Companion

A Private Offline AI Companion for Older Adults

Version: 2.0 (Evidence-Based Revision) **Supersedes:** v1.1 (Revised Development Pipeline)

What changed in v2.0, and why

v1.1 was written before any code existed. Since then, a complete software prototype has been built and validated on a Mac (`GemmaMemoryKit` , nine ordered milestones, all passing — see `Docs/Gemma_4_E2B_Persistent_Memory_Guide.md` and `PROGRESS.md`). That work produced **measured numbers**, and several of them contradict assumptions in v1.1.

This revision propagates those measurements back into the plan. The substantive changes:

#	v1.1 said	Measurement or finding	v2.0 says
1	Response latency <2–3s	1.81 tok/s CPU-only on an M5 Pro; published Pi-class figures are 2–5 tok/s for a 3–4B Q4 model. A 40-token reply is **8–20 seconds** of generation.	Metric replaced with **time-to-first-audio <2s** via sentence-streamed TTS. Total reply time is explicitly *not* the target.
2	RPi CM4 / Jetson Nano	Both are a generation behind; inference is memory-bandwidth-bound, and Jetson Nano is 4GB Maxwell-era and effectively EOL.	Shortlist is **Pi 5 / CM5** and **RK3588-class** boards.
3	Fine-tune a 3–7B model for elderly conversation (6–8 weeks, $2–5K)	A loadable personality file plus a stacking preference store already produces measurably different, appropriate tone — validated at Milestone 6 and extended since.	Fine-tuning **deferred** to post-pilot, contingent on prompting demonstrably failing.
4	Camera + face recognition in v1	The stated use case ("recognize when Sarah calls") conflates audio calls with visual recognition. A camera in a plush toy is the single largest trust cost in the design.	**No camera in v1.** Optional voice-based speaker identification instead.
5	Emotional State Engine flags withdrawal/sadness	Already flagged internally as high-stakes (issue #9, tagged `[NEEDS SCOPING]`).	Split into **tone adaptation** (in scope) and **well-being signals**

#	v1.1 said	Measurement or finding	v2.0 says
			(deferred, separate consent and safety design).
6	>8h continuous conversation, 2,000–5,000 mAh	8h at <10W needs 40–64 Wh; a 5,000 mAh 7.4V pack is ~37 Wh. Internally inconsistent.	Spec restated against a **realistic duty cycle**, not continuous generation.
7	~8 months to production, $12–27K non-salary	No line items for certification, plush tooling, or eldercare-adjacent legal review. Physical consumer hardware for a vulnerable population.	**~18–24 months**, with an honest budget and a hardware-free interim product.
8	Hardware prototype is the first real user contact	The already-built iOS path (guide Section 10) can reach real users **months** before any hardware exists.	New **Stage B: handheld field prototype** — validates the product hypothesis at near-zero hardware risk.

The single most important structural change is #8. v1.1 treated "get it in front of older adults" as something that happens after the plush toy exists. It doesn't have to. That reordering removes most of the schedule risk from the program.

Executive Summary

Companion is a privacy-first AI companion embedded in a comforting stuffed animal. It operates entirely offline during normal use: every conversation, memory, and decision happens locally on the device.

Companion's purpose is not to search the internet or answer every question. Its purpose is to become a trusted lifelong companion that listens, remembers, and protects the owner's personal history — gradually learning their life story, remembering important people and events, and offering unhurried emotional support.

Companion is designed to **encourage connection with family and friends, not replace them**. This is a load-bearing design principle, not a marketing line; it is the tiebreaker for feature decisions throughout this document.

Current status: the software core is built and validated. A memory-backed, personality-driven, voice-enabled assistant runs today on Apple hardware with persistent semantic recall across process restarts. What remains is a product problem (does this help real people?) and an embedded-systems problem (does it run acceptably on a $60 board inside a plush toy?) — in that order.

