

NEWINGTON-AI

Scanner Summary Service System Architecture, Operations & Improvement Roadmap

PUBLIC EDITION
Prepared October 16, 2026. This report describes the project as developed during September–October 2026. Historical observations are identified; live server state has not been independently audited.

A public-safety radio monitoring and AI summarization platform for NewingtonWeather.com

SYSTEM SNAPSHOT	DOCUMENTED CONFIGURATION
Operational goal	Around-the-clock Newington police, fire and EMS monitoring
Public incident window	3 hours; configured maximum 30 entries in the documented web implementation
Processing environment	Ubuntu Linux; Intel Core i7-12700T (12 cores / 20 logical processors)
Core technologies	FFmpeg, voice detection, OpenAI API, Python, PHP, JavaScript, systemd
Processing version discussed	V36; stream failover and recovery logic
Summary model observed	gpt-6-sol (October 2026 diagnostics)

“What were all those sirens for?”

Newington-AI turns fragmented radio transmissions into short public explanations of recent calls for service. The service combines audio acquisition, speech detection, hosted speech recognition, language-model analysis, duplicate handling, privacy checks, and website publication. It is designed to be informative rather than an authoritative emergency incident record.

01 Purpose and design objectives

The service reduces the need for residents to monitor live scanner traffic continuously. It identifies potentially relevant dispatches, converts them into readable incident descriptions and publishes them alongside the existing live audio player.

- Understand local radio traffic: interpret radio codes, place names, responding units and short or clipped exchanges.
- Publish meaningful information: suppress routine acknowledgements, status messages and radio noise.
- Control costs: limit avoidable AI calls while preserving accuracy and timeliness.
- Remain resilient and responsible: restart automatically, fail over between sources, generalize personal information and enable human correction.

02 End-to-end architecture

STAGE	FUNCTION
1. Audio source	Primary scanner audio stream; alternate source for failover.
2. Audio decoding	FFmpeg receives and normalizes compressed audio.
3. Voice detection	Speech-activity logic separates likely transmissions from silence/static.
4. Speech-to-text	OpenAI-hosted transcription generates provisional text.
5. AI analysis	A language model classifies traffic, interprets codes and drafts summaries.
6. Incident resolution	Matching logic attempts to join updates, prevent duplicates and protect manual edits.
7. Publication	Authenticated upload to PHP receiver; summaries appear in responsive scanner interface.

The stages should be monitored individually: a valid network connection does not prove that a dispatch was heard, and a completed AI request does not prove the public website received its result.

03 Audio sources and failover

SOURCE	PUBLIC-EDITION DETAIL
Primary	Direct stream URL intentionally withheld. Primary live scanner audio input.
Backup	Direct stream URL intentionally withheld. Alternate scanner audio input.
Public listener reference	Broadcastify feed listing: https://www.broadcastify.com/listen/feed/1813

To avoid encouraging direct access to operational audio endpoints, this edition omits source hostnames, paths and direct streaming URLs. Only the intended public listener page is supplied.

V36 introduced connection failure detection, backup switching, periodic primary checks, and a three-successful-check requirement before restoring the primary source. Email notifications report outages and recoveries.

OBSERVED EVENT (OCTOBER 16)	TIME
Backup activated	3:15:55 AM
Recovery of backup recorded	3:18:06 AM
Primary restored following checks	3:20:04 AM

This episode confirmed source-switch behavior, but the examined interval produced no voice detections, audio clips or transcripts. The next reliability upgrade should detect silent or invalid audio even when HTTP connections appear healthy.

04 Hosting and process management

Early experiments used a Windows workstation, OBS, LocalVocal, a PowerShell watcher and a private PHP test area. Local Whisper experiments included large-v3 with INT8 quantization and CPU threading. The later architecture moved toward Ubuntu, FFmpeg, Python orchestration, OpenAI-hosted transcription and systemd-managed background services.

The processing computer was documented as an Intel Core i7-12700T system with 12 physical cores and 20 logical processors and no dedicated GPU. Cloud transcription reduces local inference load. systemd supports boot-time startup, restart behavior and service logging, reducing reliance on an interactive Windows session.

05 Audio preparation and speech detection

FFmpeg decodes the network stream and prepares audio for speech recognition. Earlier testing normalized segments to 16 kHz mono. The pipeline must handle short dispatches, clipping, static, signalling tones, overlapping traffic and long quiet periods.

During testing, noise events sometimes held recordings open for 30–45 seconds. A revised voice detector used repeated speech-positive frames, including a tested threshold of three positive 30 ms frames within an approximately 256 ms chunk, to reduce phantom captures.

