CRANIMEM: CRANIAL INSPIRED GATED AND BOUNDED MEMORY FOR AGENTIC SYSTEMS

Pearl Mody[1], Mihir Panchal[1], Rishit Kar[1], Kiran Bhowmick[1], Ruhina Karani[1]
[1]Dwarkadas Jivanlal Sanghvi College of Engineering
modypearl05@gmail.com, mihirpanchal5400@gmail.com, emailrishitkar@gmail.com
kiran.bhowmick@djsce.ac.in, ruhina.karani@djsce.ac.in

ABSTRACT

Large language model (LLM) agents are increasingly deployed in long running workflows, where they must preserve user and task state across many turns. Many existing agent memory systems behave like external databases with ad hoc read/write rules, which can yield unstable retention, limited consolidation, and vulnerability to distractor content. We present **CraniMem**, a neurocognitively motivated, gated and bounded multi-stage memory design for agentic systems. CraniMem couples goal conditioned gating and utility tagging with a bounded episodic buffer for near term continuity and a structured long-term knowledge graph for durable semantic recall. A scheduled consolidation loop replays high utility traces into the graph while pruning low utility items, keeping memory growth in check and reducing interference. On long horizon benchmarks evaluated under both clean inputs and injected noise, CraniMem is more robust than a Vanilla RAG and Mem0 baseline and exhibits smaller performance drops under distraction. Our code is available at https://github.com/PearlMody05/Cranimem and the accompanying PyPI package at https://pypi.org/project/cranimem.

1 INTRODUCTION

Large language model LLM agents increasingly operate over long horizons, they plan, execute tools, collaborate with other agents, and interact with users over days or weeks. In these settings, memory is no longer a simple context window, but a system component that must decide what to store, when to retrieve it, and how to keep knowledge consistent across multiple agent roles and tasks. Recent work has proposed long-term memory stacks for agents, for example production oriented persistent stores and indexing pipelines, as well as specialized agentic memory modules for retrieval and reasoning (Chhikara et al., 2025; Xu et al., 2025; Huang et al., 2025; Hu et al., 2025). However, many current designs treat memory as an external database with heuristic write/read policies, which can lead to brittle retention where important events are overwritten by noise, weak consolidation across time, and poor synchronization when multiple agents contribute partially overlapping beliefs (Yuen et al., 2025; Jiang et al., 2025).

Motivated by findings from neuroscience and cognitive science, we argue that agentic memory should be designed as a gated, bounded, and multi-stage process, similar to how biological systems regulate encoding, consolidation, and retrieval across hippocampal cortical pathways (Yang et al., 2025; Dong et al., 2025). Building on this perspective, we introduce **CraniMem**, a neurobiologically inspired memory framework for agentic systems that explicitly separates transient working and episodic traces, consolidated long-term knowledge, and control gates that decide what enters memory, what is rehearsed, and what is exposed to downstream reasoning.

Our key contributions are:

1. We propose **CraniMem**, a gated and bounded multi-stage memory architecture that supports selective encoding, consolidation, and retrieval for long horizon agent behavior.

2. We introduce a dual-store memory architecture with explicit consolidation pathways that separate episodic and semantic memory, enabling rapid adaptation alongside stable knowledge formation without unbounded growth.
3. Through comprehensive evaluation on long horizon benchmarks, we demonstrate that selective forgetting via importance weighting and temporal decay outperforms both unlimited retention and aggressive compression, yielding higher information stability and comparable resistance to distraction.

2 Related Works

2.1 Multi Agent Memory Architectures

A growing body of work studies how to equip LLM agents with persistent memory beyond the prompt window. Systems such as Mem0 focus on production ready long-term memory with scalable storage, retrieval, and update mechanisms (Chhikara et al., 2025). A-Mem formalizes agentic memory as a component that supports writing experiences and retrieving them for downstream planning and decision making (Xu et al., 2025). Lightweight designs like LiCoMemory aim to reduce memory overhead while maintaining long-term reasoning ability (Huang et al., 2025). Broader surveys highlight recurring design axes what to store between facts and experiences, how to index and retrieve, and how to prevent drift or stale information and emphasize the gap between memory as a database and memory as an adaptive cognitive process (Hu et al., 2025; Liu & Shu, 2025).

