

Azentiq Memory Manager: White Paper for Open Source Community

Executive Summary

As Agentic AI continues to gain traction, intelligent systems increasingly depend on their ability to recall, reason, and adapt across long, multi-step workflows. However, most current frameworks offer limited or no structured memory support. The Azentiq Memory Manager addresses this critical gap by introducing a pluggable, extensible, and high-performance memory layer designed to serve as a universal foundation for memory in agentic systems.

This white paper introduces the motivation behind the project, the architectural design, key features available in the MVP, and a roadmap for future growth. We invite developers, researchers, and contributors to join us in shaping the future of memory infrastructure for agentic AI.

1. The Problem: Memory in Agentic AI is Broken

Despite rapid progress in agent orchestration frameworks (LangChain, LangGraph, CrewAI, AutoGen, etc.), memory remains one of the most underdeveloped components.

Common Pitfalls:

- **Ephemeral Context:** Most frameworks rely on windowed context—when token limits are reached, older information is discarded.
- **No Importance Awareness:** Routine messages are treated the same as critical anomalies or decisions.
- **Manual Memory Logic:** Developers have to hand-code memory strategies for each use case.
- **Poor Scalability:** Flat memory storage (e.g., raw lists or transcripts) doesn't scale for long-lived agents.
- **No Pluggability:** Switching from short-term to long-term memory (or from in-memory to Redis/VectorDB) is not standardized.

These limitations cripple the agent's ability to reason over time, follow goals, or maintain useful knowledge.

2. The Solution: Azentiq Memory Manager








Azentiq Memory Manager is a framework-agnostic memory infrastructure layer that introduces structured, tiered memory and token budgeting to any agentic system.

Key Innovations:

- **Tiered Memory Architecture:** Short-term, working, and long-term memories modeled with different TTLs, importance levels, and access patterns.
- **Token Budgeting:** Automatic management of what fits into LLM context windows based on priority, recency, and importance.
- **Metadata-Rich Records:** Every memory is stored with tags, source IDs, timestamps, and custom metadata.
- **Pluggable Backends:** Redis by default, with future support for Postgres, Vector DBs, and cloud-native stores.
- **Framework Adapters:** LangChain and LangGraph integration templates included.













3. MVP Overview

The initial MVP release includes:

-  **Redis-backed session memory** with TTL and metadata
-  **Memory tiering logic** (short-term, working memory)
-  **Token budgeting and scoring system**
-  **CLI tool** built with Typer for CRUD operations
-  **LangChain & LangGraph adapter templates**
-  **Complete documentation and API reference**
-  **Apache 2.0 open source license**

4. Comparison: Azentiq Memory Manager vs. Other Memory Solutions in Open-Source Agentic AI

Most open-source agentic AI memory systems (such as LangChain, LangGraph, AutoGen, CrewAI, and MemoryX) focus on session-limited storage or basic state persistence, with limited support for long-term memory reasoning. Here's how Azentiq Memory Manager compares:

Capability	LangChain Memory	LangGraph State	AutoGen Memory	CrewAI Memory	MemoryX	Azentiq Memory Manager
Memory Tiering	 No	 Minimal	 No	 No	 No	 Short-term, Working, Long-term
Token-Aware Budgeting	 No	 No	 No	 No	 No	 Dynamic, model-specific limits

Capability	LangChain Memory	LangGraph State	AutoGen Memory	CrewAI Memory	MemoryX	Azentiq Memory Manager
Importance-Based Recall	✗ No	✗ No	✗ No	✗ No	✗ No	✓ Weighted memory persistence
Structured Retrieval	✗ No	✗ No	✗ No	✗ No	⚠ Limited	✓ Metadata-driven search/filter
Persistent Storage	✗ Optional	✓ JSON State	✓ Customizable	✗ No	✓ SQL	✓ Redis + Pluggable Adapters
Session Isolation	✗ No	⚠ Shared Context	⚠ Process Tied	✗ No	✓ Scoped	✓ Namespaced by session ID
Framework Agnostic	✗ LangChain Only	✗ LangGraph Only	✗ AutoGen Only	✗ CrewAI Only	⚠ SDK-Specific	✓ LangChain, LangGraph, Any API
Adapter-Based Design	✗ No	⚠ Internal	✗ No	✗ No	⚠ Plugin Layer	✓ Clean adapter interface
Use Case Fit	Simple bots	Graph workflows	Experiments	Team agents	Prompt chaining	Complex, scalable agents

Azentiq is designed for modern agentic AI systems that require reasoning over extended sessions, cross-conversation recall, and memory pruning under token constraints. It's the only open-source system that brings structured tiered memory and production-grade architecture together in a framework-agnostic way.