Recommended: configurable pre-roll/post-roll buffering, metrics for rejected segments, and automated audio-level and decoded-speech heartbeat checks. Threshold changes should be tested on archived traffic to avoid losing short transmissions.

06 Hosted AI transcription

Likely speech segments are sent to OpenAI speech-to-text processing. The transcript is a fallible estimate rather than a verified record. Dispatchers speak rapidly, use abbreviated unit language and local 10-codes, and may be affected by weak reception or mixed audio.

The exact hosted transcription model identifier has not been independently confirmed from the currently installed V36 configuration. A model ID should therefore be added to a versioned operational inventory instead of inferred from prior experiments.

- Maintain a verified local vocabulary of streets, schools, parks, agencies, businesses and radio terms.
- Use vocabulary context where supported, and apply conservative corrections after transcription.
- Keep original transcript text and any corrected form distinct for troubleshooting.
- Score accuracy on manually reviewed samples, including calls with weak signal and very short dispatches.

07 GPT classification and summarization

The October 2026 diagnostics identify gpt-6-sol as the summary/publication model. The model evaluates transcript content to decide whether to ignore traffic, open an incident, or update an existing one, then drafts concise public wording.

EXAMPLE CODE	LOCAL INTERPRETATION	EXPECTED HANDLING
10-30	Motor vehicle accident	Create an accident incident when dispatch context supports it.

EXAMPLE CODE	LOCAL INTERPRETATION	EXPECTED HANDLING
10-31	Motor vehicle accident with injuries	Preserve injury qualification.
10-2	Headquarters	Usually location/status context.
10-3	Scene under control	Usually update an existing incident.
10-8	Finished / dismissed	Generally routine; avoid a new entry.

These are project-specific meanings established in development; radio-code definitions vary by jurisdiction. The AI must not use unrelated generic meanings. A report of smoke must remain a report, not become a confirmed structure fire without corroborating radio traffic.

By V35, development had addressed police pursuit handling, EMS matching, alarm descriptions, uncertain locations and controlled retries of incomplete AI responses. Model output allowances and retry behavior should be measured to distinguish useful recovery from redundant spending.

08 Incident identity, matching and updates

One event may generate a dispatch, acknowledgement, arrival, request for additional units and final clearance. The service attempts to combine related transmissions and avoid publishing six versions of one incident. This remains difficult when street names are misheard, similar calls occur close together or departments describe the same event differently.

PROPOSED INCIDENT FIELD	WHY IT MATTERS
Stable internal ID	Allows updates without matching on summary wording alone.
First heard / last updated	Supports timing and three-hour retention.
Type, location and agencies	Assists conservative incident matching.
Status and confidence	Separates dispatch, arrival, outcome and ambiguity.
Transcript evidence references	Preserves a reviewable basis for each public statement.
Human lock / revision history	Stops model updates from silently replacing corrections.

New transcript fragments should be matched against recent active records using time, type, geographic evidence and linguistic similarity. Ambiguous matches should favor preserving two separate records or seeking review over confidently merging unrelated incidents.

09 Web publication and visitor experience

The original private web test lived under /beta/scanner-ai/ and used authenticated upload, PHP storage/feed files, and a responsive player and incident list. Key implementation responsibilities included receive-summary.php, summary-storage.php, summary-feed.php, config.php and index.php. Manual editing tools were added to correct or hide entries.

SETTING	DOCUMENTED VALUE
SCANNER_SUMMARY_TTL_SECONDS	10800 (3 hours)
SCANNER_MAX_SUMMARIES	30
Public scanner detail page	https://www.broadcastify.com/listen/feed/1813
NewingtonWeather archive (ProScan)	https://newingtonweather.com/scanner-archives/

The web panel preserves the audio player and scrolling summaries for mobile layouts. Visitor disclosures explain that AI can mishear locations, miss calls or be wrong, and that radio information may be unconfirmed. The feedback address is scannerfeed@newingtonweather.com.

Deployment check: the October 15 review reportedly showed V36 processor behavior while the web receiver still identified itself as V35. Validate server/client schema compatibility and installed versions before treating the deployment as synchronized.

10 Privacy, accuracy and responsible publication

Public summaries are intended to omit personal names and generalize precise residential addresses. The live radio player remains a separate, unedited broadcast and may contain information not reproduced in written summaries.