Audience & Scope

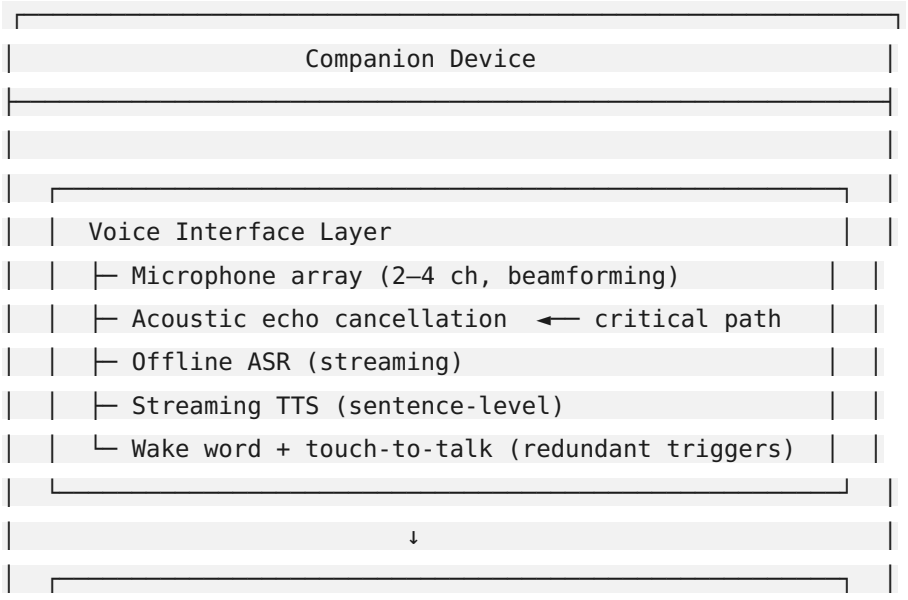
Intended for product developers and engineers, privacy researchers, stakeholders and investors, and eldercare professionals.

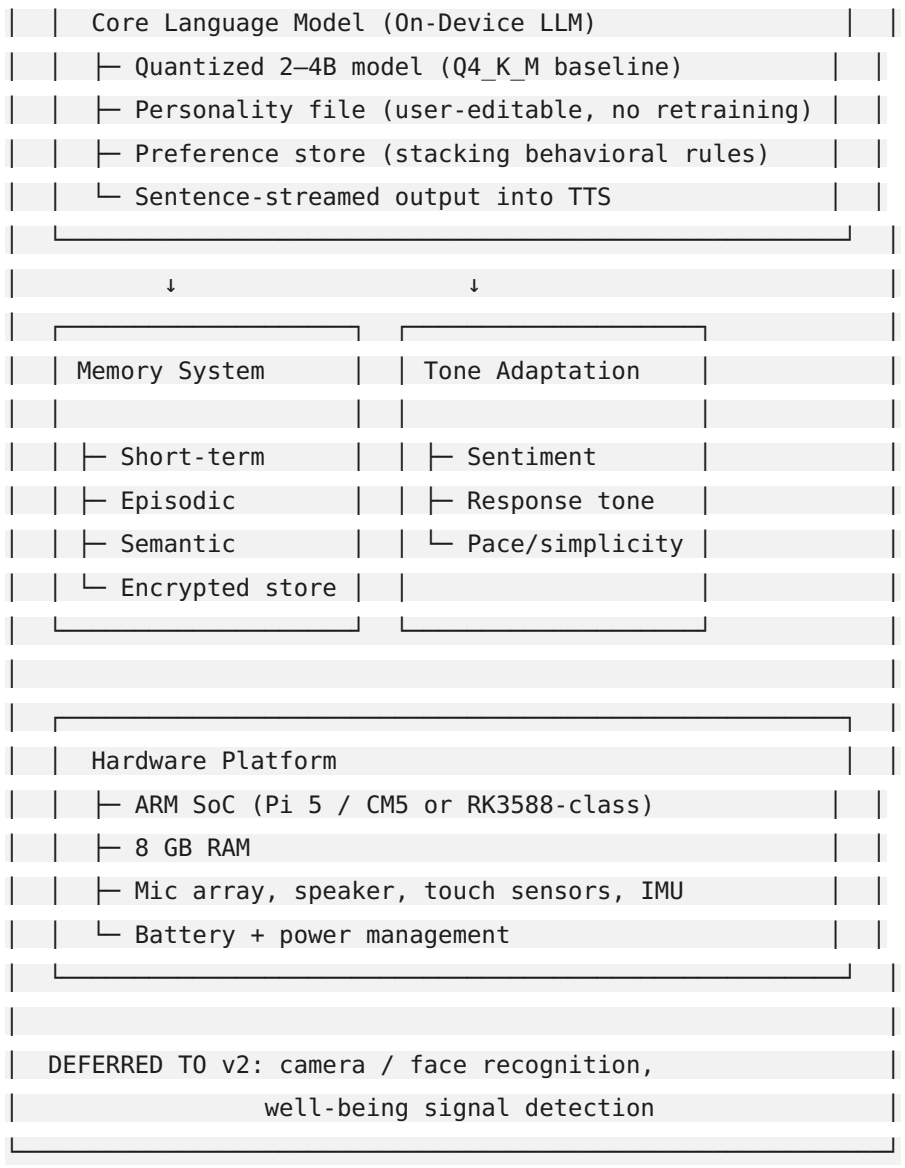
Non-technical readers: the Executive Summary, Design Principles, Staged Roadmap, and Glossary. Technical readers: Technical Architecture and the Latency Architecture section, which is where most of the engineering risk now lives.

