EN

Practical Guide

OpenClaw: Memory and Context

MEMORY.md, Compactions and Long-Term at Scale

Memory architecture for AI agents in production

Mars 2026

Table of Contents

AI has no native memory — but an operational agent needs it. BOTUM documents its memory architecture: MEMORY.md, daily notes, compactions, and management at the scale of a 15-agent production network.

1. Context Window vs. Persistent Memory

The most common confusion: conflating the context window with memory. These are two fundamentally different things.

The context window is the amount of information a model can process in a single session. It is volatile (clears at session end) and saturates (older information silently falls off when the context fills up).

Persistent memory is what survives between sessions. It is an artifact of your infrastructure, not the model. It doesn't create itself.

- Decisions made in prior weeks
- Expressed user preferences
- Already-documented errors
- Client contexts and project histories

■ *Per Gartner 2025, 71% of AI agent deployment failures are linked to inadequate memory architecture rather than model capabilities.*

2. MEMORY.md — Long-Term Memory

In the OpenClaw architecture, MEMORY.md is the consolidated memory file. It contains what must persist beyond individual sessions.

What goes in it

- Immutable agent rules
- Key organizational information
- Important decisions and their rationale
- Documented errors not to repeat
- Integrations and their quirks

Curation rules

- Controlled size: target 2,000 to 5,000 tokens maximum
- Stable sections organized by domain
- Active writing principle: nothing enters without something leaving
- Git versioning: every modification is a commit

> *“This file is read at the start of each main session. It is the context injection that allows the agent to remember.”*

3. Daily Notes — Raw Daily Logs

Daily notes (memory/YYYY-MM-DD.md) are the agent's logbook. Each day has its file. Things are captured in real time as they happen.

When to write

• After each significant action (email sent, script executed, configuration modified)

• When an implicit rule was applied (to make it explicit)

• When unexpected behavior was observed

• When a decision was made that might be questioned later

What to capture

• The what (precise action), the why (context), the result

• Heuristics used to resolve an ambiguity

• References (ticket number, client identifier, relevant URL)

• Warning signals that deserve attention

■ *Daily notes are not written to be read in the current session. Their value is not immediate: it is in accumulation.*

4. Compactions — Distilling Memory

Compaction is the distillation operation: take a volume of daily notes and extract what deserves to survive in MEMORY.md.

When to compact

• When the context window approaches saturation (>40% utilization)

• Periodically, independent of utilization (weekly or monthly)

• Before approaching a new domain or project

• When MEMORY.md itself becomes too heavy

BOTUM 4-step protocol

1. Rereading daily notes from the period

2. Extracting persistence candidates (rules, decisions, documented errors)

3. Updating MEMORY.md with merging and deduplication

4. Git commit with descriptive message — daily notes remain in the archive

Tests for deciding what to keep

• Permanence test: will this information still be relevant in 3 months?

• Frequency test: will this element be consulted often, or is it a one-off event?

• Derivability test: can this info be found elsewhere if needed?

• Rule test: does this event reveal a general rule applicable to other situations?

5. Shared Memory vs. Isolated Memory

In a multi-agent network, what information is shared between agents, and what remains private?

Shared memory

The common workspace: MEMORY.md, CODEX.md, AGENTS.md. Accessible to all agents. This is organizational memory: common rules, system state, collective decisions.

Isolated memory

Agent directories (agents/agent-name/). Each agent can have its private notes, specific configurations. What stays in that directory is not automatically injected into other agents.

BOTUM separation principle

• Rules that apply to everyone → shared workspace (MEMORY.md)

• Domain-specific rules → the relevant agent's directory

• Execution logs → the relevant agent's daily notes, not in the common workspace

• Inter-agent artifacts → dedicated directory (handoffs/)

> *"A shared workspace that grows without control quickly becomes a noise source — and ends up degrading the context quality for all agents."*

6. Semantic Search in Memory

After 6 months of production, a network of 15 agents may generate several thousand files. Two search levels available in OpenClaw:

Text search

```
grep -r "keyword" memory/
```

Effective for finding exact occurrences or known identifiers.

Semantic search

Via natural language queries on daily notes. The agent reads the most relevant files and synthesizes. This approach consumes tokens — reserved for important searches where precision outweighs cost.

Practices for keeping notes searchable

• Always include explicit identifiers (proper names, system identifiers, URLs)

• Structure entries with consistent headers (## Decision, ## Error, ## Rule)

• Maintain a monthly summary index with key decisions and dates

• For critical decisions: duplicate in MEMORY.md with reference to source daily note

7. At Scale: 15 Agents, 6 Months of Memory

Concrete challenges BOTUM has encountered at this scale:

Volume

15 agents x 6 months = potentially 2,700 daily note files. Volume quickly exceeds human supervisory capacity.

Quality degradation

Without active maintenance, MEMORY.md accumulates obsolete rules, contradictory information, references to completed projects.

BOTUM protocols

• Dedicated memory system agent (JARVIS): weekly audit, compactions, conflict resolution

• Contribution rules: only the system agent can modify MEMORY.md directly

• Daily note archiving after 90 days outside the active workspace

• Size monitoring: daily cron with alert if MEMORY.md exceeds threshold

8. Anti-Patterns to Avoid

Anti-pattern 1: Putting everything in MEMORY.md

A 20,000-token MEMORY.md is a permanent context problem. It takes up space in every session and dilutes the relevance of important information.

Anti-pattern 2: Never compacting

Daily notes accumulate indefinitely. Without periodic compaction, the archive grows, searches become expensive, and long-term memory remains trapped in raw log granularity.

Anti-pattern 3: Confusing context and memory

Active context is not memory. Massively injecting files into context does not replace a well-designed memory architecture — and uselessly overloads the token window.

Conclusion

Agent memory is not a feature — it's infrastructure. It is designed, deployed, maintained. It degrades without upkeep. It scales if you anticipate its constraints.

The MEMORY.md + daily notes + periodic compaction architecture has been validated by BOTUM in production over 6 months and more than 50,000 agent exchanges. Simple to understand, robust in use.

Post 10: OpenClaw and databases — how to connect agents to PostgreSQL, SQLite, and structured APIs to go beyond file-based memory.

Full article: blog.botum.ca/openclaw-memory-context-memory-md-compactions-long-term

Website: www.botum.ca • contact@botum.ca