

Eptesicus Labs

Long-Term Business Plan

Focus: making Small Language Models the default choice for real products where cost, privacy, and control matter.

Prepared by: Deyan Todorov

Date: 2025-12-15

Confidential - for discussion purposes only

Executive summary

Eptesicus Labs builds small language models (SLMs) and a reliability layer that steers answers toward correctness without relying on cloud inference. The goal is to make local-first AI practical for day-to-day consumers and viable for enterprise deployment, with predictable costs and privacy by default.

Our initial commercial stack is Lumis-1: a small, fine-tuned model paired with lightweight validator models and an orchestrator that performs up to three repair attempts before returning an answer. If the system cannot verify a claim, it returns a low-confidence answer with clear uncertainty rather than fabricating specifics.

Core thesis

- Cloud-first AI pushes products into recurring subscription costs and per-token pricing that becomes unpredictable at scale.
- Local models reduce ongoing costs and data exposure, but today they are hard to trust due to inconsistent behavior and uncorrected errors.
- A small-model stack needs reliability engineering (validators, steering, evaluation), not only better base weights.

Business approach

- Consumers: free personal-use product to drive adoption and feedback.
- Enterprises: commercial licensing of Lumis-1 (model + validator-guided steering) with support/SLA tiers.
- Partnerships: OEM/ISV licensing for bundling into devices and vertical software products.

Vision and principles

Eptesicus Labs is built around an engineering-first premise: small models should be dependable enough to be the default choice, not a compromise. We do not treat reliability as a marketing claim. We treat it as something you measure, stress-test, and improve release by release.

Principles

- Local-first: reduce dependence on remote APIs where possible; keep user data under user control.
- Predictable costs: avoid business models that require perpetual per-token spending for basic functionality.
- Honest outputs: when verification is limited, label low confidence and uncertainty.
- Reproducible evaluation: ship with benchmarks, traces, and acceptance tests that show what improved and what did not.

Market problem

Teams and consumers want AI features, but cloud inference often implies recurring subscription costs and vendor lock-in. At the same time, sending private prompts and documents to third-party servers introduces privacy and compliance risk. Local models can avoid both, but the experience is frequently unreliable: answers drift, errors slip through, and it is unclear when the model should be trusted.

What customers ask for (in plain language)

- I do not want another monthly AI subscription just to use basic features.
- I do not want private data leaving the device unless I explicitly choose to.
- If the model is unsure, I want it to say so and still be helpful.

Product strategy

Lumis-1 is both the model and the reliability system around it. It combines three parts:

- Speaker model: the base SLM, fine-tuned on high-quality instruction and domain data.
- Validator Council: small discriminative models that score safety, consistency, and likely factual support.
- Orchestrator: a steering loop that proposes a correction and retries generation up to three times before responding.

Default behavior is steering, not silence. The system can refuse unsafe requests. For all other cases it returns an answer, and when verification is limited it attaches low-confidence language instead of inventing details.

Long-term product line

- Lumis-1 (near-term): fine-tuned SLM + validators + steering, delivered as a complete stack.
- Lumis-2 (mid-term): improved base model trained on proprietary reliability-focused datasets, with portable validators.
- Lumis-3 (long-term): a fully trained SLM family designed from the ground up for reliability under tight compute budgets.

Technology roadmap (24-36 months)

Phase 1 - Reliability foundation

- Build a reproducible evaluation harness with stress tests (contradictions, unknowns, refusal classes).
- Train initial validators and integrate steering with a maximum of three retries.
- Define acceptance criteria: lower unsupported claims, improved self-correction, consistent low-confidence behavior.

Phase 2 - Data and training scale

- Expand proprietary datasets focused on failure cases: contradictions, near-misses, and low-confidence prompts.
- Iterate on validator calibration to avoid over-refusal and preserve usefulness.
- Introduce model-version portability tests (how validators behave across speaker variants).

Phase 3 - First native model family

- Train a small model family optimized for local use and reliability (not only instruction-following).
- Tighten the repair loop: better corrective prompting, better consistency scoring, better uncertainty labeling.
- Prepare enterprise-grade packaging and update cadence.

Data and evaluation strategy

Reliability improvements must be proven with measurement. Eptesicus Labs invests in datasets that represent failures, not only successes. The focus is on: (1) where the model should refuse, (2) where it should answer but label uncertainty, and (3) where it can be steered into a correct answer.