In multi-agent settings, memory must also address synchronization and heterogeneity. Intrinsic Memory Agents introduce structured contextual memory to coordinate heterogeneous agents and improve collaboration (Yuen et al., 2025). PersonaMem-v2 targets personalization by learning implicit user personas and maintaining long-term user specific memory, which introduces additional challenges around stability and updating user models over time (Jiang et al., 2025). (Wheeler & Jeunen, 2025) argue that procedural memory alone is insufficient for robust agent behavior, motivating richer memory types and control mechanisms .

2.2 Neuroscience Inspired Memory Components

Neuroscience inspired approaches increasingly inform how memory should be organized for LLMs and agents. HippoRAG proposes a long-term memory mechanism driven by neurobiological motivation that draws inspiration from hippocampal indexing for retrieval (Jimenez Gutierrez et al., 2024). Other work explores hippocampal inspired multimodal memory to support long understanding of audiovisual events (Lin et al., 2025). Position papers argue that episodic memory is a key missing ingredient for long-term agent coherence and that agents should store causally linked time stamped experiences rather than only distilled facts (Pink et al., 2025).

Complementary lines of research examine cognitive memory mechanisms in LLMs and agentic frameworks, including how memory representations might be structured, consolidated, and used for reasoning (Shan et al., 2025). Brain inspired multi-memory agent frameworks for embodied learning further motivate separating memory stores and using control policies to regulate encoding and retrieval (Lei et al., 2025). Our work follows this trend, but focuses specifically on gated and bounded memory operations to improve retention and consistency in long horizon agentic systems.

3 CraniMem

CraniMem is a memory component for long horizon LLM agents designed around two neurocognitive principles. First, attentional gating reduces interference by suppressing low-salience inputs before they reach memory, analogous to brainstem and thalamic control systems that regulate access to downstream processing. Second, systems consolidation stabilizes knowledge by transferring selected experiences from a fast episodic store to a slower semantic store through replay.

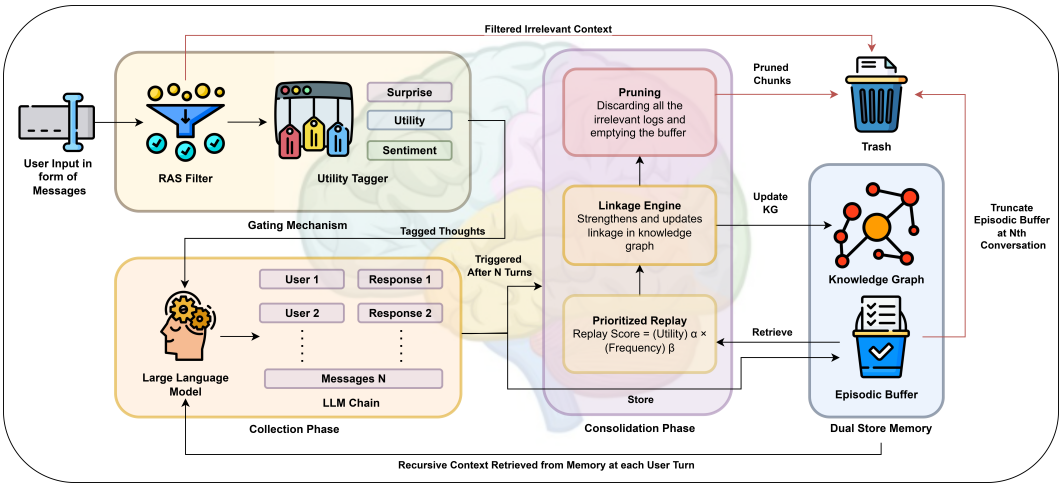

Figure 1: CraniMem architecture. The pipeline implements ingestion via RAS inspired gating and utility tagging, short-term storage in a bounded episodic buffer, optimization through replay selection and pruning to Trash, long-term structural storage via a linkage engine that updates a knowledge graph, and dual-path retrieval using both the buffer and the graph.

