AI Factory v2 — Phase 0 Backbone

AI Factory v2 — Phase 0 Backbone

Author: ALAI
Version: 2026-04-27
Status: COMPLETE

Executive Summary

AI Factory v2 Phase 0 restored critical feedback loops and observability infrastructure across 5 build tasks (MC #9865-9869). This work unblocks the 9-point CEO vision by fixing broken learning mechanisms: the Mehanik dispatch gate now enforces scope discipline, LightRAG container is restored for token deduplication, quality_score wiring enables self-learning routing, cost telemetry closes a $163K/week blind spot, and trace capture creates the corpus for future distillation and fine-tuning.

Status: 5/5 builder tasks COMPLETE per Proveo validation (MC #9870). Documentation task complete (MC #9871). Phase 0 is GREEN.

Objective: Restore feedback loops and activate architectural gates to prepare ALAI for compounding self-improvement phases post-triage (2026-05-02+).

Vision Reminder

The CEO approved a 9-point AI Factory vision:

1. **Self-building** — AutoCoder that writes and executes plans
2. **Self-learning** — Quality scores feed back into routing decisions
3. **Self-healing** — Autowork daemon drains task queues autonomously

4. **No SPOF** — All critical databases replicated, multi-cloud backup
5. **Portable** — Multi-provider LLM routing (Anthropic, OpenAI, Groq, Ollama)
6. **Free + paid models** — Tier routing balances cost vs quality
7. **LightRAG token saving** — Dedupe uploaded docs, query before planning
8. **Own fine-tuned model** — Post-revenue: distill from traces.db corpus
9. **AIOS** — Autonomous OS that schedules and executes work

Pre-Phase 0 realization: 10-12% (per 5 expert lens convergent analysis). Bottleneck: broken feedback loops. Every database designed to convert effort into learning operated write-only.

Full plan: `/Users/makinja/system/specs/ai-factory-v2-plan.md`

Phase 0 Goals

Phase 0 is the triage-compatible foundation layer that closes broken feedback loops, activates dispatch gates, and eliminates observability blind spots. All tasks absorb into existing Lane 2 (infra restart) with zero CEO touch during execution.

Key outcomes:

- Mehanik Phase 2 gate enforces 13-field marker schema (prevents scope creep disasters)
- LightRAG container restored (Vision 7 token savings unblocked)
- Quality score wiring enables self-learning routing (Vision 2 + 6)
- Cost telemetry blind spot closed ($163K/week now visible)
- Trace capture pipeline creates distillation corpus (gates Vision 8)

Architecture Diagram

```
flowchart LR
    subgraph Dispatch Gate
        A[John receives task] --> B{Mehanik clearance?}
        B -->|No marker| C[BLOCKED: exit 2]
        B -->|Valid 13-field marker| D[CLEAR: dispatch]
    end

    subgraph Tier Routing
        D --> E[tier-router.js classify]
        E --> F{quality_score feedback}
        F -->|avg < 0.6| G[Escalate tier+1]
```

```
        F -->|avg > 0.85| H[Demote tier-1]
        F -->|else| I[Keep tier]
    end

    subgraph Observability
        G --> J[routing_log write]
        H --> J
        I --> J
        J --> K[(tool-audit.db)]

        D --> L[PostToolUse hook]
        L --> M[(traces.db)]

        D --> N[cost-tracker parseAndTrack]
        N --> O[(costs.db)]
    end

    subgraph Token Optimization
        D --> P{LightRAG STEP 0}
        P --> Q[Query existing context]
        Q -->|Hit| R[Reduce re-discovery]
        Q -->|Miss| S[Normal dispatch]
    end

    K -.quality_score read path.-> F
    M -.corpus for Phase 3 distillation.-> T[Future: Fine-tune]
    O -.daily cost report.-> U[CEO visibility]
```

Task 0.1 — Mehanik Phase 2 Activation

MC: #9865
Owner: FlowForge
What: Activate Mehanik Phase 2 BLOCKING mode with 13-field marker schema enforcement.

Why

Single highest-leverage architectural fix. The `pre-dispatch-gate.sh` hook now enforces scope discipline at dispatch time, preventing the 11-agent scope-creep disasters that previously derailed

builds. Per MC #9223 root cause analysis, missing pre-dispatch validation allowed unbounded work expansion.

Changes

File: `~/.claude/hooks/pre-dispatch-gate.sh`
Line 72-79: Extended field validation loop from 4 fields to 13 fields (canonical schema).