System Design Principles

1. **100% offline in normal operation.** No cloud dependency for any conversational function.
2. **On-device processing only.** No raw audio, text, or embeddings leave the device.
3. **Low-power embedded hardware.** <10W sustained.
4. **Privacy-by-design**, including the negative space: features are declined when their privacy cost exceeds their value (see: no camera).
5. **Fail-safe simplicity.** Graceful degradation over complex failure modes.
6. **Unhurried by design.** The interaction pace is a feature for this audience, not a constraint to engineer away. This principle is what makes the real latency envelope acceptable.

System Overview





Latency Architecture

This section is new in v2.0 and is the most important engineering content in the document. v1.1's latency target was not achievable as written, and the fix is architectural rather than a matter of faster hardware.

The measured reality

Configuration	Throughput	Source
Apple Silicon, GPU-offloaded	86.98 tok/s	Milestone 9, measured

Configuration	Throughput	Source
Apple Silicon, CPU-only (`--pi-preview`)	1.81 tok/s	Milestone 9, measured
Pi 5, 3–4B Q4, quad A76	2–5 tok/s	Published llama.cpp benchmarks
Pi 5, 1–1.5B Q4	5–15 tok/s	Published llama.cpp benchmarks

A natural companion reply is roughly 30–50 tokens. At 3 tok/s that is **10–17 seconds** of generation alone, before ASR and TTS. v1.1's "<2–3 seconds per response" was never reachable on this class of hardware with this class of model.

Why the target was wrong, not the hardware

The error was measuring the wrong thing. In a spoken interface the user does not experience "total response time" — they experience **silence**. Silence before the first word is the entire perceptual cost; once speech begins, the remaining generation happens underneath it, invisibly, as long as generation outruns speech.

Speech runs at roughly **2.5–3 words per second**, which is about **3.5–4 tokens per second** of text consumed. So:

> If the model sustains ~4 tok/s or better, it stays ahead of the voice indefinitely. Total reply length becomes irrelevant to perceived latency.

This turns an apparently fatal 10–17 second problem into a tractable 2-second one, and it changes what the hardware has to be good at: not peak throughput, but **time-to-first-sentence and sustained ≥4 tok/s**.

The revised latency budget

Stage	Budget	Notes
Wake word / touch trigger → capture	<200 ms	Local, cheap
End-of-utterance detection	<500 ms	VAD, tunable; longer for this audience
ASR (streaming)	<1.0 s after speech ends	Whisper-small class, streaming
Memory retrieval (embed + top-k)	<300 ms	Measured cheap in prototype
LLM → **first complete sentence**	<2.0 s	~10–14 tokens at ≥6 tok/s
TTS first audio chunk	<400 ms	Streaming synthesis
Time-to-first-audio (the metric)	**<3.5 s**	Sum of the above

Stage	Budget	Notes
Sustained generation	**≥4 tok/s**	Must outrun speech

Three techniques that make this work

1. **Sentence-level streaming into TTS.** Generation and synthesis are pipelined: the moment the model emits a sentence boundary, that sentence goes to TTS while generation continues. This is the single highest-leverage implementation decision in the product.
2. **Pre-rendered acknowledgment tokens.** A small cache of natural short utterances ("Mm.", "Let me think.", a soft hum) plays immediately on trigger, from disk, with zero inference. This covers the ASR and prompt-assembly window and reads as *listening* rather than as *lag*. Used sparingly, it is the difference between "slow" and "thoughtful."
3. **Model sizing driven by time-to-first-sentence, not capability benchmarks.** If a 3–4B model cannot hold ≥4 tok/s on the chosen SoC, the correct response is a smaller model, not a longer latency budget. A 1–1.5B model at 5–15 tok/s comfortably clears the bar. For a companion whose job is listening and remembering — not answering trivia — this trade is very likely correct, and Stage C must test it explicitly rather than assume.