Operationally, CraniMem maintains two bounded stores and a periodic optimization loop. Recent interaction traces are stored in an episodic buffer that supports high fidelity short range continuity and provides candidates for consolidation. Durable information such as user preferences, constraints, and stable plans is stored in a structured knowledge graph that supports consistent multi-hop retrieval. A gating module filters candidate writes and assigns utility signals that drive which items are rehearsed. During consolidation, high-utility episodic traces are integrated into the knowledge graph and low utility traces are discarded to keep memory bounded.

Algorithmic details for gating, replay scoring, linkage, and retrieval are provided in Appendix B.

4 Evaluation Protocol

Clean vs. Noisy evaluation In the clean setting, the agent receives only task relevant interaction traces, and the memory write stream contains only those traces. In the noisy setting, we inject distractor snippets into the same write stream to simulate context pollution. Following a fixed injection schedule, we add m irrelevant distractor memories per injection event. The distractors are unrelated to the current goal, but are written to memory using the same write policy as task traces, so they compete with relevant items at retrieval time. We report results under both settings and quantify robustness using the clean and noisy performance gap using the evaluation metrics mentioned in Appendix C.

5 Results

We evaluate our proposed memory architecture, CraniMem, against Vanilla RAG and Mem0 on Qwen2.5-7B-Instruct. Additionally, to assess generalization across LLMs, we compare CraniMem with Vanilla RAG on multiple instruction-tuned models. For each setup, the underlying LLM and decoding configuration are kept fixed, isolating the impact of the memory system. All models are evaluated under clean inputs and under injected distractor noise of HotpotQA Dataset (Yang et al., 2018), following the protocol described in Appendix A and using the prompt templates in Appendix D.

Architecture	Noisy				Clean				Noise Drop Δ_{noise}
	P	R	F1	Latency (s)	P	R	F1	Latency (s)	
Vanilla RAG	0.058	0.339	0.068	58.968	0.092	0.366	0.095	68.146	0.027
Mem0	0.205	0.281	0.198	10.387	0.234	0.262	0.234	2.191	0.036
CraniMem (Ours)	0.315	0.317	0.312	112.440	0.329	0.325	0.323	53.977	0.011

Table 1: Comparison of different architectures on **Qwen2.5-7B-Instruct** under noisy and clean settings on multi-hop HotpotQA dataset.

Table 1 shows that CraniMem performs overall better than Vanilla RAG and Mem0 under both clean and noisy settings, and exhibits the smallest noise drop. This indicates stronger robustness to distractors, with the main cost being higher latency due to CraniMem's more complex memory pipeline that includes the consolidation phase and storage through knowledge graph.

Model	Architecture	Type	Precision	Recall	F1	Noise Drop Δ_{noise}	Latency (s)
Qwen2.5-Coder-7B-Instruct	Vanilla RAG	Noisy	0.100	0.117	0.096	0.079	6.810
		Clean	0.170	0.220	0.175		4.439
	CraniMem	Noisy	0.289	0.279	0.280	0.004	252.915
		Clean	0.294	0.282	0.284		9.029
Gemma-2-9B-IT	Vanilla RAG	Noisy	0.123	0.116	0.116	0.160	61.023
		Clean	0.303	0.268	0.276		44.448
	CraniMem	Noisy	**0.339**	**0.318**	**0.323**	0.015	134.126
		Clean	0.352	0.334	0.338		70.152
Qwen2.5-7B-Instruct	Vanilla RAG	Noisy	0.058	0.339	0.068	0.027	58.968
		Clean	0.092	0.366	0.095		68.146
	CraniMem	Noisy	0.315	0.317	0.312	0.011	**112.440**
		Clean	0.329	0.325	0.323		53.977
Mistral-7B-Instruct-v0.3	Vanilla RAG	Noisy	0.115	0.177	0.118	0.129	5.514
		Clean	0.243	0.353	0.247		7.701
	CraniMem	Noisy	0.300	0.300	0.294	0.022	381.230
		Clean	0.320	0.325	0.316		13.943

