LSAS: A Practical, Auditable Safety-Orchestration Architecture for Generative AI Outputs in Regulated Healthcare and MedTech

Author: Matt Vegas, Principal AI Consultant @ Inference-Stack.com

Version: v1.2 (Enterprise and Academic Edition)

Date: January 23, 2026

Origin note: The term "LSAS" (Layered Safety and Accuracy System) was coined by Matt Vegas during Soluna AI Platform systems design work (April 2025).

Abstract

Generative AI is increasingly embedded into healthcare and medtech workflows, including requirements discovery, clinical workflow design, documentation, and live code generation inside developer tools. Yet raw model outputs are unsafe by default in regulated environments due to predictable failure modes: unsupported assertions, overconfident compliance language, privacy leakage, insecure code generation, prompt injection, insecure output handling, and drift in standards and regulation.

This paper introduces LSAS (Layered Safety and Accuracy System): a practical safety-orchestration architecture that treats all AI outputs (text and code) as untrusted intermediate artifacts until they are grounded, validated, and governed through deterministic controls with auditable evidence. LSAS composes five layers: (1) intent and risk classification; (2) authoritative knowledge grounding with optional live retrieval; (3) real-time validation and filtering for accuracy, compliance posture, security, and accessibility; (4) escalation protocols for uncertainty or high-risk domains; and (5) continuous learning through policy versioning, feedback loops, regression harnesses, and a regulatory watcher.

LSAS is designed to align with HIPAA privacy and security realities, NIST implementation guidance for HIPAA safeguards, and healthcare accessibility requirements, while remaining extensible through policy packs for adjacent regimes such as PCI DSS, SOC 2, HITRUST CSF, and medical device lifecycle governance. This document is written as a build guide: it includes reference architectures, validator catalogs, operational checklists, and concrete implementation instructions intended for enterprise adoption and academic scrutiny.

Keywords

Healthcare AI, MedTech AI, HIPAA, SaMD, CDS, AI governance, validation pipeline, grounded generation, output governance, LLM security, auditability, accessibility

Executive Summary

Healthcare and medtech organizations do not fail with AI because models are incapable; they fail because outputs are treated as authoritative without governance. LSAS is an engineering answer: it turns "AI output" into a governed pipeline with evidence, validation, and escalation.

If you are a health system, payer, digital health company, research institution, or medical device manufacturer, LSAS is designed to satisfy expectations already applied to production systems:

- Safety is enforced by architecture, not training alone.
- High-risk claims are grounded to authoritative sources and are auditable.
- Outputs are validated across privacy, security, accessibility, and regulated design patterns.
- Controls evolve with standards and threats through versioned policy packs and regression testing.

This paper provides an implementable blueprint and a common vocabulary. It aims to reduce ambiguity and enable reproducible safety posture across teams, vendors, and institutions.

How to Use This Paper

- If you are deciding whether LSAS is relevant: read Sections 1 to 3 and the capability gating table.
- If you are implementing LSAS: read Sections 6 to 12 and Appendix B (policy pack example).
- If you are evaluating LSAS for enterprise or medtech compliance: read Sections 2, 4, 12, 14, 15, and Appendix D (threat mapping).

Table of Contents

1. Introduction

Healthcare product teams increasingly use generative AI for two categories of work:

- Information outputs: strategic guidance, requirements, product design analysis, clinical workflow reasoning, and compliance-sensitive documentation.
- Code outputs: live code generation inside an IDE (front-end components, API routes, infrastructure snippets, and supporting utilities).

Both output types must meet strict standards for privacy, security, accessibility, and regulatory posture. The dominant operational risk is not that an AI assistant "gets something wrong." It is that teams unknowingly ship incorrect guidance, insecure code, or unsafe workflows because model output was accepted without validation.

LSAS establishes a repeatable contract: no response or code snippet is treated as final until it is classified, grounded, validated, and logged with an audit artifact. This transforms AI assistance from a probabilistic tool into an enforceable system boundary.