Key dataset buckets

- Synthetic negatives: controlled contradictions and unanswerable prompts to teach detection and calibration.
- Refusal classes: unsafe or disallowed requests, with clear boundaries and consistent policy outcomes.
- Repair pairs: prompts where an initial draft is wrong but a corrected reasoning path exists.
- Calibration sets: prompts designed to tune thresholds so the system remains useful and does not become overly conservative.

Key metrics

- Unsupported claim rate (how often the system asserts specifics without support).
- Self-correction rate (how often steering converts a wrong draft into a correct final).
- Low-confidence precision (how often uncertainty labeling is appropriate).
- User usefulness proxy (task completion without unacceptable fabrication).

Business model

The company is built for predictable, enterprise-grade revenue without monetizing consumers.

Revenue streams

- Enterprise licensing: annual commercial licenses for Lumis-1 (model + validator steering) with support/SLA tiers.
- OEM/ISV partnerships: licensing for bundling into devices and vertical software products.
- Services (selective): onboarding, integration support, and long-term maintenance contracts for regulated deployments.

Value retention comes from proprietary reliability datasets, continuous evaluation infrastructure, and release-by-release improvements that are measurable and auditable.

Distribution and IP strategy

Consumer distribution is designed to maximize reach while keeping enterprise usage commercial. Enterprise distribution is governed by commercial terms and support commitments. Because models run on customer-controlled hardware, technical controls can deter casual extraction but cannot guarantee prevention. Protection is therefore a combination of licensing, attribution, and operational gating.

Approach

- Consumers: free personal-use access to the product.
- Enterprises: commercial rights, governance features, and contractual support.
- IP protection: clear license terms, watermarking/fingerprinting where appropriate, and the ability to revoke access for violations.

Go-to-market plan

Initial traction comes from consumers and technical communities that already want local-first AI. Enterprise adoption follows once reliability and update cadence are proven with public, reproducible evaluation.

Acquisition

- Public demos and benchmarks that show measurable improvements (not marketing claims).
- Creator and developer channels: GitHub, forums, and technical communities.
- Early enterprise pilots with privacy-sensitive teams that want to reduce API spend.

Retention

- Frequent releases with clear changelogs and measurable deltas.
- Fast feedback loops: error reports translate into new failure datasets and validator updates.
- Enterprise support: predictable update schedule and stable versions.

Team and operations plan

The company is founder-led at inception. Growth is staged around the minimum roles needed to ship and support a commercial stack.

Planned hires (when funding allows)

- ML evaluation and data lead (datasets, calibration, and acceptance testing).
- Product engineer (integration, packaging, telemetry for offline-safe debugging).
- Partnerships lead (OEM/ISV pipeline, enterprise pilots).

Key risks and mitigations

Technical risks

- Over-conservatism: validators may suppress useful answers. Mitigation: calibration sets and usefulness metrics.
- Generalization: validators may overfit synthetic patterns. Mitigation: mixed data sources and targeted human curation for validation.
- Latency/compute budget: steering retries increase cost. Mitigation: cap retries at three, optimize validator inference, and tune retry triggers.

Business risks

- Competitive pressure: larger players may improve local models quickly. Mitigation: differentiate on measurable reliability and portability of the reliability layer.
- IP leakage: models can be extracted. Mitigation: licensing, watermarking, gating, and making enterprise value include support/governance beyond weights.
- Enterprise adoption cycles: long procurement. Mitigation: start with pilots and partnerships while building consumer traction.

Milestones and success criteria

Success is defined by shipping improvements that are measurable and repeatable, not by claims about being the best model.

Near-term (6-12 months)

- Lumis-1 stack integrated and testable end-to-end with steering and trace logs.
- Public evaluation report showing reduced unsupported claims and improved self-correction vs baseline.
- First enterprise pilot with a privacy- or cost-motivated buyer.

Mid-term (12-24 months)

- Expanded proprietary reliability datasets and improved calibration across versions.
- OEM/ISV partnership discussions with at least one signed pilot.
- Stable release cadence and support process for commercial customers.

Long-term (24-36 months)

- First native small-model family trained with reliability as a core objective.
- Commercial product maturity: predictable updates, reproducible evaluation, and growing deployment base.
- Clear pathway to expand into additional local-first product categories beyond the initial assistant use case.

Appendix: glossary (short)

- SLM (Small Language Model): a model designed to run under constrained compute and memory budgets.
- Validator: a discriminative model that scores a draft on specific properties (safety, consistency, likely support).
- Steering: a controlled loop that uses validator feedback to correct and retry generation.
- Low-confidence answer: a response that remains useful while explicitly labeling uncertainty when verification is limited.