13-Field Schema:

1. `timestamp:` — ISO8601 marker creation time
2. `task_id:` — MC task ID
3. `project_path:` — Absolute path to project root
4. `blueprint_read:` — Path to BUILD-BLUEPRINT.md or N/A
5. `deploy_map_read:` — Path to DEPLOY-MAP.md or N/A
6. `deploy_path_summary:` — One-line deploy mechanism
7. `ceo_item_count:` — CEO-authored items in plan
8. `approved_subtask_count:` — Approved subtask count
9. `ceiling:` — Scope ceiling (ceo_item_count + 2)
10. `approved_agents:` — Comma-separated agent list
11. `orchestration_surface:` — one-shot-Task | claude-chains | dag | pi-factory | cron
12. `tool_contract_required:` — true | false (research tasks)
13. `mehanik_session_id:` — Unique session identifier

7 BLOCK paths (exit 2):

- No MC task ID
- No Mehanik clearance marker
- Marker stale (>4h old)
- Missing required field
- Scope ceiling exceeded
- Research dispatch missing TOOL_CONTRACT
- Invalid marker format

Validation Results

Canary tests: 3/3 PASS

- Valid 13-field marker → exit 0 (CLEAR)
- No marker file → exit 2 (BLOCKED)
- Partial marker (5/13 fields) → exit 2 (BLOCKED on missing field)

Evidence: `/tmp/aif-v2-task-0.1-evidence.md`

Task 0.2 — LightRAG Container Restore

MC: #9866
Owner: FlowForge
What: Restore LightRAG main container (was missing from `docker ps`) and verify drain worker functionality.

Why

Vision 7 (LightRAG token saving) was at 0% realization because main container was down. Each day without deduplication costs Anthropic tokens that LightRAG should eliminate. 114K docs uploaded historically, but container absent since unknown date.

Before State

- LightRAG main container: MISSING
- Local health endpoint: UNREACHABLE (curl localhost:9621/health → timeout)
- Drain worker: LaunchAgent NOT LOADED
- Queue: 276 records in outbox

After State

- Container: HEALTHY (docker ps shows `lightrag`, Up 26s)
- Health endpoint: RESPONSIVE (http://localhost:9621/health)
- Pipeline status: `pipeline_busy: false`
- Neo4j: HEALTHY (Up 41h)
- Configuration verified:
  - LLM: ollama @ host.docker.internal:11434, model qwen3:8b-q8_0
  - Embedding: ollama @ host.docker.internal:11434, model bge-m3:latest
  - Graph: Neo4JStorage (bolt://neo4j:7687)
  - Vector: NanoVectorDBStorage (22,771 entities + 43,582 relationships loaded)
- Drain worker: FUNCTIONAL (manual execution, 276/276 processed)

Caveats

1. **LaunchAgent bootstrap failure** — Manual execution works, but `launchctl bootstrap` → I/O error. Drain worker runs manually until resolved.
2. **Platform mismatch** — Container image linux/amd64 on Apple Silicon (arm64), runs via Rosetta emulation.

3. **Health endpoint blocks during pipeline_busy** — Single-process design limitation; /health unavailable during active ingestion (follow-up task recommended).

Evidence: `/tmp/aif-v2-task-0.2-evidence.md`

Task 0.3 — Quality Score Read Path Wiring

MC: #9867
Owner: AgentForge
What: Wire `quality_score` read path in tier-router.js to enable feedback-informed routing.

Why

36,671 rows existed in legacy `agent-routing.db` with NULL `quality_score`. Wiring the read path closes Vision 2 (self-learning) and Vision 6 (free + paid models) with zero new data collection — routing decisions now adjust based on historical agent performance.

Schema Migration

Database: `~/system/databases/tool-audit.db`

Extended `routing_log` table with 4 new columns:

- `quality_score` REAL — Success metric (0.0 = failure, 1.0 = success)
- `caller_agent` TEXT — Calling agent name
- `target_tier` TEXT — Target tier before adjustment
- `mc_task_id` INTEGER — MC task reference

Implementation

Write Path:
Function `updateQualityScore(routingLogId, score)` at line 60.
Heuristic v1 (interim until Phase 1.4 eval harness):

- Task marked `ready` → 1.0
- Task orphaned → 0.5
- Task failed/blocked → 0.0

Read Path:
Function `getRecentQualityScores(callerAgent, targetTier)` at line 76.
Returns last 20 scores for {agent, tier} pair.