Table 2: CraniMem benchmark performance on multi-hop HotpotQA dataset under injected noise, reported across LLM backbones.

Table 2 shows that across multiple instruction-tuned models, CraniMem improves noisy condition performance and substantially reduces noise drop compared to Vanilla RAG. For example, with Gemma-2-9B-IT, CraniMem achieves the best noisy F1 of 0.323 and a noise drop of 0.015, while the Vanilla RAG baseline exhibits a larger degradation with noise drop 0.160. Similar patterns hold for Qwen2.5-Coder-7B-Instruct, where CraniMem has near zero noise drop 0.004 versus 0.079 for Vanilla RAG, indicating that goal directed gating prevents distractors from entering memory and protects retrieval quality. Qwen2.5-7B-Instruct yields the lowest CraniMem latency among the tested backbones at 112.44 seconds in the noisy setting, while Qwen2.5-Coder-7B-Instruct and Mistral-7B-Instruct-v0.3 incur higher overhead. These results suggest that CraniMem is most beneficial in settings where distraction resistance and information stability are prioritized over raw throughput.

6 Limitations and Conclusion

In this paper, we introduce Cranimem, a neurocognitively inspired agentic memory architecture that supports encoding, consolidation, and retrieval for long horizon agent behavior. We compare our architecture against baselines on a similar architecture of Mem0 and Vanilla RAG. CraniMem consistently outperforms both baselines, showing stronger robustness to distractors and more stable long horizon recall. This improved robustness comes with a deployment trade-off, CraniMem incurs

higher latency. Due to computational constraints, our experiments were conducted on 100 sampled instances, which limits the statistical strength of the reported numbers. Future work will expand the empirical study to larger evaluation sets and perform systematic comparisons and ablations against other memory architectures, including HippoRAG and Mem0, under standardized engineering and evaluation settings. For reproducibility, we deployed our framework as a PyPI package and release the accompanying code and prompts, enabling reproduction of our results and comparable benchmarking against Vanilla RAG.

REFERENCES

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Cody V Dong, Qihong Lu, Kenneth A Norman, and Sebastian Michelmann. Towards large language models with human-like episodic memory. *Trends in Cognitive Sciences*, 2025.

Yuyang Hu, Shichun Liu, Yanwei Yue, Guibin Zhang, Boyang Liu, Fangyi Zhu, Jiahang Lin, Honglin Guo, Shihan Dou, Zhiheng Xi, et al. Memory in the age of ai agents. *arXiv preprint arXiv:2512.13564*, 2025.

Zhengjun Huang, Zhoujin Tian, Qintian Guo, Fangyuan Zhang, Yingli Zhou, Di Jiang, and Xiaofang Zhou. Licomemory: Lightweight and cognitive agentic memory for efficient long-term reasoning. *arXiv preprint arXiv:2511.01448*, 2025.

Bowen Jiang, Yuan Yuan, Maohao Shen, Zhuoqun Hao, Zhangchen Xu, Zichen Chen, Ziyi Liu, Anvesh Rao Vijjini, Jiashu He, Hanchao Yu, et al. Personamem-v2: Towards personalized intelligence via learning implicit user personas and agentic memory. *arXiv preprint arXiv:2512.06688*, 2025.