Consequence for hardware selection

Because the binding constraint is sustained tok/s, and LLM inference at this scale is **memory-bandwidth-bound rather than compute-bound**, memory bandwidth is the primary SoC selection criterion:

- **Raspberry Pi 5 / CM5** — quad Cortex-A76, substantially higher bandwidth than CM4. The baseline target.
- **RK3588-class** (Orange Pi 5, Radxa Rock 5) — eight cores plus a usable NPU, higher bandwidth again. The upside option.
- **Removed from consideration:** Jetson Nano (4 GB, Maxwell-era, effectively EOL) and CM4 (Cortex-A72, materially lower bandwidth).

Technical Architecture

1. Hardware Platform

Compute: 8 GB RAM minimum (not 4 GB — a Q4 model plus KV cache plus ASR plus TTS plus OS does not fit comfortably in 4 GB). NPU acceleration is desirable but must not be *required*, since llama.cpp's NPU support across these boards is uneven; CPU inference is the guaranteed path.

Sensors:

Component	Purpose	Notes
Microphone array (2–4 ch)	Far-field capture	Beamforming; physically isolated from speaker
Speaker	Voice output	Acoustically decoupled from mics — see AEC below
Touch sensors	Hug/press interaction; also a **wake trigger**	Redundancy for wake word failure
IMU	Motion/handling awareness	Cheap, useful for presence
~~Camera~~	~~Face recognition~~	**Removed from v1**

Power: the v1.1 figures did not reconcile. Restated:

- Realistic duty cycle: 15–20 conversational exchanges/day, ~2–3 hours of active listening, remainder in low-power wake-word standby.
- Standby draw dominates total consumption and is the number that matters. Wake-word detection must run on a low-power path.
- Target: **48 hours between charges at realistic use**, docking base for overnight charging.
- "8 hours of continuous conversation" is removed as a spec; it does not describe any real usage pattern and inflated the battery requirement.

2. Acoustic Echo Cancellation — a named program risk

A speaker and a microphone array inside the same plush body, inches apart, is close to the worst-case AEC scenario. This project has already hit an echo-cancellation wall once: on macOS, `setVoiceProcessingEnabled` failed because the OS attributed the audio session to the terminal rather than the application, and the microphone captured nothing (documented in issue #11). That specific blocker is Apple-specific and disappears on embedded Linux, where the audio stack is fully under our control — but the underlying acoustic problem does not.

Because always-listening barge-in (the user interrupts and the companion stops talking immediately) is central to the interaction model, AEC is promoted to a **Stage C gate with an explicit pass/fail test**, not a bullet in a hardware workstream:

> **Gate test:** with the companion speaking at normal volume, a user speaking at conversational level from 2 m away is transcribed correctly, and playback stops within 300 ms, in ≥9 of 10 trials, in a room with typical domestic background noise.

Mitigations, in order of preference: physical isolation and baffling between speaker and mic array within the plush form; software AEC (WebRTC APM or Speex); and, as a guaranteed fallback, **touch-to-talk** — squeezing the toy interrupts and starts listening. The fallback is not a consolation prize: a keyboard-driven equivalent already shipped in the prototype and tested well, and squeezing a stuffed animal to talk to it is arguably a *better* interaction for this audience than a wake word.

3. Core Language Model

Size: 2–4B parameters, Q4_K_M baseline, with an explicit obligation to drop to a smaller model if the ≥4 tok/s sustained target is not met. Q3_K_M and Q2_K are available levers if RAM rather than speed binds, with manual output-quality validation required after any such change.

Behavior shaping — no fine-tuning in v1. The prototype demonstrates that a loadable personality description plus a stacking preference store produces genuinely different and appropriate conversational register. This approach has three advantages over fine-tuning that matter more than the marginal quality difference:

- **User-editable at runtime.** The owner or family can adjust tone by speaking to it, with no retraining cycle.
- **Debuggable.** When behavior is wrong, the cause is readable text.
- **Auditable.** A caregiver can inspect exactly what the companion has been told to do.

Fine-tuning is reclassified as a **post-pilot optimization**, justified only by specific documented failures that prompting could not fix.