Tier Adjustment Logic:

- If ≥5 scores exist for {agent, tier}:
  - avg < 0.6 → escalate to tier+1 (e.g., tier 2 → tier 3)
  - avg > 0.85 → demote to tier-1 (e.g., tier 3 → tier 2)
  - else → keep current tier

Validation Results

Smoke test: 5/5 PASS

- Write path: 5 failures (quality_score=0.0) persisted
- Read path escalation: avg=0.00 → tier 2 escalated to tier 3
- Write path: 5 successes (quality_score=1.0) persisted
- Read path demotion: avg=1.00 detected (logic verified)
- Schema validation: all 4 columns exist

Legacy archive:
`agent-routing.db` renamed to `agent-routing.db.legacy-archive-2026-04-27` (36,671 rows, 3.5MB).
Not migrated — does not reflect current routing reality.

ADR: `/Users/makinja/system/specs/adr/ADR-quality-score-read-path.md`
Evidence: `/tmp/aif-v2-task-0.3-evidence.md`

Task 0.4 — Cost Telemetry Blind Spot Fix

MC: #9868 (existing, now resolved)
Owner: CodeCraft
What: Backfill claude-cli cost data for 2026-04-17 → 2026-04-24 and add real-time stderr parser.

Why

Week magnitude cost was invisible. `node ~/system/tools/cost-tracker.js summary today` showed $0 for 967 claude-cli requests. Cannot optimize without measurement. This blocked all routing optimization work.

Before State

- claude-cli rows in range: 27 rows, ALL $0.00
- Root cause: Stop hook only started logging sessions with token data from 2026-04-24

Backfill Results

Script: `~/system/tools/backfill-claude-cli-costs.js`

- Files processed: 21 session transcripts (2026-04-17 to 2026-04-24)
- Sessions inserted: 19
- Sessions already in DB: 2 (skipped — idempotent)
- Total cost backfilled: $41.46
- Model: claude-sonnet-4-6 (all sessions)
- Pricing: cache_write=$3.75/MTok, cache_read=$0.30/MTok, input=$3/MTok, output=$15/MTok

Week Total (2026-04-27)

- Total requests: 711
- Total cost: **$163,223.11**
- claude-cli: 671 req, $163,223.11
  - claude-opus-4-7: 636 req, $163,182.96
  - claude-sonnet-4-6: 29 req, $40.15

Magnitude: $163K/week aligns with OpenAI lens estimate ($162,945/wk).

Real-Time Capture

Added to `cost-tracker.js`:

- `parseAndTrack(stdoutJson, opts)` — Parse --output-format json output, track cost
- `parseStderrLine(line, opts)` — Parse individual stderr line, idempotent

Daily Cron

Script: `~/system/tools/cost-daily-report.sh`
LaunchAgent: `~/Library/LaunchAgents/com.alai.cost-daily-report.plist`
Schedule: 23:55 daily
Output: `~/system/reports/cost-daily.md`

Evidence: `/tmp/aif-v2-task-0.4-evidence.md`

Task 0.5 — Trace Capture Pipeline

MC: #9869
Owner: AgentForge
What: Add PostToolUse hook that captures per-dispatch metadata to traces.db for future distillation and fine-tuning.

Why

Every agent run currently exits and disappears. Trace capture creates a passive corpus that gates ALL future AI Factory learning: distillation (Phase 2), eval harness (Phase 1.4), and fine-tuning (Phase 3). Without this, Vision 8 (own fine-tuned model) remains at 0%.

Database Schema

Location: `~/system/databases/traces.db`

14 fields:

1. `id` — Primary key
2. `timestamp` — DATETIME DEFAULT CURRENT_TIMESTAMP
3. `task_id` — MC task ID
4. `agent` — Subagent type or "john"
5. `session_id` — Join key to costs.db
6. `tool_name` — Agent, Bash, Read, Write, Edit
7. `prompt_hash` — SHA256(tool_input), 16-char prefix
8. `response_hash` — SHA256(tool_response), 16-char prefix
9. `duration_ms` — Tool execution time
10. `exit_code` — 0=success, 1=error, 2=blocked
11. `model` — Model used (if Agent)
12. `tokens_in` — Input tokens
13. `tokens_out` — Output tokens
14. `cost_usd` — Computed cost

7 indexes: timestamp, agent, model, tool_name, prompt_hash, session_id, task_id

PostToolUse Hook

Location: `~/.claude/hooks/trace-capture.py`
Language: Python 3 (fast JSON parsing, sqlite3 stdlib)
Registered: `~/.claude/settings.json` PostToolUse hooks array (async: true)

Key features:

- Fire-and-forget (always exit 0 per ZAKON PI2)
- Privacy-preserving (only hashes, no raw prompts/responses)
- MC task ID extraction via regex
- Session ID from env or date fallback
- Error handling: logs to stderr, never blocks tool execution

Latency Measurement

Method: 10-iteration synthetic hook call

Results:

- Average: 45ms
- Budget: <50ms
- Status: PASS (10% under budget)

Privacy Posture

CRITICAL: No raw prompts or responses stored in traces.db.

