Memoir.ai — AI Subsystem Technical Manual

Architecture, Execution, and Operational Reference

Document Version: 1.0

Status: Engineering Reference

Owner: AI Systems Engineering

Last Updated: YYYY-MM-DD

----

1. Purpose

This technical manual defines the architecture, operational flow, safeguards, and integration requirements of the Memoir.ai AI subsystem. It serves as the authoritative reference for engineers implementing, maintaining, and extending AI-powered features within the platform.

----

2. Scope

This manual covers:

- AI subsystem architecture
- Snapshot generation pipeline
- Evidence extraction and context construction
- Model inference and orchestration
- Citation anchoring logic

- Hallucination prevention mechanisms
- Narrative versioning and regeneration logic
- Evaluation and quality measurement
- Safety and privacy safeguards
- Performance considerations
- Failure recovery patterns

User-facing documentation is out of scope.

----

3. System Overview

Memoir.ai AI operations run primarily on-device to preserve privacy and prevent unintended data exposure. Processing occurs through isolated worker processes interacting with the encrypted vault.

Subsystem components include:

- Evidence ingestion pipeline
- Prompt context builder
- Local LLM inference engine
- Citation engine
- Post-processing verification layer
- Version storage system
- Evaluation and safety checks

All AI outputs must remain traceable to source evidence.

----

4. Architectural Components

4.1 Evidence Extraction Engine

Normalizes imported event data into structured records used for AI prompts. Responsibilities include:

- Participant detection
- Temporal anchor resolution
- Event clustering
- Media association
- Deduplication handling

Output feeds prompt construction.

4.2 Prompt Context Builder

Constructs context windows supplied to inference models. Context includes:

- Evidence summaries
- Timeline anchors
- Entity mappings
- Tone and length parameters
- User preferences

Context assembly prioritizes factual grounding.

4.3 Model Inference Engine

Local models generate narratives or summaries. Responsibilities include:

- Model loading and lifecycle control
- Resource scheduling
- Prompt injection
- Token limit enforcement
- Output streaming support

Models may vary by hardware capability.

4.4 Citation Engine

Maps narrative claims back to event identifiers.

Rules:

- Citations must support explicit claims.
- Claims without evidence receive no citation.
- Conflicting evidence produces conflict markers.

----

5. Snapshot Generation Pipeline

Execution stages:

1. User selects event range or conversation.
2. Evidence set assembled.

3. Entities and timeline anchors extracted.

4. Prompt context constructed.

5. Local inference executed.

6. Output passed through verification checks.

7. Citations linked to events.

8. Sanitization and filtering applied.

9. Narrative stored with version metadata.

Failures trigger retries or error states.

----

6. Narrative Versioning

Version rules:

- Each regeneration creates immutable versions.
- Manual edits create protected user versions.
- Automatic regeneration cannot overwrite manual edits.
- Only five recent versions retained unless pinned.

Version history enables rollback and comparison.

----

7. Hallucination Prevention

7.1 Natural Language Inference Guard

Each generated sentence is validated against cited evidence. Contradictions trigger rejection or revision.

7.2 Entity Validation Guard

Names, locations, and dates must exist in evidence metadata. Invented entities cause hard failures.

7.3 Tone Guard

Overly assumptive or moralizing language is flagged and softened.

----

8. Evaluation Framework

Evaluation occurs during testing and optional runtime checks.

Metrics include:

- Claim accuracy rate
- Citation density
- Narrative flow quality
- Hallucination frequency
- Generation latency

Snapshots must exceed defined thresholds to be marked verified.

----

9. Safety & Privacy Constraints

AI must not:

- Provide medical, legal, or psychological diagnoses.
- Infer criminal or malicious intent.
- Provide financial advice.
- Expose personal identifiers in public outputs.

Sensitive data must be redacted when exporting summaries.

----

10. Performance Considerations

Operational constraints include:

- Snapshot generation targets under 10 seconds on modern hardware.
- Worker memory usage capped per job.
- Background jobs throttled to prevent UI blocking.
- Large evidence sets chunked for inference.

Performance degradation triggers diagnostics alerts.

----

11. Failure Modes & Recovery

Common failures include:

Inference timeout:

- Retry inference with reduced context.

Citation mismatch:

- Re-run citation mapping.

Guard failure:

- Mark snapshot unverified and prompt regeneration.

Worker crash:

- Restart worker and restore job state.

----

12. Model Lifecycle Management

Responsibilities include:

- Hardware capability detection
- Model tier selection
- Efficient model loading
- Graceful shutdown of idle models
- Memory cleanup after inference

Model upgrades must preserve output compatibility.

----

13. Integration Interfaces

Subsystem interacts with:

- Vault database for evidence retrieval
- Job runner for background tasks
- UI components for progress display
- Entitlement system for generation limits

All IPC messages require schema validation.

----

14. Operational Best Practices

Engineering teams should:

- Validate evidence normalization pipelines regularly.
- Monitor hallucination guard effectiveness.
- Benchmark inference latency across releases.
- Maintain backward compatibility of narrative formats.
- Run regression evaluation tests before releases.

----

15. Conclusion

This technical manual defines how Memoir.ai generates trustworthy narratives while preserving user privacy and data integrity. Proper adherence ensures AI outputs remain verifiable, performant, and safe while supporting future platform evolution.