LYRN Episodic Memory: Technical Architecture for Storing and Indexing Interactions

1. Scope and Purpose

This paper details the technical foundation of episodic memory in the LYRN (Living Yield Relational Network) framework. Its focus is on how every user/AI interaction is persistently stored, indexed, and made efficiently retrievable. This infrastructure enables rich, context-aware retrieval and lays the groundwork for higher-level relational memory formation, which is addressed in separate LYRN documentation.

2. System Overview: Modular Episodic Memory Components

LYRNs episodic memory system is engineered for transparency, efficiency, and extensibility. It consists of several interlocking modules:

Chat Entry Logs:

Every user input and AI response is captured as a plain text file, timestamped for precise chronological tracking. These logs are minimal by design, containing only the actual conversational content.

Block Indexes:

For scalability, chat entries are grouped into blocks (typically 50 entries per block). Each block is accompanied by an SQL-based index file that stores entry summaries and metadata (timestamp, project tag, emotional context, topic, etc.).

Conversation Index:

Above the block level, conversation indexes link multiple blocks into session ranges. Each session

includes a reference to the block range and a high-level summary of the sessions content or purpose.

Project Indexes:

Blocks and sessions are associated with dynamic or static project IDs. This allows memory to be scoped and retrieved by project or purpose, supporting workflows that span multiple sessions or objectives.

Snapshot:

The snapshot file acts as a symbolic, up-to-date mind-state for the agent. It contains current goals, past insights, open loops, and relational cues. The snapshot is referenced during reasoning and can be updated by new interactions or system cycles.

Delta Updates:

Every interaction can generate a delta entry, which records updates to memory or insights derived from reflection or user correction. Deltas are layered on top of the snapshot, allowing for granular, field-level updates without overwriting the original data.

Heartbeat Cycle:

This background process maintains system rhythm. Between conversations or actions, the heartbeat reflects, summarizes, prunes, and updates memory tables, ensuring the system stays aligned and current.

Reflection Cycle:

The reflection cycle is a deeper reasoning pass that extracts insights, fills in missing metadata, and combines project data. It does not delete chat history; instead, it builds meaning on top of it, preparing the data for higher-level processing.

3. Technical Process: Encoding, Storage, and Retrieval

The LYRN episodic memory system is designed to mirror the essential stages of human memory: encoding, storage, and retrieval.

Encoding:

Each conversational turn is immediately encoded as a timestamped text entry, optionally tagged with metadata for later indexing.

Storage:

Entries are stored in a hierarchical directory structure (by block, session, and project), with SQL-based indexes for rapid lookup and summary.

Retrieval:

Memory retrieval is initiated by a user query or an LLM trigger. The process is as follows:

1. A script parses block folders in reverse chronological order.
2. Chat entries are searched for keyword matches. Block indexes can also be searched by project tag, summary context, or any tracked metadata.
3. Matching entries are returned in order of relevance or recency to a single text file for review.
4. The LLMs snapshot logic determines which entries to review, summarize, or load into short-term context.

This approach avoids token bloat by only pulling what is relevant, letting the LLM reason over curated context, and leveraging indexed metadata for efficient filtering.

4. Design Principles and Technical Advantages

Transparency and Auditability:

All memory, insight, and reasoning trails are saved as human-readable logs. This makes the system fully auditable and debuggable.

Efficiency and Scalability:

By grouping entries and using SQL-based indexes, the system can scale to years of interaction history without performance loss.

Contextual Flexibility:

Memory can be retrieved by time, project, topic, or emotional context, supporting a wide range of user and agent needs.

Separation of Episodic and Relational Memory:

This episodic memory layer provides the raw, structured data that higher-level relational memory processes (such as topic indexing and semantic linking) can later synthesize into long-term understanding.

5. Relationship to Higher-Level Memory (Relational/Semantic)

It is important to note that this paper describes only how interactions are stored and indexed for long-term use. The actual formation of relational, topic-based, or semantic memoryhow the system synthesizes episodic data into concepts, tracks topics, and generates self-insightsis handled by LYRNs topic indexing and reflection modules, described in separate documentation.

In summary, the episodic memory infrastructure ensures that every detail of the agents experience is persistently, efficiently, and transparently stored, providing a robust substrate for advanced cognition.

6. Conclusion

LYRNs episodic memory system is the foundation for persistent, context-rich AI cognition. By combining modular logging, hierarchical indexing, and flexible retrieval, it enables local language models to maintain a detailed, auditable record of their interactions. This technical groundwork is essential for building higher-level relational memory and truly human-like reasoning in AI.