Method:

1. SHA256 hash of full tool_input
2. SHA256 hash of full tool_response
3. Store only 16-char hex prefix (collision-resistant for corpus size)
4. Original content never persists

Rationale:

- Prevents PII leakage (credentials, API keys, personal data)
- Enables duplicate detection
- Supports eval harness (hash matching for golden tasks)
- Future fine-tuning uses hashes as index, not content

Smoke Test Results

Test 1: Row insertion — +10 rows captured (PASS)
Test 2: Privacy validation — 0 raw prompts/responses stored (PASS)
Test 3: Schema integrity — All 14 fields populated correctly (PASS)

Live integration: 64 rows captured during Proveo validation.

Evidence: `/tmp/aif-v2-task-0.5-evidence.md`

Caveats & Follow-ups

From Proveo Validation (MC #9870)

1. **LightRAG health endpoint blocks during pipeline_busy**
   - Root cause: Single-process design (no separate health worker)
   - Impact: `/health` unavailable during active ingestion
   - Recommendation: Separate health check process or async health handler
   - Severity: LOW (operational monitoring gap, not functional block)
2. **Hash prefix length (16-char) may need adjustment at scale**
   - Current corpus: 64 rows (negligible collision risk)
   - At 100K rows: <0.01% collision probability
   - Recommendation: Monitor at 10K rows, extend to 24-char if needed
   - Severity: LOW (future consideration)
3. **Table name typo in smoke test**
   - Test script referenced `routing_logs` (wrong), actual table `routing_log`
   - Impact: None (test passed via fallback query)
   - Resolution: Fixed in final evidence file
   - Severity: TRIVIAL
4. **Row count delta across validation runs**
   - Different smoke test runs show varying baselines (304 vs 337 rows)
   - Root cause: Multiple validation passes appending to same DB
   - Impact: None (idempotent inserts verified)
   - Severity: TRIVIAL

How To Verify

Run these commands to validate Phase 0 backbone functionality:

Task 0.1 — Mehanik Gate

```
# Verify 7 exit-2 block paths exist
grep -c "exit 2" ~/.claude/hooks/pre-dispatch-gate.sh
# Expected: 7

# Test BLOCK path (no marker)
MC_TASK_ID=9999 ~/.claude/hooks/pre-dispatch-gate.sh
```

```
# Expected: exit 2, error message

# Test ALLOW path (valid marker)
# (Requires /mehanik clearance file in /tmp/)
MC_TASK_ID=9865 ~/.claude/hooks/pre-dispatch-gate.sh
# Expected: exit 0
```

Task 0.2 — LightRAG

```
# Verify container running
docker ps | grep lightrag
# Expected: 2 containers (lightrag, lightrag-neo4j)

# Verify health endpoint
curl -s http://localhost:9621/health | jq .
# Expected: {"pipeline_busy": false, ...}

# Check vector/graph load
docker logs lightrag 2>&1 | grep "Loaded"
# Expected: 22,771 entity vectors, 43,582 relationship vectors
```

Task 0.3 — Quality Score

```
# Verify schema extended
sqlite3 ~/system/databases/tool-audit.db ".schema routing_log"
# Expected: quality_score, caller_agent, target_tier, mc_task_id columns

# Check non-NULL quality scores
sqlite3 ~/system/databases/tool-audit.db \
  "SELECT COUNT(*) FROM routing_log WHERE quality_score IS NOT NULL"
# Expected: >0 (any recent dispatches)

# Verify legacy DB archived
ls -lh ~/system/databases/agent-routing.db.legacy-archive-2026-04-27
# Expected: 3.5MB file
```

Task 0.4 — Cost Telemetry

```
# Verify today's cost non-zero
node ~/system/tools/cost-tracker.js summary today | grep claude
# Expected: $>0 for claude-cli

# Verify week magnitude
node ~/system/tools/cost-tracker.js summary week
# Expected: ~$163K total

# Verify daily report cron loaded
launchctl list | grep cost-daily-report
# Expected: com.alai.cost-daily-report with PID or status 0
```