2. Enterprise and Academic Review: Likely Objections and Hardening Decisions

This section probes LSAS through the lens of top healthcare organizations and medtech manufacturers. It enumerates common objections and the hardening decisions required for adoption at scale. The goal is implementation credibility.

2.1 "This is just guardrails or a rules engine"

Objection: Large organizations already have policies, prompt guidelines, and retrieval-augmented generation (RAG). They will dismiss LSAS if it appears to be a generic rule set.

Hardening decision: LSAS is positioned as a safety-orchestration system with:

- Separation of control plane (policy and governance) from data plane (runtime execution).
- Policy packs as versioned, testable artifacts ("policy as code").
- A deterministic validator pipeline with explicit evidence contracts.
- A standard audit artifact (the LSAS Safety Report) that makes behavior traceable and reproducible.

2.2 "Where does this live in our HIPAA program, audits, and incident response?"

Objection: HIPAA programs require safeguards and documentation. Systems must support audit readiness and breach investigations without increasing exposure.

Hardening decision: LSAS treats audit evidence as a first-class product output. It enforces minimum-necessary principles for prompts, logs, and audit artifacts, consistent with HIPAA privacy guidance and the minimum necessary standard [2][3]. LSAS can be aligned to HIPAA implementation guidance via NIST SP 800-66 Rev. 2 [7].

2.3 "What about cloud providers, BAAs, and vendor risk?"

Objection: Many AI deployments are blocked or delayed due to uncertainty about whether cloud services are business associates and whether "no-view" encrypted storage changes BAA obligations.

Hardening decision: LSAS explicitly models vendor boundaries and requires BAA posture when a cloud service provider creates, receives, maintains, or transmits ePHI, including "no-view" services where the provider lacks the decryption key [4].

2.4 "How do you prevent PHI leakage through prompts, logs, embeddings, and citations?"

Objection: The most common AI governance failure in healthcare is accidental PHI exposure in prompts, logs, observability tooling, or vector stores.

Hardening decision: LSAS includes explicit data boundary controls:

- Prompt minimization and deterministic redaction prior to model invocation.
- Prohibition (by default) on embedding PHI into vector stores without a documented justification, compensating controls, and access boundaries.
- Safety Reports designed to support investigation with hashes and metadata rather than raw PHI retention.
- Use of HIPAA de-identification methods and guidance for synthetic or properly de-identified data [5].

2.5 "How do we prove this works, and how do we keep it working after changes?"

Objection: Frameworks are not systems. Enterprises demand measurable outcomes and regression safety.

Hardening decision: LSAS includes an evaluation and change-control model:

- Golden prompt suites and adversarial prompt sets mapped to OWASP LLM risks [17][18].
- Metrics for evidence coverage, validator failure rates, false-positive cost, latency budgets, and defect reduction.
- Policy pack versioning and regression harnesses as release gates.

2.6 "How does this integrate with medtech quality systems and device lifecycle expectations?"

Objection: Medtech manufacturers operate within quality system expectations and device lifecycle processes. If LSAS is not compatible with design controls, risk management, cybersecurity guidance, and post-market monitoring, it will not be adopted for regulated workflows.

Hardening decision: LSAS is structured to integrate with medtech artifacts and workflows:

- Cybersecurity guidance for premarket submissions and quality system considerations [13].

- Quality Management System Regulation (QMSR) transition incorporating ISO 13485 with enforcement beginning February 2, 2026 [12].
- AI-enabled device change management concepts such as Predetermined Change Control Plans (PCCPs) [15].
- Clear boundaries for Clinical Decision Support (CDS) software guidance scope and what constitutes non-device CDS [14].

2.7 "Transparency expectations: how will we explain what the system did?"

Objection: Healthcare increasingly expects algorithmic transparency, including requirements for predictive decision support interventions in certified health IT contexts.

Hardening decision: LSAS produces explainable artifacts by design: it stores evidence ledgers, validator findings, and policy versions for each output. This aligns with transparency expectations reflected in ONC's HTI-1 final rule focus on algorithm transparency and decision support intervention documentation [16].

2.8 "Where is the governance model and accountability boundary?"