License: Gemma 4 is Apache 2.0 — commercial use, modification, and redistribution inside a shipped product are permitted without royalty or approval. This is materially better than the custom Gemma Terms of Use governing earlier generations, and it matters for a hardware product that ships the model to customers. (Confirm against current terms before shipping; this is not legal advice.)

4. Memory System

Substantially validated in the prototype. Retained from v1.1 with corrections from implementation experience.

Memory Type	Purpose	Storage
Short-term	Current conversation context	In-memory, per session
Episodic	Life events, timestamped	Encrypted SQLite
Semantic	People, preferences, routines	Encrypted SQLite + vectors

Implementation findings worth carrying forward:

- **Retrieval must blend similarity with recency.** Pure embedding similarity silently dropped recently-stated facts: a short question ("what's my name?") does not always score close enough to the long declarative sentence that answers it. The fix — always fold in the N most recent stored facts regardless of score — was a real bug fix, not a refinement. Expect the same class of failure to recur at scale.
- **Not everything should be stored.** Storing questions and small talk crowded genuine facts out of the retrieval window. A memory-worthiness filter is required, not optional.
- **Memory is decoupled from KV-cache continuity.** Each turn is a fresh decode of personality + retrieved context + message. This is why memory survives restarts for free, and why the design ports to embedded hardware. It should not be "optimized" into long-lived context.
- **Encryption at rest (AES-256)** from Stage C onward, integrated early rather than retrofitted — v1.1 was right about this.

5. Tone Adaptation (formerly "Emotional State Engine")

The v1.1 module bundled two very different things. v2.0 separates them.

In scope for v1 — conversational tone adaptation. Sentiment of the current exchange informs response register: simpler language when the user seems confused, a calmer pace when they seem anxious, warmth when they seem low. This is low-risk, immediately valuable, and largely achievable through the existing personality/preference mechanism.

Deferred — well-being signal detection. Tracking longitudinal mood, flagging "sudden withdrawal" or "repeated sadness," and surfacing that to family is a different product with different obligations. Three reasons it does not belong in v1:

1. **Reliance.** Once a family believes the companion is watching for decline, they rely on it. A false negative is then not a missed feature but a failure of care.
2. **Consent.** The person being monitored may not be able to meaningfully consent, particularly in a dementia context. Consent from the family is not consent from the user.
3. **Regulatory posture.** Health-signal inference invites a classification this product should not casually accept.

This is already tracked internally as high-stakes and needing scoping (issue #9). It should be designed deliberately, with clinical input, as a separate opt-in product decision — not inherited from an architecture diagram.

6. Identity Recognition — camera removed from v1

What v1.1 proposed: a camera module with on-device face embedding, a registry of 50–100 family members, and a physical privacy switch.

Why v2.0 removes it:

- The stated benefit does not hold up. "Recognize when Sarah calls → load Sarah's memories" describes a phone call, which is audio; face recognition requires line of sight and would not fire.
- The privacy cost is the highest in the design. A camera inside a plush toy in an older adult's home performing face recognition is a large trust ask even when provably local. **"It has no camera"** is a strictly stronger claim than any switch, and it is the kind of claim that sells a privacy-first product.
- It carries real cost in BOM, power, thermals, and biometric-data regulatory exposure — for a feature that is not load-bearing to the emotional core.

What replaces it: optional **voice-based speaker identification**, which delivers the actual desired capability (knowing who is talking, and loading their context) at a fraction of the cost, using a microphone the device already has. Face recognition returns as a candidate for a future version only if pilot users specifically ask for it.

Staged Roadmap

Reconciling two numbering schemes

The implementation guide (`Docs/Gemma_4_E2B_Persistent_Memory_Guide.md`) uses "Phase 1" for the Mac software build and "Phase 2" for the iOS app. v1.1 used "Phase 0–4" for product development. These collide. v2.0 uses **lettered stages** and maps them explicitly:

v2.0 Stage	Guide equivalent	v1.1 equivalent
A — Software core (Mac)	Phase 1, Milestones 1–9	Phase 0 Workstream A + B
B — Handheld field prototype (iOS)	Phase 2	*(no equivalent — new)*
C — Embedded port (Linux/ARM)	*(separate project)*	Phase 0 Workstream C + Phase 1
D — Form factor & pilot	—	Phase 3
E — Refinement & launch	—	Phase 4

STAGE A — Software Core COMPLETE

Goal: prove the conversational and memory architecture works at all.