Bernal Jimenez Gutierrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. Hipporag: Neurobiologically inspired long-term memory for large language models. *Advances in Neural Information Processing Systems*, 37:59532–59569, 2024.

Mingcong Lei, Honghao Cai, Zezhou Cui, Liangchen Tan, Junkun Hong, Gehan Hu, Shuangyu Zhu, Yimou Wu, Shaohan Jiang, Ge Wang, Yuyuan Yang, Junyuan Tan, Zhenglin Wan, Zhen Li, Shuguang Cui, Yiming Zhao, and Yatong Han. Robomemory: A brain-inspired multi-memory agentic framework for interactive environmental learning in physical embodied systems, 2025. URL https://arxiv.org/abs/2508.01415.

Yueqian Lin, Qinsi Wang, Hancheng Ye, Yuzhe Fu, Hai Li, Yiran Chen, et al. Hippomm: Hippocampal-inspired multimodal memory for long audiovisual event understanding. *arXiv preprint arXiv:2504.10739*, 2025.

Lihui Liu and Kai Shu. Unifying knowledge in agentic llms: Concepts, methods, and recent advancements. *ACM SIGKDD Explorations Newsletter*, 27(2):88–96, 2025.

Mathis Pink, Qinyuan Wu, Vy Ai Vo, Javier Turek, Jianing Mu, Alexander Huth, and Mariya Toneva. Position: Episodic memory is the missing piece for long-term llm agents, 2025. URL https://arxiv.org/abs/2502.06975.

Lianlei Shan, Shixian Luo, Zezhou Zhu, Yu Yuan, and Yong Wu. Cognitive memory in large language models, 2025. URL https://arxiv.org/abs/2504.02441.

Schaun Wheeler and Olivier Jeunen. Procedural memory is not all you need: Bridging cognitive gaps in llm-based agents. In *Adjunct Proceedings of the 33rd ACM Conference on User Modeling, Adaptation and Personalization*, pp. 360–364, 2025.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*, 2025.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Zuomin Yang, Anis Salwa Binti Mohd Khairuddin, Joon Huang Chuah, Wei Ru Wong, Hafiz Muhammad Fahad Noman, Tri Wahyu Oktaviana Putri, Xin Xu, and Md Minhazul Haq. A review of short-term, long-term, and memory consolidation mechanisms in the hippocampus and cerebral cortex. *IEEE Access*, 13:63248–63283, 2025. doi: 10.1109/ACCESS.2025.3555539.

Sizhe Yuen, Francisco Gomez Medina, Ting Su, Yali Du, and Adam J Sobey. Intrinsic memory agents: Heterogeneous multi-agent llm systems through structured contextual memory. *arXiv preprint arXiv:2508.08997*, 2025.

A Datasets

HotpotQA is a multi-hop question answering dataset that requires combining evidence across multiple supporting facts. We evaluate CraniMem on a sampled subset of HotpotQA dataset.

We construct a sample of 100 instances from the HotpotQA validation split and use this subset to test the long horizon behavior of our memory pipeline, including selective write gating, bounded episodic storage, and consolidation into the semantic store.

B CraniMem Implementation Details

B.1 Input gating and utility tagging

Given the user message u_t and the agent goal and state g_t, CraniMem computes semantic relevance using an embedding model $E(\cdot)$:

$$\mathrm{Sim}(u_t, g_t) = \mathrm{CosineSimilarity}\big(E(u_t), E(g_t)\big).$$

If $\mathrm{Sim}(u_t, g_t) < \tau_{\mathrm{noise}}$, the input is discarded. Otherwise, an LLM-based tagger produces scalar signals for importance, surprise, and emotion, which are aggregated into a base utility score:

$$\mathrm{BaseUtility}(u_t) = \tfrac{1}{3}\big(I(u_t) + S(u_t) + E(u_t)\big).$$

B.2 Episodic buffer

Accepted items are stored in a bounded FIFO episodic buffer over the most recent N turns. Each entry contains the raw snippet, timestamp and turn id, extracted entities, and utility metadata. The buffer supports short-range continuity and provides the candidate set for consolidation.