Task 0.5 — Trace Capture

```
# Verify traces.db exists and has rows
sqlite3 ~/system/databases/traces.db "SELECT COUNT(*) FROM traces"
# Expected: >10 (grows with each dispatch)

# Verify hook registered
grep -A3 "trace-capture.py" ~/.claude/settings.json
# Expected: PostToolUse hook entry with async:true

# Verify privacy (no raw content)
sqlite3 ~/system/databases/traces.db \
  "SELECT prompt_hash, response_hash FROM traces LIMIT 5"
# Expected: Only 16-char hex strings, no full text
```

References

Parent Plan

- **AI Factory v2 Full Plan:** `/Users/makinja/system/specs/ai-factory-v2-plan.md`
- **CEO Approval:** 2026-04-27 (option B, override DA-BLOCKED + triage-mode)

Lens Reports (5 expert convergent analysis)

- `/tmp/ai-factory-v2-petter.md` — Architecture (Petter Graff)
- `/tmp/ai-factory-v2-anthropic.md` — Token economics (Anthropic Chief AI Architect)
- `/tmp/ai-factory-v2-openai.md` — Multi-provider/distillation (OpenAI Chief Architect)
- `/tmp/ai-factory-v2-alem-clone.md` — CEO reality check (Alem-Clone)
- `/tmp/ai-factory-v2-da.md` — Risk audit (Devil's Advocate)

Root Cause Analysis

- **MC #9223 Final Synthesis:** Mehanik Phase 2 architectural decision
- **Scope creep incident 2026-04-24:** 11-agent dispatch without gate (pre-Mehanik)

Architecture Decision Records

- **ADR — Quality Score Read Path:** `/Users/makinja/system/specs/adr/ADR-quality-score-read-path.md`

Evidence Files

- `/tmp/aif-v2-task-0.1-evidence.md` — Mehanik Phase 2 activation
- `/tmp/aif-v2-task-0.2-evidence.md` — LightRAG container restore
- `/tmp/aif-v2-task-0.3-evidence.md` — Quality score integration
- `/tmp/aif-v2-task-0.4-evidence.md` — Cost telemetry backfill
- `/tmp/aif-v2-task-0.5-evidence.md` — Trace capture pipeline
- `/tmp/aif-v2-task-0.8-evidence.md` — This documentation task

Proveo Validation

- **MC #9870:** Cross-validation of all 5 builder tasks (COMPLETE)

Next Steps

Immediate (Phase 0 closure)

1. Proveo validates this BookStack page exists and is discoverable
2. John marks MC #9870 and #9871 done
3. Phase 0 declared COMPLETE

Phase 1 — Token Economics Wiring (Post-2026-05-02)

Gate: CEO must explicitly close triage mode before Phase 1 begins.

6 tasks planned:

1. Anthropic prompt caching wire-up (50-70% input token reduction)
2. Sub-agent context isolation (prevents 7M token bleed)
3. LightRAG STEP 0 injection in 8 active agents
4. Eval harness with 25 golden tasks (gates all future routing changes)
5. Multi-provider fallback chain (Groq adapter wire-up)
6. Proveo E2E + Skillforge docs (ZAKON PLAN mandatory)

Expected savings: $144-240/week conservative (prompt caching alone). Upper bound: $14,778/week (sub-agent isolation).

Phase 2 — Capability Expansion (Weeks 2-4)

Gate: Phase 1 must show measurable token savings (≥$3K/week) AND eval harness green.

7 tasks planned:

- AutoCoder.js Phase 1 (dry-run mode)
- ANVIL SPOF: replicate 13 P0 databases to Azure
- MCP tool schema portability
- Distillation candidate scoring
- Archive 44 orphan agents
- TTL sweep on hivemind.db
- Phase 2 Proveo E2E + Skillforge docs

Phase 3 — Strategic Horizon (Q3 2026+)

Gate: ALAI must have ≥1 paid AI Services engagement closed.

5 tasks planned:

- Fine-tune candidate review
- AIOS competitor evaluation (Cursor, Devin, OpenAI Operator, Gemini Extensions)
- Operator-style browser agents
- Anti-lying enforcement hooks
- Multimodal expansion (Realtime API, OCR)

Last Updated: 2026-04-27
Maintained By: ALAI
Document Version: 1.0
BookStack Path: Engineering / AI Factory v2 — Phase 0 Backbone

---

Revision #2
Created 2026-04-27 23:12:56 UTC by John
Updated 2026-05-31 20:06:37 UTC by John