Nine ordered milestones, each independently verifiable, all passing: package builds → model loads → streaming REPL → memory persists across processes → semantic recall discriminates correctly → personality changes tone → story import and recall → voice output → efficiency baseline.

Delivered: a memory-backed, personality-driven, voice-enabled assistant running offline on Apple hardware, plus the measured throughput figures this document's latency architecture is built on.

Also delivered, and underrated: a validated *development method*. Every milestone surfaced a real defect — a KV-cache position collision on the second message, a TTS voice identifier that resolved to a valid object with zero bytes of audio behind it, a synthesizer stop call that cancelled synthesis but not already-queued playback, and a few-shot classification prompt whose own examples were teaching the model the wrong answer. None of these would have been caught by code review. The milestone-with-explicit-pass/fail discipline is a program asset and should govern Stages B–E.

STAGE B — Handheld Field Prototype (iOS) · 8–10 weeks

Goal: get Companion into the hands of real older adults before spending anything on hardware. This stage did not exist in v1.1 and is the largest single risk reduction in this revision.

An iPhone or iPad already has an excellent microphone array, speaker, on-device ASR, and high-quality TTS. The software core already runs on it. That means the *product* hypothesis — does an offline companion that remembers actually help someone? — can be tested months before a plush prototype exists, at a cost of essentially zero hardware.

Workstreams:

- **B1 — App shell.** SwiftUI wrapper over the existing package; settings for personality, voice, and memory review. Increased-memory-limit entitlement for the model. (Guide Section 10.)
- **B2 — Voice input and barge-in.** The interaction model's first real test. iOS has none of the macOS audio-session attribution problems that blocked this on the Mac.
- **B3 — Caregiver memory review.** A family-facing view of what the companion has remembered. Tests the transparency story early.
- **B4 — Field study, 8–12 participants, 4 weeks.** Older adults with family support, daily use, structured interviews.

What Stage B is designed to find out — the questions that actually decide the product:

- Does the memory feel continuous, or uncanny?
- Is an unhurried pace experienced as calm, or as broken?
- Does it demonstrably increase family contact, per the core principle?
- What do people actually talk to it about? (This should drive everything downstream, and cannot be guessed.)
- Is voice interaction reliable enough for users with age-related speech and hearing changes?

Gate: proceed to hardware only if the field study supports the emotional-value hypothesis. **If it does not, the program stops here having spent no hardware money** — which is the entire point of adding this stage.

STAGE C — Embedded Port · 12–16 weeks

Goal: prove the experience survives $60 hardware.

The Swift package does **not** port. This is a separate codebase — realistically Python via `llama-cpp-python`, or C++ — reusing the model, quantization, prompt design, and retrieval architecture, none of which are Apple-specific. Budget for a rewrite of the application layer, not a port.

Workstreams (parallel):

- **C1 — Inference on target.** Benchmark Pi 5 / CM5 and RK3588 at Q4_K_M, Q3_K_M, and multiple model sizes. **Select the largest model that sustains ≥4 tok/s**, explicitly set thread count to core count (llama.cpp under-threads by default on Pi 5, roughly halving throughput if left unset).
- **C2 — Streaming voice pipeline.** ASR → LLM → sentence-streamed TTS, measured against the time-to-first-audio budget. This is where the latency architecture is proven or disproven.
- **C3 — Acoustic gate.** AEC and barge-in against the pass/fail test above, with touch-to-talk as the guaranteed fallback.
- **C4 — Security.** Encrypted storage, local-only logging, verified no-network operation, data export and secure wipe.

Gate: time-to-first-audio <3.5 s, sustained ≥4 tok/s, and the acoustic gate passed on target hardware.

STAGE D — Form Factor & Pilot · 16–20 weeks

Goal: a plush device real people live with.

- **D1 — Industrial design.** Electronics into plush: mic array placement, speaker acoustic isolation, thermals inside an insulating material (a stuffed animal is, by construction, a thermal blanket — this is a genuine and under-appreciated constraint), battery and charging, washability, durability.
- **D2 — Safety and compliance.** Lithium battery safety, any applicable radio certification, materials safety, and product-liability review for a device used by a vulnerable population. **This is a long-lead item and must start at the beginning of Stage D, not the end.**
- **D3 — Extended pilot.** 10–15 users, 8 weeks (not one week — memory value only becomes visible over time, and a one-week trial structurally cannot evaluate the product's central claim).
- **D4 — Independent security audit.**

STAGE E — Refinement & Launch · 12+ weeks

Iterate on pilot findings; documentation for users, families, and caregivers; manufacturing readiness; staged rollout.

