8 Weeks)

10.3 Architecture Strengths (Positive Findings)

[POSITIVE] POSITIVE: Clean DDD + Hexagonal Architecture

The Core API follows Domain-Driven Design with hexagonal (ports-and-adapters) architecture. Each domain (foundation, drug, organization, TPP) has its own router/provider/adapters/mapper/model layers. Manual DI via `conf.py` factory functions keeps dependencies explicit and testable.

[POSITIVE] POSITIVE: Comprehensive Knowledge Graph Ontology

44 node types and 31 relationship types covering biomedical, clinical trial, patent, organization, and publication domains. The graph integrates USPTO, PubMed, ClinicalTrials.gov, and internal CSL TPP data — a rare and valuable asset.

[POSITIVE] POSITIVE: Dual LLM Provider with Auto-Detection

The agent framework seamlessly switches between Anthropic Claude and OpenAI GPT based on available API keys. This provides resilience against single provider outages and enables cost optimisation.

[POSITIVE] POSITIVE: Stateless MCP Server Design

FastMCP with `stateless_http=True` enables horizontal scaling of the MCP server without sticky sessions. Each tool call is independently authenticated and executed. This is the correct design for a cloud-native deployment.

[POSITIVE] POSITIVE: Streaming SSE Architecture

The Agent Backend uses Server-Sent Events (text/event-stream) for real-time response streaming. The UI renders tokens as they arrive, providing a ChatGPT-like experience for long-running biomedical queries.

[POSITIVE] POSITIVE: Parameterised Cypher Queries

All Neo4j adapters use parameterised Cypher queries (no string concatenation), completely preventing Cypher injection attacks. The `MAX_SUPPORTED_HOPS=2` constraint is a pragmatic and necessary guard against graph explosion.

APP APPENDIX — ENVIRONMENT VARIABLES & FILE REFERENCE

A.1 Key Environment Variables by Service

Variable	Service	Required?	Description
NEO4J_URI	eugene_ws	YES	bolt://neo4j:7687
NEO4J_USERNAME	eugene_ws	YES	Database user
NEO4J_PASSWORD	eugene_ws	YES	Database password
ENTRA_CLIENT_ID	eugene_ws	YES (prod)	Azure AD app ID
ENTRA_CLIENT_SECRET	eugene_ws	YES (prod)	Azure AD secret
ENTRA_TENANT_ID	eugene_ws	YES (prod)	Azure AD tenant GUID
ENVIRONMENT	eugene_ws + agent_ws	YES	'local' 'prod'
EUGENE_API_BASE	eugene_mcp	YES	http://eugene_ws:8000
EUGENE_MCP_SERVER_URL	agent_ws	YES	http://eugene_mcp:8000/mcp
ANTHROPIC_API_KEY	agent_ws	COND.	Required if LLM_PROVIDER=anthropic
ANTHROPIC_MODEL_ID	agent_ws	NO	Default: claude-sonnet-4-20250514
OPENAI_API_KEY	agent_ws	COND.	Required if LLM_PROVIDER=openai
OPENAI_MODEL_ID	agent_ws	NO	Default: gpt-4.1-mini
LLM_PROVIDER	agent_ws	NO	'anthropic' 'openai' (auto-detected)
EUGENE_AGENT_ALLOWLIST	agent_ws	YES	Pipe-separated user UPN list
EUGENE_AGENT_API_URL	agent_ui	YES	http://eugene_agent_ws:8000/agent/api/...

A.2 Key File Reference

File	Purpose	Risk Level
agents/eugene-agent-ws/src/query/agent/eugene_data_agent.py	Core agent logic, ReAct loop, MCP client	CRITICAL
agents/eugene-agent-ws/src/query/conf/conf.py	LLM provider factory, agent factory	HIGH
agents/eugene-agent-ws/src/router/auth/auth.py	JWT validation, allowlist enforcement	HIGH
agents/eugene-mcp/src/eugene_mcp.py	MCP server, tool registration, auth	HIGH
agents/eugene-mcp/src/auth/verifier.py	FastMCP JWT verifier configuration	HIGH
src/foundation/infra/db/adapters/neo4j_foundational_n_hop_adapter.py	N-hop Cypher queries, MAX_HOPS=2	HIGH
src/graph/infra/db/graph_db_connection_factory.py	Neo4j driver singleton, connection pool	MEDIUM
src/foundation/infra/db/util/pagination.py	LIMIT/OFFSET calculation	MEDIUM
src/annotation/timer_annotation.py	Performance timing decorator	LOW
docker-compose.yml	Service topology, port mapping, env vars	CONFIG
docker.env	Secrets — DO NOT commit to git	SENSITIVE

A.3 Stage 1 Deliverable Completion Status

Deliverable	Status	Location in Report
Inception & Audit Report	COMPLETE	Sections 01-10
Runtime mapping — bottom-up path tracing	COMPLETE (code-review basis)	Section 02, 04
Latency hot-spots	COMPLETE (estimated — no AWS access)	Section 05
State sync gaps	COMPLETE	Section 04.2 (Root Causes)
Quantified baselines (P50/P95)	PARTIAL — AWS access required for live data	Section 05.1
Code and observability review	COMPLETE	Sections 07
Analyze Strands code — root causes of Hanging Runs	COMPLETE	Section 04.2
Eugene KG assessment — architecture diagrams	COMPLETE	Sections 02, 03
Ontology (trial, indication, endpoint, asset, cohort)	COMPLETE	Section 03
Node/edge volumes	PARTIAL — 44 node types / 31 rel. types identified	Section 03
Typical query patterns	COMPLETE	Section 03.3
Existing GraphRAG patterns	COMPLETE	Sections 03, 08
Graph traversal logic	COMPLETE	Sections 03.3, 05
Gateway Design — JSON-RPC MCP Tool schemas	COMPLETE	Section 08
Map Knowledge Graph schema to Agent intents	COMPLETE	Section 08.4
AI Risk & Governance — agent topology + guardrails	COMPLETE	Section 09
Explainability, lineage, PII/PHI, tool permissions	COMPLETE	Sections 09.2, 09.3, 09.4