Fix Backlog

4.3 — Prioritized Fix Backlog (MC-Stub List)

AI Factory Audit — 2026-05-09 Author: Petter Graff (CodeCraft Lead Architect) **Source:** 4.1-petter-synthesis.md + 4.2-devils-advocate.md **Status:** AUDIT-LEVEL ONLY — no MCs created in live system. CEO selects from this list.

Section 1 — Prioritized MC-Stub List

Composite = Leverage (1–10) × Severity (1–10) ÷ Effort (S=1, M=2, L=4) Devils-advocate score adjustments applied. Final ordering is post-rebuttal.

MC-STUB-01: Restore RAG drain-worker — fix Vaultwarden session + CF Access credentials

- **Subsystem:** Daemon fleet / RAG ingest pipeline
- **Owner-company:** FlowForge
- **Priority:** H
- **Composite (Leverage × Severity / Effort):** 81 (9 × 9 / 1)
- **Effort:** S (≤2h)
- **Cost (token + CEO-action time):** ~$0.20 tokens / 5 min CEO (approve billing session if needed)
- **Acceptance criteria** (machine-checkable):
  - ☐ `cat /tmp/bw-session` exits 0 and returns a non-empty string
  - ☐ `curl -s http://localhost:9621/health` returns `{"status":"healthy"}` (LightRAG reachable)
  - ☐ `launchctl list | grep rag-drain-worker` shows LastExitStatus = 0 within 15 min of fix
  - ☐ `stat ~/system/state/rag-drain.prom` shows mtime within last 10 min (metric is live)

☐ Live queue depth is written to `~/system/state/rag-drain-live.json` (new artifact — see MC-STUB-03)

- **Evidence path:** 4.1 §3 Gap #1, 4.2 Gap #1 (CONFIRMED), P3.1 H1, P1.4 §3
- **Why now / Why this owner:** This single credential fix unblocks 3 adapters simultaneously and drains 3,150+ queued items (live SQLite count 2026-05-09; stale prom snapshot showed 454 as of 2026-04-23). FlowForge owns daemon lifecycle and credentials management.
- **BlockedBy:** None

MC-STUB-02: Resolve canonical dispatch path — pi-orch HTTP vs durable-runner

- **Subsystem:** Orchestration kernel
- **Owner-company:** CodeCraft
- **Priority:** H
- **Composite (Leverage × Severity / Effort):** 18 (8 × 9 / 4) — design work, L effort
- **Effort:** L (≤2d — includes live probes + decision doc + architectural note)
- **Cost (token + CEO-action time):** ~$1.50 tokens / 20 min CEO (one architectural decision required)
- **Acceptance criteria** (machine-checkable):

  ☐ A file `~/system/specs/dispatch-path-canonical.md` exists with mtime today

  ☐ The file explicitly states which of {pi-orch HTTP port 8401 | durable-runner port 3052} is the canonical dispatch layer

  ☐ If pi-orch HTTP is canonical: `curl -s http://localhost:8401/health` returns HTTP 200 after fix

  ☐ If durable-runner is canonical: `grep -c "dispatched" ~/system/logs/durable-runner.log` shows at least 1 entry with today's date within 24h of fix

  ☐ No dispatch logs older than 2026-04-01 are the NEWEST entry (proves dispatch is current)
- **Evidence path:** 4.1 §4 (dual-process dispatch pattern), 4.2 Gap #2 (CONFIRMED BUT MISDESCRIBED), 4.2 New Gap B
- **Why now / Why this owner:** Every other orchestration fix is blocked on knowing which process is authoritative. CodeCraft holds kernel architecture; the decision requires architectural judgment, not just ops execution.
- **BlockedBy:** None (this IS the unblocking action for MC-STUB-05)

MC-STUB-03: Implement live RAG queue depth monitoring

- **Subsystem:** Daemon fleet / Observability
- **Owner-company:** FlowForge
- **Priority:** H
- **Composite (Leverage × Severity / Effort):** 17.5 (5 × 7 / 2)
- **Effort:** M (≤8h)
- **Cost (token + CEO-action time):** ~$0.30 tokens / 0 min CEO (no decision needed)
- **Acceptance criteria** (machine-checkable):
  - ☐ `~/system/state/rag-drain-live.json` exists and contains `queue_depth` key
  - ☐ mtime of that file is within 5 min of any check
  - ☐ `launchctl list | grep rag-queue-monitor` shows LastExitStatus = 0
  - ☐ HiveMind receives an alert if queue_depth exceeds 100 (verify via `node ~/system/agents/hivemind/hivemind.js query "rag queue"` showing a row within last 1h)
- **Evidence path:** 4.2 New Gap C — 454-item figure was a 16d-stale metric; true queue depth unknown when rag-drain-worker crashed today
- **Why now / Why this owner:** Without live queue depth, every future RAG incident assessment will rely on stale file mtimes. FlowForge owns the monitoring daemon pattern.
- **BlockedBy:** MC-STUB-01 (drain-worker must be restored first; queue depth metric is only meaningful when writer is live)