Reconsider here, with evidence: fine-tuning, well-being signals, speaker/face identification, family sync.

Timeline Summary

Stage	Duration	Cumulative	Risk
A — Software core	Complete	—	Retired
B — Handheld field prototype	8–10 weeks	~2.5 months	Low (no hardware)
C — Embedded port	12–16 weeks	~6.5 months	**High** (latency, acoustics)
D — Form factor & pilot	16–20 weeks	~11.5 months	**High** (physical, regulatory)
E — Refinement & launch	12+ weeks	~15 months+	Medium

Realistic total: 18–24 months to production, against v1.1's 8 months.

The increase is not pessimism. v1.1's estimate excluded certification, plush manufacturing tooling, and legal review; assumed a 6–8 week fine-tuning effort now deferred; and compressed pilot testing to one week, which

cannot evaluate a memory product. v2.0 also *adds* Stage B — and that addition is what makes the longer schedule lower-risk overall, because the expensive stages are no longer entered on an unvalidated hypothesis.

Resource Requirements

Team

Role	FTE	Stages
ML / Systems Engineer	1.0	B, C, D, E
Embedded Engineer	0.75	C, D
Hardware / Industrial Designer	0.5	D
Security Engineer	0.25	C, D
Product Manager / Researcher	0.5	B, D, E
Total	**3.0 FTE**	(staged, not concurrent)

Lower than v1.1's 3.75 FTE because fine-tuning is deferred and the software core is complete.

Budget (non-salary)

Item	Cost	Notes
Stage B field study	$2K–$4K	Devices, participant compensation
Hardware prototyping	$8K–$15K	Multiple SoCs, mic arrays, iterations
Plush tooling & samples	$10K–$25K	**Absent from v1.1**
Certification & compliance	$8K–$20K	Battery safety, radio, materials — **absent from v1.1**
Security audit	$5K–$10K	Third-party
Legal review	$5K–$15K	Eldercare-adjacent product — **absent from v1.1**
Contingency (25%)	$10K–$22K	
Total	**$48K–$111K**	vs. v1.1's $12K–$27K

The increase is almost entirely categories v1.1 omitted rather than underestimated. Fine-tuning compute ($2–5K in v1.1) is removed.

Success Metrics

Technical

Metric	Target	Change from v1.1
Time-to-first-audio	<3.5 s	**Replaces** "response latency <2–3 s"
Sustained generation	≥4 tok/s	New — must outrun speech
Barge-in stop latency	<300 ms	New — acoustic gate
Memory retrieval accuracy	>95% on known facts	Unchanged
Battery between charges	>48 h realistic duty cycle	Replaces "8 h continuous"
Thermal	<60 °C surface, sustained, inside plush	Now specified *inside* the form factor
Uptime	No crashes over an 8-week pilot	Extended from 1 week

User (Stages B and D)

Metric	Target
Perceived companionship	≥7/10
Engagement	>3 interactions/day sustained past week 4
Memory accuracy, as perceived	"remembers" >80% of the time
Family contact change	**Measurable increase** — the core principle, measured
Privacy confidence	Users and families report feeling protected
Retention past 8 weeks	>70% still in daily use

Two additions matter most: engagement **sustained past week 4** (novelty decays; a one-week pilot measures novelty, not value), and family contact measured directly rather than assumed.

Risk Register

Risk	Prob.	Impact	Mitigation
Latency unacceptable on target SoC	Medium	High	Sentence-streamed TTS; model sizing driven by tok/s; Stage C gate before form factor
AEC fails inside plush form	**High**	High	Physical isolation; software AEC; **touch-to-talk fallback already proven**
Thermals inside insulating plush	Medium	High	Explicit Stage D modeling; SoC power headroom reserved
Product hypothesis is wrong	Medium	**Critical**	**Stage B tests it before hardware spend**
Novelty decay after weeks 2–4	Medium	High	8-week pilot; retention metric
Certification delays	Medium	High	Compliance starts at Stage D open, not close
Model quality drops at smaller size	Medium	Medium	Milestone 6/7-style manual validation after any quantization or size change
Memory retrieval degrades at scale	Medium	Medium	Recency+similarity blend already implemented; needs multi-year simulation
Family over-relies on well-being signals	—	**Critical**	**Feature deferred** rather than mitigated

