SG-HIMS — AAMO (Automated AI Memory Optimization) Implementation Guide

Version: V1.0
Scope: Adds a self-learning loop to SG-HIMS that analyzes recent telemetry, proposes safer RAM-pressure thresholds, and updates policy over time (with guardrails and cooldown).
This document provides the full implementation procedure for AAMO, including prerequisites, integration, learning logic, safety checks, and operational workflow.

0) Prerequisites

Ensure SG-HIMS core and Python venv are active. Required files include sg_hims.py, sg_hims_daemon.py, sg_utils.py, and policies.yaml.

1) Files Added by AAMO

aamo_learning.py performs statistical learning (p95 window, headroom, safety clamps).

2) Daemon Integration

Integrate AAMO via sg_hims_daemon.py loop; it triggers hourly learning and logs to sg_hims.jsonl.

3) CLI Operations

Enable manual control through python sg_hims.py learn balanced.

4) Algorithm Logic

Uses p95 of RAM ratio, adds headroom, and applies safety clamps to thresholds.

5) Power Awareness

Adapts thresholds dynamically depending on AC/Battery state with 5% tighter limits on battery.

6) Dashboard Integration

SG-HIMS dashboard visualizes current RAM/SWAP metrics, thresholds, and last AAMO update.

7) Logs & Artifacts

All telemetry, learning logs, backups, and metrics are stored in the logs/ directory.

8) Operation Procedure

Run daemon → collect data → learn → auto-update policies on safe intervals.

9) Safety & Rollback

Automatic YAML backups and cooldown management ensure safety and recoverability.

10) Troubleshooting

Includes checks for data sufficiency, cooldown, and macOS-specific permissions.

11) Design Philosophy

Combines p95 learning + headroom for stability; bounds thresholds between 70–90% (High) and ≤98% (Critical).

12) Roadmap Hooks

Future updates: evidence-based gates, per-mode learning, and policy intent awareness.

13) Quick Commands

Core CLI shortcuts for start, dashboard, learning, log tailing, and policy restoration.

— Safeway Guardian R&D; / TRINITY AI Integration Team