Ollama Fleet Architecture

Ollama Fleet Architecture

MC: #8522 | **Related:** #8477 (triage preload), #8471 (vault-keeper watchdog), #8472 (YouTube daemon fix) | **Date:** 2026-04-20

Overview

ALAI operates a two-node Ollama fleet: ANVIL (local dev Mac) and FORGE (Ubuntu 22.04 GPU workstation). ANVIL handles triage workloads (email, TLDR, quick classification), FORGE handles heavy inference (32B+ models, RAG pipelines).

ANVIL Ollama Configuration

Capacity Limits

- **MAX_LOADED_MODELS:** 2 (prevents RAM exhaustion)
- **KEEP_ALIVE:** 30s (default for on-demand models)
- **Hardware:** M1 Pro, 32GB RAM, 5GB reserved for triage model

LaunchAgent: com.alai.ollama-serve-v2

```
Label: com.alai.ollama-serve-v2
Plist: ~/Library/LaunchAgents/com.alai.ollama-serve-v2.plist
Port: 11434
Environment:
  OLLAMA_FLASH_ATTENTION=1
  OLLAMA_KV_CACHE_TYPE=q8_0
  OLLAMA_MAX_LOADED_MODELS=2
  OLLAMA_KEEP_ALIVE=30s
```

Triage Preload Pattern

MC #8477 — Prevent qwen2.5-coder:32b (23GB) from blocking email/TLDR triage.

Strategy

Preload `llama3.1:8b` with `keep_alive=-1` (indefinite) so it's always resident for fast triage operations. 5GB footprint.

LaunchAgent: com.john.ollama-triage-preload

```
Label: com.john.ollama-triage-preload
Script: ~/system/tools/ollama-triage-preload.sh
Trigger: RunAtLoad + StartInterval 300s (every 5 min)
Log: ~/system/logs/ollama-triage-preload-stdout.log
```

Script Logic (ollama-triage-preload.sh)

1. Check if `llama3.1:8b` is already loaded via `/api/ps`
2. If not loaded, send minimal prompt with `keep_alive=-1`
3. Log success/skip

```
curl -sf -X POST "$OLLAMA_URL/api/generate" \
  -H "Content-Type: application/json" \
  -d "{
    \"model\": \"llama3.1:8b\",
    \"prompt\": \"ready\",
    \"stream\": false,
    \"keep_alive\": -1,
    \"options\": {
      \"num_predict\": 1
    }
  }"
```

Model Tier System

Tier	Model	Size	Use Case	Keep Alive	Node
Triage	llama3.1:8b	5GB	Email classification, TLDR summarization, quick routing	-1 (indefinite)	ANVIL
Heavy	qwen2.5-coder:32b	23GB	Code generation, architecture review, complex reasoning	30s (on-demand)	ANVIL
Primary	devstral:24b	~15GB	Agent orchestration, planning, context routing	300s	FORGE

FORGE Failover

Consumers (email-agent.js, tldr-briefing.js, YouTube daemon) can set `FORGE_FIRST=0` environment variable to skip FORGE and use ANVIL directly.

```
# Force ANVIL-only
export FORGE_FIRST=0
node ~/system/daemons/youtube-daemon.js
```

Default behavior: Try FORGE (10.0.0.2:11434), fallback to ANVIL (localhost:11434) on timeout.

Vault-Keeper Watchdog (MC #8471 — PENDING)

Monitors `~/system/.cache/vault-keeper-heartbeat` file. If stale > 1 hour, SENTINEL alerts.

Implementation

```
LaunchAgent: com.john.vault-keeper-watchdog
Interval: 600s (10 min)
Script: ~/system/daemons/vault-keeper-watchdog.sh
Alert: Slack #sentinel-alerts
```

Logic

1. Read heartbeat file timestamp
2. Compare with current time
3. If > 3600s, send SENTINEL alert with vault-keeper logs

YouTube Daemon Lesson (MC #8472)

Log redirection corruption: `tee` + subshell arithmetic capture caused output mangling.

Anti-Pattern

```
# WRONG — tee inside $() breaks arithmetic
NEW_COUNT=$(node ~/system/daemons/youtube-processor.js | tee -a "$LOG")
```

Correct Pattern

```
# RIGHT — separate logging stream
node ~/system/daemons/youtube-processor.js >> "$LOG" 2>&1
```

LaunchAgent Duplication

Never use both `KeepAlive` and `StartInterval` in same plist. `StartInterval` triggers even if process is still running, causing overlap.

```
# WRONG
<key>KeepAlive</key>
<true/>
<key>StartInterval</key>
<integer>3600</integer>

# RIGHT (pick one)
<key>StartInterval</key>
<integer>3600</integer>
```

Fleet Monitoring

ANVIL

```
curl http://localhost:11434/api/ps
curl http://localhost:11434/api/tags
tail -f ~/system/logs/ollama-triage-preload-stdout.log
```

FORGE

```
curl http://10.0.0.2:11434/api/ps
ssh forge "tail -f /var/log/ollama.log"
```

Mission Control

```
node ~/system/tools/mc.js list --tag ollama
node ~/system/tools/cost-tracker.js summary --service ollama
```

Generated by Skillforge | ALAI, 2026

Revision #2
Created 2026-04-20 19:06:04 UTC by John
Updated 2026-05-31 20:06:17 UTC by John