MC-STUB-04: Restore or unload 5 deleted-script daemon plists

- **Subsystem:** Daemon fleet / Monitoring
- **Owner-company:** FlowForge
- **Priority:** M (pi-orch-health sub-task is H)
- **Composite (Leverage × Severity / Effort):** 35 (5 × 7 / 1)
- **Effort:** S (≤2h)
- **Cost (token + CEO-action time):** ~$0.15 tokens / 0 min CEO
- **Acceptance criteria** (machine-checkable):
  - ☐ `launchctl list | grep -E "pi-orch-health|cost-daily-report|daily-planning|legal-docs-azure-sync|mcp-health-check"` shows ZERO entries (unloaded) OR shows LastExitStatus = 0 (restored)
  - ☐ `ls ~/system/daemons/pi-orch-health.sh` exits 0 if restored; if unloaded, plist file is absent from `~/Library/LaunchAgents/`
  - ☐ Zero exit-127 entries for these 5 daemon names in `launchctl list` within 24h of fix
  - ☐ If pi-orch-health is restored: it writes a report to `~/system/state/pi-orch-health-latest.json` with mtime within last 1h
- **Evidence path:** 4.1 §3 Gap #6, 4.2 Gap #6 (CONFIRMED), P1.4 §2, P3.1 G4/G5
- **Why now / Why this owner:** pi-orch-health.sh was the last known diagnostic for orchestrator state; it was deleted on 2026-05-06 when the last recorded status was CRITICAL. Blind monitoring of the primary kernel is not acceptable. FlowForge owns

daemon lifecycle.
- **BlockedBy:** MC-STUB-02 (pi-orch-health.sh restoration requires knowing which health signal to probe — depends on canonical dispatch decision)

MC-STUB-05: Enforce blueprint score gate — eliminate WARN bypass and missing-MC-ID hole

- **Subsystem:** BUILD-BLUEPRINT discipline / Mehanik gate
- **Owner-company:** CodeCraft
- **Priority:** M
- **Composite (Leverage × Severity / Effort):** 30 (6 × 5 / 1)
- **Effort:** S (≤2h)
- **Cost (token + CEO-action time):** ~$0.10 tokens / 5 min CEO (score floor decision: 60 or 90?)
- **Acceptance criteria** (machine-checkable):

  ☐ `grep -n "WARN\|warn" ~/system/hooks/pre-dispatch-gate.sh` shows no bypass path that allows WARN to proceed without explicit CEO override token

  ☐ A test run with a blueprint scoring 65 exits gate with non-zero exit code (BLOCKED)

  ☐ A run without MC-ID also exits gate with non-zero exit code (BLOCKED)

  ☐ `grep "SCORE_FLOOR" ~/system/hooks/pre-dispatch-gate.sh` returns a numeric value (60 or 90, per CEO decision)
- **Evidence path:** 4.1 §3 Gap #8, 4.2 Gap #8 (CONFIRMED), P2.3 §2
- **Why now / Why this owner:** A gate that emits warnings but allows dispatch is theater. The CEO's Mehanik enforcement ceremony is trusted — the underlying gate code must match the ceremony's intent. CodeCraft owns the gate scripting.
- **BlockedBy:** CEO decision on score floor value (see Section 4)

MC-STUB-06: Design decision + routing update for agent fleet coverage

- **Subsystem:** Agent fleet / Routing
- **Owner-company:** CodeCraft (design) + Resolver (if Resolver is activated)
- **Priority:** M
- **Composite (Leverage × Severity / Effort):** 18 (7 × 5 / 2) — post-rebuttal adjusted
- **Effort:** M (≤8h — requires design decision first, then data entry)
- **Cost (token + CEO-action time):** ~$0.40 tokens / 15 min CEO (routing policy decisions)
- **Acceptance criteria** (machine-checkable):

☐ A file `~/system/specs/agent-routing-policy.md` exists defining: which agents are routable via discover.js vs internal-only vs experimental