B.3 Replay selection and consolidation

CraniMem triggers consolidation every N turns or during idle states. For each buffer item m, a replay score combines intrinsic utility with a frequency bonus based on repeated entities and repeated access:

$$\mathrm{ReplayScore}(m) = \mathrm{BaseUtility}(m)\,\Big(1 + \alpha\,\mathrm{FreqBonus}(m)\Big).$$

Only items with $\mathrm{ReplayScore}(m) > \tau_{\mathrm{consolidation}}$ are selected for transfer to long-term memory.

B.4 Linkage engine and knowledge graph updates

Selected memories are converted into a knowledge graph by extracting typed entities and typed relations, applying checks, and performing an upsert and merge operation that reinforces repeated facts. This representation supports multi-hop recall through graph traversal, which is difficult for flat vector stores.

B.5 DUAL-PATH RETRIEVAL

At generation time, CraniMem retrieves evidence from the episodic buffer for short-range continuity and from the knowledge graph for long-range semantic recall. The two streams are merged into a compact context block that conditions the LLM.

C EVALUATION METRICS

We report both clean and noisy performance to quantify robustness of memory under context pollution.

Clean metrics: On a clean evaluation setting without injected distractors, we compute the averaged standard classification and retrieval quality measures of precision, recall, and F1 score.

Noisy metrics: To test distraction resistance, we evaluate under a noisy setting where irrelevant context is injected into the agent's input stream. We again report the averaged precision, recall, and F1 score.

Noise drop: We measure degradation due to noise using noise drop, defined as the gap between clean and noisy F1 score:

$$\Delta_{\text{noise}} = \text{F1}_{\text{clean}} - \text{F1}_{\text{noisy}}.$$

Lower noise drop indicates higher robustness and better memory gating and retrieval precision under distraction.

Latency: Finally, we report latency as the end-to-end per turn inference time, including memory read, memory write, and consolidation overhead when triggered. Latency captures the practical cost of memory operations and highlights the trade-off between robustness and runtime efficiency.

D PROMPTS

This section reports the prompt templates used for goal directed gating, utility tagging, and knowledge graph extraction.

Gating System Prompt

```
You are the goal directed Gating Mechanism.
Your job is to filter information based strictly on its RELEVANCE to
    the current Agent Goal.

### CURRENT AGENT GOAL:
"{goal}"

### INSTRUCTIONS:
Analyze the User Input. Does it contain information that helps achieve
     the Goal?

1. **HIGH PRIORITY (Score 8-10):**
   - Directly contributes to the goal (e.g., specific facts,
       constraints, deadlines).
   - "Project X is due Friday" (If goal is Project Management).
   - "I am allergic to peanuts" (If goal is Food Ordering).

2. **MEDIUM PRIORITY (Score 4-7):**
   - Context that *might* be useful later.
   - Clarifications or minor updates.

3. **NOISE (Score 0-3):**
   - Information irrelevant to the current goal.
   - Chitchat, greetings, or off-topic distractions.
```

```
...
IMPORTANT: Output ONLY raw JSON. Do not include any preamble, markdown
    , or explanations.
Output strictly in JSON format:
{{
  "is_noise": boolean,
  "priority_score": integer, // SCALE: 1-10.
                      // 10 = Critical technical facts, research data,
                          or goal-aligned info.
                      // 7-9 = Relevant context or useful updates.
                      // 1-6 = Mundane chitchat, emotional noise, or
                          irrelevant distractions.
  "entities": [list of strings],
  "reasoning": "string"
}}
```