- Introduce a deterministic privacy validation pass immediately before public publication.
- Separate “reported,” “dispatched,” “on scene,” and “confirmed” facts in incident status and wording.
- Minimize retained sensitive source data, secure editor access, and audit human changes.
- Ensure public documentation excludes internal endpoints, secrets, authentication tokens and hidden upload routes.

11 Workload and operating cost

Historical analysis for September 30–October 5 showed approximately 324–413 AI requests daily, often 9,000–11,000 input tokens per request, with 87–92% cached input. One October 5 measure was about 66 output tokens per request. The overall observed cache hit rate was approximately 89.6%.

HISTORICAL INDICATOR	OBSERVATION
Daily AI requests	~324 to 413
Input tokens per request	~9,000 to 11,000
Cached input	~87% to 92%; ~89.6% overall observed
Output tokens per request	~66 in one Oct. 5 review
Historical API spending pace	~\$65-\$70 per month, estimated from early October billing

The monthly figure is a historical projection, not verified current expenditure. It should not be treated as a guarantee, and transcription versus GPT analysis spending were not separately established in that measurement.

- Skip clear non-incidents before expensive model calls without suppressing genuine emergencies.
- Send only the most relevant active incident context rather than expanding the prompt indefinitely.
- Track transcription, analysis, cache usage, retries and publication failures independently.
- Benchmark cheaper first-pass classification against a labeled evaluation set before relying on it.

12 Observed reliability and limitations

An October 15 review reported an examined run containing 17 calls, two updates, one manual-edit protection event and no errors in that particular view. This is useful evidence, but not a formal validation of uptime, accuracy or complete delivery.

- Transport-level health checks can miss connected-but-silent streams.
- Speech recognition mistakes can turn into confident, misleading summaries.
- Text-based duplicate matching may split one call or merge unrelated calls.
- Successful processing does not guarantee successful website publication.
- Receiver version drift can produce incompatible data formats.

13 Prioritized improvement roadmap

PRIORITY	CHANGE	EXPECTED RESULT
1 - High	End-to-end audio + publication heartbeat	Detect failures sooner.
2 - High	Local code/vocabulary registry + test corpus	Improve dispatch interpretation.
3 - High	Persistent incident IDs + evidence-backed merging	Reduce duplicates and accidental merges.
4 - Medium	Cost instrumentation + smart AI call selection	Lower unnecessary spend.
5 - Medium	Confidence-aware publication + privacy validator	Reduce misleading or identifying output.
6 - Later	Private searchable incident history	Support reviews and quality measurement.

Suggested evaluation metrics

- Capture: decoded-audio continuity, voice segments/hour, missed known dispatches.
- Transcription: location and radio-code error rate on verified samples.
- Classification: meaningful calls detected, false postings, unnecessary calls.
- Incident matching: duplicate rate, false merges and update correctness.
- Publishing: capture-to-display latency, receiver errors and backlog.
- Costs: API spend/day, cost per published incident and retry percentage.

14 Next-generation design recommendation

The most pragmatic next step is to evolve, not rewrite, the present service. Keep the Linux ingestion pipeline, OpenAI integration and PHP presentation layer, while introducing more explicit logical boundaries between audio capture, AI intelligence, incident storage, and monitoring/publication.

LOGICAL SERVICE	RESPONSIBILITIES
Audio	Stream connections, switching, VAD, capture, local buffering.
Intelligence	Speech-to-text, dispatch interpretation, incident classification and summary draft.
Incident	Stable IDs, update history, manual locks, evidence and privacy checks.
Web + monitoring	Public feed, secured editor, metrics and operational alerts.

A small durable queue can preserve audio segments during transient API outages or throttling. Versioned event schemas and idempotent upload IDs can prevent duplicates after retries. A fixed

library of reviewed real-world examples should be replayed before adopting new prompts, model versions or matching rules.

15 Overall assessment and next actions

Newington-AI has evolved from a Windows-based experiment into a Linux-hosted, API-powered service that automatically transforms scanner traffic into a public three-hour activity summary. Its strengths include useful incident interpretation, automatic publication, manual corrections and improved stream redundancy.

Recommended sequence: (1) verify deployed versions and secrets; (2) add decoded-audio and website-delivery health monitoring; (3) create a labeled set of actual calls; (4) improve radio vocabulary and duplicate matching against that set; (5) add per-stage usage and cost reporting.

The primary success criterion is not writing polished prose. It is whether the published account is timely, appropriately cautious, privacy-respecting and actually supported by the radio transmission.

Document basis: Project design notes, source files and operational measurements discussed September–October 2026. Technical values are historical observations; no live credentials or secret upload tokens are provided.