☐ `node ~/system/tools/discover.js routing "validate acceptance criteria"` returns a non-empty company/agent result

☐ `node ~/system/tools/discover.js routing "distill text"` returns a non-empty company/agent result

☐ `grep -c '"company"' ~/system/agents/specialist-mapping.json` is >= the previous count + however many new entries are added (verifiable by diff)

- **Evidence path:** 4.1 §3 Gap #5, 4.2 Gap #5 (CONFIRMED BUT UNDER-SPECIFIED)
- **Why now / Why this owner:** validator (44 skill references) and distiller (21 references) are the most-cited agents without routing entries. Silent dispatch failures are guaranteed when John tries to route tasks that map to these agents. Design decision first, then data entry.
- **BlockedBy:** CEO decision on routing policy scope (see Section 4); MC-STUB-02 for overall dispatch health

---

MC-STUB-07: Register or formally archive Axiom / Datavera / Resolver companies

- **Subsystem:** Agent fleet / Routing
- **Owner-company:** CodeCraft
- **Priority:** L
- **Composite (Leverage × Severity / Effort):** 10 (5 × 4 / 2)
- **Effort:** M (≤4h — inventory work products, then register or archive)
- **Cost (token + CEO-action time):** ~$0.20 tokens / 5 min CEO
- **Acceptance criteria** (machine-checkable):

  ☐ Each of Axiom, Datavera, Resolver, Lexicon appears EITHER in `specialist-mapping.json` (if active) OR has a `STATUS: experimental` or `STATUS: archived` entry in their `company.json` file

  ☐ `node ~/system/tools/discover.js routing "axiom"` returns a result or a clear "experimental — contact via direct session" message

  ☐ No company directory under `~/system/agents/personas/` has an unresolved routing status (every dir has an explicit status flag)
- **Evidence path:** 4.1 §3 Gap #7, 4.2 Gap #7 (CONFIRMED — all 4 unroutable: Axiom, Datavera, Resolver, Lexicon; Lexicon is absent from specialist-mapping.json)
- **Why now / Why this owner:** Silent routing fallthrough is a user-experience failure. When a task arrives that maps to Resolver or Lexicon capability, John will receive no routing error — the task will silently fall to the wrong handler. Four companies is a manageable cleanup.
- **BlockedBy:** MC-STUB-06 (routing policy decision must precede adding more entries)

MC-STUB-08: Restore pi-orchestrator dispatch to operational status

- **Subsystem:** Orchestration kernel
- **Owner-company:** CodeCraft
- **Priority:** H (blocked — becomes H after MC-STUB-02 resolves)
- **Composite (Leverage × Severity / Effort):** 22.5 (10 × 9 / 4) — Petter's original; blocked on design decision
- **Effort:** L (≤2d)
- **Cost (token + CEO-action time):** ~$2.00 tokens / 30 min CEO (architecture + approval of restored config)
- **Acceptance criteria** (machine-checkable):

  ☐ If pi-orch HTTP is the canonical path: `curl -s http://localhost:8401/health` returns HTTP 200

  ☐ If durable-runner is canonical: `node ~/system/tools/mc.js list --status ready --limit 1` followed by 5 min wait shows the task state has changed (dispatched or assigned) without manual John intervention

  ☐ Dispatch log file exists and has an entry with today's date: `grep "$(date +%Y-%m-%d)" ~/system/logs/pi-orchestrator.log | tail -1`

  ☐ No task with status "ready" sits unprocessed for more than 30 min in an idle queue (monitored via cron probe)
- **Evidence path:** 4.1 §3 Gap #2, 4.1 §4 (dual-process dispatch pattern), 4.2 Gap #2 (CONFIRMED BUT MISDESCRIBED)
- **Why now / Why this owner:** pi-orchestrator is the load-bearing wall of the factory. Without it dispatching automatically, John IS the factory. This is the gap that converts the system from manual radionica to automated pipeline. CodeCraft owns kernel architecture.
- **BlockedBy:** MC-STUB-02 (canonical dispatch path must be defined before this can be correctly fixed)

MC-STUB-09: Audit and archive Chroma + stale mem0 orphan collections

- **Subsystem:** Memory plane / Cleanup
- **Owner-company:** CodeCraft
- **Priority:** L
- **Composite (Leverage × Severity / Effort):** 15 (3 × 5 / 1)
- **Effort:** S (≤2h)
- **Cost (token + CEO-action time):** ~$0.10 tokens / 0 min CEO
- **Acceptance criteria** (machine-checkable):