Utility Tagging System Prompt

```
You are a Utility Tagger.
Analyze the input for three RAS factors and return JSON only.
IMPORTANT: Output ONLY raw JSON. Do not include any preamble, markdown
    , or explanations.
Output ONLY JSON. Do not write any conversational text, preambles, or
    explanations.

Factors:
1) Importance: Is it technical/goal-oriented?
2) Surprise: Is it new or unexpected?
3) Emotion: Is there strong sentiment?

Guidelines:
- Each factor is a float in [0, 1].
- If the input is clearly irrelevant or trivial, keep scores near 0.
- "entities" should be key proper nouns or concepts, if any.

Return JSON:
{{
  "importance": float,
  "surprise": float,
  "emotion": float,
  "entities": [list of strings]
}}
```

Reflex Utility System Prompt

```
"You are a Utility Tagger.
The vector match is HIGH, so the input is already considered relevant.
Return ONLY the utility scores.
IMPORTANT: Output ONLY raw JSON. Do not include any preamble, markdown
    , or explanations.

Return JSON:
{{
  "importance": float,
  "surprise": float,
  "emotion": float,
  "entities": [list of strings]
}}
```

Cortex Gating System Prompt

```
You are the Cortex Gate for low vector similarity inputs.
The vector match is LOW, so you must decide whether this input is:
1) a command,
2) a goal change / context shift,
3) relevant context, or
4) noise.

IMPORTANT: Output ONLY raw JSON. Do not include any preamble, markdown
    , or explanations.

Return JSON:
{{
  "is_noise": boolean,
  "category": "command|goal_change|relevant_context|noise",
  "importance": float,
  "surprise": float,
  "emotion": float,
  "entities": [list of strings],
  "reason": "short string"
}}
```

Reasoning Prompt

```
You are a precise answering machine.
GOAL: "{current_goal}"

CONTEXT:
{context}

USER INPUT:
{current_input}

INSTRUCTIONS:
1. Answer the question accurately using the context.
2. Output ONLY the answer entity (Name, Place, Date, etc.).
3. NO full sentences. NO "The answer is...".
4. If the answer is a long name (e.g., "The United States of America")
    , write the FULL name.
5. Wrap your final answer inside <RESPONSE> tags.

Example:
User: Who directed the movie?
<RESPONSE>Scott Derrickson</RESPONSE>

User: What organization?
<RESPONSE>International Boxing Hall of Fame</RESPONSE>

YOUR TURN:
```

Entity Relation Extraction System Prompt

```
You are a Knowledge Graph extraction system.
Extract only the entities and relationships that are explicitly
    supported by the text.

STRICT FILTERING:
1) Ignore general knowledge or common sense statements (e.g., "The sky
     is blue", "Pizza is tasty", "Python is a language").
```

```
2) Focus ONLY on specific, unique facts about people, places,
    organizations, events, and dates.
3) Prefer dense/complex information: multi-entity, multi-relation, or
    highly specific facts.
4) If a sentence contains no specific named entities, SKIP IT.

IMPORTANT: Output ONLY raw JSON. Do not include any preamble, markdown
    , or explanations.

Return JSON only, with this schema:
{{
  "entities": [
    {{"type": "Project|Issue|Task|Person|Tool|Feature|Location|Date|
        Other", "name": "string"}}
  ],
  "relations": [
    {{"source": "entity name", "relation": "string", "target": "entity
        name"}}
  ]
}}

Rules:
1) If no entities exist, return empty lists.
2) Do not invent entities or relations.
3) Use concise names, no labels in the name field.
4) relation should be a short verb phrase (e.g., "has_issue", "
    blocked_by", "uses").
```

Entity Extraction Prompt

```
You are a Knowledge Graph Engineer.
   Extract the key "Entities" (Proper Nouns, Project Names, Locations,
       Dates) from the text below.

   Rules:
   1. Return ONLY a comma-separated list of entities.
   2. Do not include labels (e.g., don't say "Person: Sarah", just say
       "Sarah").
   3. If no entities are found, return "None".

   TEXT: {input}

   ENTITIES:
```