Future Expansion

Reconsidered after Stage E, with pilot evidence:

- **Well-being signals + caregiver dashboard** — with clinical input and explicit, separately-obtained consent.
- **Voice-based speaker identification** — likely; low cost, real benefit.
- **Camera / face recognition** — only if pilot users ask for it.
- **Optional secure family sync** — manual encrypted export, never automatic.
- **Memory export to a printed memory book** — a genuinely appealing expression of the core promise, and possibly the most emotionally resonant feature in this list.
- **Multi-device household mesh**, offline.
- **Condition-specific companionship models** (grief, dementia care) — requires clinical partnership.

Conclusion

Companion is a long-term memory-preserving intelligence system designed to live beside a person for years. v1.1 established that vision and a sensible development structure. v2.0's contribution is to replace its estimates with measurements, and to reorder the program so the cheapest experiments answer the most important questions first.

Three changes carry most of the value:

1. **The latency target was wrong, and the fix is architectural.** By streaming sentence-by-sentence into speech, an apparently fatal 10–17 second generation time becomes a sub-2-second perceived response. The hardware requirement shifts from peak throughput to sustained throughput that outruns the voice.
2. **Real users can be reached before hardware exists.** Stage B puts a working companion in older adults' hands using devices they may already own, months earlier and at negligible cost. If the emotional hypothesis is wrong, that is the moment to learn it.
3. **Two features were removed rather than engineered.** The camera and longitudinal well-being detection each carried privacy, trust, or safety costs out of proportion to their contribution. Declining them makes the privacy claim simpler and stronger — for a privacy-first product, "it has no camera" is a feature.

The software core is built and validated. What remains is the harder question, and Stage B is designed to answer it early: does an offline companion that genuinely remembers make an older person's life better? Everything downstream should be contingent on that answer.

Glossary

Technical

- **AEC (Acoustic Echo Cancellation):** Removing the device's own speaker output from what its microphones hear, so it doesn't respond to itself.
- **ASR (Automatic Speech Recognition):** Converting speech to text.
- **Barge-in:** The user interrupting mid-reply and the device stopping immediately.
- **Embedding:** A numerical vector representing text, used to find related memories by meaning rather than keyword.
- **Fine-tuning:** Retraining a model on specific data.
- **GGUF:** The file format llama.cpp uses for quantized models.
- **KV Cache:** Stored intermediate computation that speeds up generation.

- **llama.cpp:** The inference engine used here, chosen because the same engine and model file run on macOS, iOS, and ARM Linux.
- **Memory bandwidth:** How fast a processor moves data to and from RAM. The primary limit on LLM speed at this scale — more important than core count.
- **NPU:** Hardware specialized for AI computation.
- **Quantization (Q4_K_M etc.):** Reducing numeric precision to shrink and speed up a model, at some quality cost. Lower numbers are smaller and faster but less capable.
- **SoC (System-on-Chip):** A complete computer on one chip.
- **Time-to-first-audio:** How long from the user finishing speaking until the device starts speaking. **The latency metric that matters.**
- **tok/s (tokens per second):** Generation speed. Roughly 3.5–4 tok/s is the rate ordinary speech consumes text, so that is the floor for uninterrupted speech.
- **TTS (Text-to-Speech):** Converting text to spoken audio.
- **VAD (Voice Activity Detection):** Detecting when someone has stopped speaking.

Product

- **Duty cycle:** The realistic proportion of time a device is actively working, used for honest battery estimation.
- **Edge computing / on-device:** Processing locally rather than in the cloud.
- **Offline:** Operating with no internet connection.
- **Privacy-by-design:** Building privacy in from the start — including declining features whose privacy cost is too high.

Document Version History: - v1.0: Original white paper - v1.1: Revised development pipeline, readability improvements, glossary, architecture diagram - v2.0: Evidence-based revision. Latency architecture rebuilt around time-to-first-audio; hardware shortlist updated; fine-tuning deferred; camera and well-being detection removed from v1; Stage B (handheld field prototype) added; timeline and budget corrected; stage numbering reconciled with the implementation guide.