☐ `curl -s http://localhost:8000/api/v1/collections` either returns a list with a documented owner for each collection, or returns connection refused (service confirmed decommissioned)

☐ If Chroma is decommissioned: its entry is removed from `~/.claude/settings.json` MCP server list

☐ `curl -s http://localhost:9000/v1/memories/?user_id=john` returns either 0 results or a documented "archived" state

☐ A `~/system/specs/memory-plane-canonical.md` file exists documenting the final memory topology: .md as SoR, LightRAG as searchable index, mem0/Chroma status (deprecated/experimental)

- **Evidence path:** 4.1 §3 Gap #9, 4.2 Gap #9 (CONFIRMED), 4.2 Gap #4 (DISMISSED — mem0 was never SoR; this cleanup is the correct response)
- **Why now / Why this owner:** Cognitive overhead from orphaned stores creates false recovery paths during incidents. The decommission is straightforward. The documentation artifact (memory-plane-canonical.md) satisfies the dismissed Gap #4 reframing.
- **BlockedBy:** None (can run in parallel with any Wave A task)

MC-STUB-10: Raise B2 storage cap and verify litestream replication health

- **Subsystem:** Backup / Infra
- **Owner-company:** FlowForge
- **Priority:** M
- **Composite (Leverage × Severity / Effort):** 28 (4 × 7 / 1)
- **Effort:** S (≤2h — primarily a billing console action)
- **Cost (token + CEO-action time):** ~$0.05 tokens / 10 min CEO (billing console access)
- **Acceptance criteria** (machine-checkable):

☐ `curl -s -H "Authorization: applicationKey:..." https://api.backblazeb2.com/b2api/v2/b2_get_bucket_info` returns `storageCapacity` > current used value (cap raised)

☐ `launchctl list | grep litestream` shows LastExitStatus = 0

☐ A litestream replication log entry exists from the last 24h: `grep "$(date +%Y-%m-%d)" ~/system/logs/litestream.log | tail -1`

☐ Nightly snapshot script exits 0: check `~/system/state/backup-status.json` shows last_success within 24h

- **Evidence path:** 4.1 §3 Gap #10, 4.2 Gap #10 (CONFIRMED), P1.4 §3, P2.1 Edge 38
- **Why now / Why this owner:** A capped backup bucket means data loss risk grows each day until raised. The fix is a billing action — no code required. FlowForge owns infra/backup.
- **BlockedBy:** None; requires CEO credentials for Backblaze console

---

MC-STUB-11: Document .md + LightRAG as canonical memory pipeline (doc-only)

- **Subsystem:** Memory plane / Documentation
- **Owner-company:** Skillforge
- **Priority:** L
- **Composite (Leverage × Severity / Effort):** 8 (4 × 4 / 2)
- **Effort:** M (≤4h — research + write + BookStack publish)
- **Cost (token + CEO-action time):** ~$0.30 tokens / 5 min CEO (approve publish)
- **Acceptance criteria** (machine-checkable):

  ☐ `~/system/specs/memory-plane-canonical.md` exists (may be produced by MC-STUB-09 instead — share artifact if so)

  ☐ CLAUDE.md "auto memory" section contains phrase "`.md is canonical`" or equivalent explicit statement

  ☐ BookStack page exists under the Infrastructure book for "Memory Plane Architecture" — `curl -s https://docs.alai.no/books/infrastructure | grep -i "memory"` returns a hit

  ☐ mem0 status is documented as "sandbox/experimental" in the spec (not "active SoR")
- **Evidence path:** 4.2 Gap #4 (DISMISSED — but reframed as doc task, not fix task); 4.2 Gap #4 recommendation: "Document .md is canonical"
- **Why now / Why this owner:** The dismissed Gap #4 still requires a documentation response. Without an authoritative statement, the next engineer touching the system will re-investigate and potentially re-introduce mem0 wiring. Skillforge produces technical documentation.
- **BlockedBy:** MC-STUB-09 (confirm Chroma/mem0 decommission state before documenting the final topology)

MC-STUB-12: Wire verify-fix-loop as optional /task-postflight enhancement (Wave C)

- **Subsystem:** Verifier / QA skill
- **Owner-company:** Proveo
- **Priority:** L
- **Composite (Leverage × Severity / Effort):** 16 (8 × 4 / 2) — post-rebuttal, demoted from H
- **Effort:** M (≤8h)
- **Cost (token + CEO-action time):** ~$0.40 tokens / 0 min CEO
- **Acceptance criteria** (machine-checkable):