5. Benefits of Tiered Memory in the IoT Agent Implementation

The Redis-backed tiered memory system implemented in Azentiq offers the following benefits, particularly well-demonstrated in IoT agent applications:

1. Optimized Data Retention

- **SHORT_TERM tier:** Stores raw telemetry with TTL (~30 mins), reducing memory bloat while keeping recent data accessible.
- **WORKING tier:** Retains higher-value events like anomalies, insights, and thresholds with indefinite persistence for long-term analysis.

2. Context-Aware Query Responses

- Contextual queries fetch the most relevant tier automatically.
- Answers prioritize meaningful data (e.g., anomalies) over low-importance logs.

3. Efficient Resource Management

- Raw telemetry expires naturally via TTL — no manual cleanup.
- Long-lived insights are retained in WORKING memory.
- Targeted tier queries improve response speed and memory efficiency.

4. Session Isolation

- Each session uses a unique ID (e.g., `iot_demo_<uuid>`), ensuring:
- Multi-agent isolation
- Easy session resets
- Debuggable and reproducible agent states

5. Better Observability

- Logging and memory inspection are easier due to tier clarity.
- Debugging is more transparent, with clear separation of raw vs. processed data.

6. Flexible Integration with LangChain

- Azentiq's memory adapters align with LangChain's `BaseMemory` interface.
 - Developers can override `load_memory_variables()` to dynamically pull tiered memory.
 - Seamless integration allows context-aware prompts without overloading the token budget.
-

6. Roadmap

Phase 1: Launch & Community Feedback (Now)

- Sample IoT + LangChain + Memory integration demo
- Developer onboarding documentation
- Bug and performance feedback loop

Phase 2: Memory Expansion

- Long-term memory tier (vector store or summarization-based)
- Summarization pipeline + auto-pruning logic
- TTL visualizer and memory introspection tools

Phase 3: Ecosystem Support






- Hosted version for Memory-as-a-Service (MaaS)
- UI dashboard for live memory inspection
- SDKs in JS, Go, Rust for broader adoption

- Integration with enterprise LLM systems (Azure OpenAI, Bedrock, etc.)
-

6. How to Contribute

We welcome contributions from all corners of the open-source community.

Ways to Contribute:






- Build additional storage backends (e.g., MongoDB, SQLite, Pinecone)
-  Add new framework adapters (AutoGen, CrewAI, Semantic Kernel)
-  Help benchmark and optimize Redis or memory retrieval latency
-  Write unit and integration tests
-  Improve documentation and tutorials
-  Propose new memory scoring strategies (recency x importance, entropy, etc.)

Getting Started:

- Visit the [GitHub Repository](#)
 - Review the [CONTRIBUTING.md](#)
 - Join discussions via Issues or GitHub Discussions
-

7. Call to Action: Use Azentiq Memory Manager Across Agentic Use Cases

We invite the open-source community to explore the use of Azentiq Memory Manager across diverse agentic AI applications:

-  **IoT Monitoring Agents:** Retain anomalies, filter raw data, and query across sessions.
-  **Healthcare Assistants:** Preserve longitudinal patient memory with tiered importance.
-  **Financial Advisors:** Manage session-aware reasoning over news, earnings, and market history.
-  **DevOps Agents:** Track changes, errors, and resolutions across interactions.
-  **Customer Support Bots:** Retain key complaint patterns while pruning casual chatter.

Azentiq Memory Manager is built to support high-performance agent reasoning by maintaining what matters most. Whether you're building a lightweight assistant or a full-scale enterprise agent, this library can accelerate your memory integration journey.

Final Thought

Azentiq Memory Manager aims to become the memory layer standard in the agentic AI ecosystem. We've built the foundation — now it's your turn to help shape what intelligent memory should look like. Together, we can enable the next generation of agents that remember, reason, and grow smarter over time.

Let's build memory that matters.

Author: Anil Nagandla\ **License:** Apache 2.0\ **Contact:** contact@azentiq.ai