☐ `grep -n "verify-fix-loop" ~/system/agents/skills/task-postflight/SKILL.md` returns at least 1 match (Section 2b exists)

☐ The section has a conditional trigger: domain IN {docs, system, refactor} AND Proveo PASS

☐ A dry-run of /task-postflight on a docs-domain MC shows verify-fix-loop invoked (not just Proveo)

☐ verify-fix-loop invocation does NOT replace Proveo (both must appear in the postflight log)

- **Evidence path:** 4.1 §3 Gap #3, 4.2 Gap #3 (DISPUTED — demoted; Proveo IS the required gate; this is an enhancement)
- **Why now / Why this owner:** verify-fix-loop is a fully built capability sitting idle. Wiring it as a conditional enhancement (not a required gate) improves self-correction for low-risk domains. Proveo owns the verification pipeline.
- **BlockedBy:** MC-STUB-08 (pi-orchestrator must be dispatching for auto-invocation to work reliably; in the interim, a manual invocation pattern is acceptable)

Section 2 — Sequencing Graph

Wave A — Immediate, S effort, high leverage (ship first)

These are unblocked today. Combined effort: ~6h. No CEO decisions needed to START.

```
MC-STUB-01 (RAG drain-worker credential fix)
    |
    +---> MC-STUB-03 (Live queue depth monitor)  [depends on 01 being live]

MC-STUB-04 (Restore 5 dead-script plists) [sub-task: pi-orch-health blocked on STUB-02]
MC-STUB-09 (Chroma/mem0 orphan audit)     [parallel, no deps]
MC-STUB-10 (B2 storage cap raise)          [parallel, no deps — billing action]
```

Wave A ships: 01, 03, 09, 10 (immediately); 04 partially (4 of 5 plists — pi-orch-health blocked on STUB-02).

Wave B — After Wave A + CEO decisions

These depend on an architectural decision or on Wave A completing.

```
MC-STUB-02 (Canonical dispatch path decision)
    |
    +---> MC-STUB-04 [remainder: pi-orch-health script restoration]
    |
    +---> MC-STUB-08 (Restore pi-orchestrator dispatch — actual kernel fix)
    |         |
    |         +---> MC-STUB-12 (wire verify-fix-loop — optional enhancement, needs dispatch
working)
    |
    +---> MC-STUB-06 (Routing policy decision + specialist-mapping update)
              |
              +---> MC-STUB-07 (Register Axiom/Datavera/Resolver or archive them)

MC-STUB-05 (Blueprint score gate enforce)  [needs CEO score floor decision — otherwise ship at
60]
```

CEO decision trigger: before MC-STUB-02 can produce a useful output, the CEO must make one call (see Section 4 item #1).

Wave C — Cleanup / hygiene (non-urgent)

No blocking dependencies. Run when bandwidth allows.

```
MC-STUB-09 --> MC-STUB-11 (memory-plane doc — safe to write after Chroma state is known)
MC-STUB-12  [verify-fix-loop wiring — Wave C because Wave B must stabilize dispatch first]
```

Full DAG (text form)

```
[NOW]
  STUB-01 (RAG creds)  ————————————————————> STUB-03 (queue monitor)
  STUB-04 partial (4 plists)
  STUB-09 (Chroma/mem0 audit)  ————————————> STUB-11 (memory doc)
  STUB-10 (B2 billing)

[CEO DECISION on dispatch path]
  STUB-02 (canonical dispatch decision)
    ├——> STUB-04 remainder (pi-orch-health)
    ├——> STUB-08 (pi-orch restore)  ——————————> STUB-12 (verify-fix-loop wire)
```

```
    └──> STUB-06 (routing policy)  ──────────> STUB-07 (3 phantom companies)


[CEO DECISION on score floor]
  STUB-05 (blueprint gate enforce)
```

Section 3 — Out of Backlog (and Why)

DISMISSED gaps — not a fix

mem0 SoR wire break (original Gap #4): Not a break. `.md + LightRAG` is the actual working design — Claude Code writes .md natively; lightrag-auto-ingest.sh routes .md writes to LightRAG. mem0 was a prototype that was never wired into the active pipeline. CLAUDE.md has zero mention of mem0 as SoR. The correct response is NOT to wire mem0 back — it is to document the actual design (see MC-STUB-11, a documentation-only stub).

verify-fix-loop "unwired" structural gap (original Gap #3): Framing was misleading. CLAUDE.md Hard Constraint #4 requires Proveo verification — and Proveo IS wired and called by /task-postflight. verify-fix-loop is an optional enhancement for docs/system/refactor domains, not the required gate. Adding it is a feature improvement (see MC-STUB-12, demoted to Wave C), not a structural fix.

DEMOTED gaps — lighter scope than original claim

4 phantom companies (original Gap #7 — scope confirmed at 4, not demoted): All 4 companies (Axiom, Datavera, Resolver, Lexicon) are absent from specialist-mapping.json. None are phantom in the sense of missing directories — all have full persona directories — but none are routable via the normal John → discover.js flow. The fix is: inventory work products, then register OR mark as experimental. Addressed in MC-STUB-07 at L priority (documentation + optional routing).

Verifier loop (original Gap #3 — demoted from H to L): Retained as MC-STUB-12 but explicitly classified Wave C, marked as optional enhancement not structural fix. Proveo is the real gate and it is working.

Section 4 — CEO Decision Items

These are blocking decisions that no engineer can make unilaterally. They gate specific MCs.

Decision 1 (CRITICAL — gates MC-STUB-02, 04, 08): Canonical dispatch path

The question: Is `durable-runner` (port 3052, 20d uptime, stable) the canonical dispatch layer — with pi-orchestrator HTTP (port 8401, dead) being an old control plane that can be decommissioned? OR is pi-orchestrator HTTP supposed to be online, and its deadness is a regression that must be fixed?

Why only CEO can decide: This is an architectural fork. If durable-runner is canonical, FIX is: document it, verify it's processing tasks, and decommission the old HTTP endpoint. If pi-orch HTTP is canonical, FIX is: diagnose startup gating (likely an initialization hang on Ollama or a flag file), restore it, and ensure durable-runner is correctly subordinate.

Options:

- A. durable-runner is canonical dispatcher. pi-orch HTTP is legacy. Document this, decommission port 8401.
- B. pi-orch HTTP is canonical. Diagnose and restore it. durable-runner is subordinate.
- C. Both should be operational. Hybrid model (requires Petter to specify the interaction model).

Decision 2 (M — gates MC-STUB-05): Blueprint score gate floor

The question: What is the enforced minimum score for dispatching a task through Mehanik gate?

Context: Observed practice allows dispatch at score 65 (WARN range). Original spec says 90 is the floor. The gate code currently treats WARN as pass-through. The correct floor must be chosen and hardcoded.

Options:

- A. Lower floor to 60 — match observed practice; WARN is acceptable.
- B. Floor stays at 90 — WARN becomes BLOCK; blueprints must be updated to score higher.
- C. Introduce tiered floors: 60 for L tasks, 75 for M, 90 for H+.

Decision 3 (M — gates MC-STUB-06, 07): Specialist-mapping.json scope policy

The question: Should specialist-mapping.json be comprehensive (cover all 66 agents, all 12 companies) — or curated (cover only primary dispatch paths, leaving internal/helper agents out)?

Why it matters: validator and distiller have 44 and 21 skill references respectively, but may be internal-only agents (called from other agents, not from John). If they're internal-only, they must NOT be in the routing table — they should be in the agent definition files only. If they ARE routable by John, they must be added.

Options:

- A. Curated: only John-dispatchable agents enter the routing table. Internal agents documented separately.
- B. Comprehensive: all agents mapped; entry type field distinguishes dispatch-routable from internal.

Decision 4 (L — informs MC-STUB-09, 11): mem0 future role

The question: What is mem0's long-term status?

Context: 865 stale facts in mem0_john. Zero active writers. `.md + LightRAG` is the working pipeline. mem0 server is running and consuming resources.

Options:

- A. Deprecate: stop mem0 server; archive its Qdrant vectors; remove from settings.json.
- B. Keep as parallel experimental sandbox: document it as optional enrichment layer, not canonical.
- C. Promote: wire a PostToolUse hook that writes every .md memory update to mem0 simultaneously (highest effort, not recommended).

Petter's recommendation: Option A (deprecate). The .md pipeline is working. mem0 is cognitive overhead with no active consumer.

Report produced by Petter Graff — CodeCraft Lead Architect Source: 4.1-petter-synthesis.md, 4.2-devils-advocate.md Audit date: 2026-05-09 MC stubs: 12 total. CEO selects 1-3 per session from top of each wave.

Revision #2
Created 2026-05-09 19:44:24 UTC by John

Updated 2026-06-14 20:03:00 UTC by John