Objection: Enterprise and academic reviewers will ask who owns policy changes, who approves high-risk behaviors, and how accountability is enforced across teams and vendors.

Hardening decision: LSAS explicitly separates runtime enforcement (data plane) from policy ownership and approval (control plane). Section 15 defines recommended roles, approval paths, and change control gates. The Safety Report enables accountability by recording policy versions and validator results for each output.

3. Regulatory and Assurance Baseline (Healthcare-First, MedTech-Ready)

This paper is healthcare-first. It emphasizes U.S. regulatory baselines that frequently anchor enterprise requirements, while enabling extension to other regimes through policy packs.

3.1 HIPAA privacy, minimum necessary, and data handling

HIPAA privacy guidance emphasizes that protected health information should not be used or disclosed when not necessary, and that regulated entities should make reasonable efforts to limit PHI to the minimum necessary to accomplish intended purposes [3]. LSAS enforces minimum-necessary by design for prompts, outputs, logs, and audit artifacts.

3.2 HIPAA security, safeguards, and implementation guidance

The HIPAA Security Rule requires administrative, physical, and technical safeguards to protect ePHI [1]. NIST SP 800-66 Rev. 2 provides practical guidance and resources for regulated entities to implement HIPAA safeguards [7]. LSAS is designed to support these expectations through documented policies, deterministic validation, and evidence capture.

HHS has published technical safeguard guidance that highlights addressable encryption flexibility under the current rule while requiring reasonable and appropriate measures [6]. In addition, HHS has published a HIPAA Security Rule Notice of Proposed Rulemaking (NPRM) to strengthen cybersecurity expectations for ePHI, underscoring that requirements and enforcement posture may become more prescriptive over time [10][11].

3.3 HIPAA cloud computing guidance and business associate status

HHS guidance clarifies that a cloud service provider that maintains ePHI is a business associate even if the provider does not have access to the decryption key ("no-view services") [4]. This is a common enterprise gating item for AI

deployments, and LSAS explicitly models vendor boundaries and required agreements.

3.4 De-identification and synthetic data posture

HHS provides guidance regarding methods for de-identification of PHI under the HIPAA Privacy Rule, including Expert Determination and Safe Harbor approaches [5]. LSAS treats de-identification and synthetic data as first-line options for examples, test sets, and demonstrations.

3.5 NIST CSF, AI RMF, and the Generative AI Profile

NIST CSF 2.0 provides a cybersecurity risk management taxonomy used broadly across regulated sectors [8]. NIST AI RMF 1.0 provides a risk management framework for AI systems [9]. NIST AI 600-1 provides a generative AI profile that can be used as a companion resource for GenAI risk management [20]. LSAS can be mapped to these frameworks to support governance alignment and executive reporting.

3.6 FDA digital health and medtech expectations

For medtech contexts, FDA has published guidance on cybersecurity in medical devices addressing quality system considerations and premarket submission content [13]. FDA has also issued guidance related to Clinical Decision Support (CDS) software scope and non-device criteria [14]. For AI-enabled device software functions, FDA has provided recommendations for Predetermined Change Control Plans (PCCPs) to support iterative improvements while maintaining assurance of safety and effectiveness [15].

FDA's QMSR final rule incorporates ISO 13485 by reference and is enforceable beginning February 2, 2026, affecting how medtech manufacturers manage controlled documents, records, and design controls [12].

3.7 ONC HTI-1 transparency for decision support interventions

ONC's HTI-1 final rule advances algorithm transparency requirements for predictive algorithms included in certified health IT [16]. Even outside certification contexts, this rule influences how major healthcare organizations think about documentation, transparency, and explainability for decision support tools.

4. System Boundary and Data Lifecycle: Preventing "PHI by Accident"

Many GenAI programs fail because they treat the model call boundary as the only boundary. In healthcare, the real boundary is the full data lifecycle: prompts, retrieved context, logs, embeddings, analytics, and audit artifacts. LSAS makes this explicit.

4.1 Data artifacts governed by LSAS

- Input prompt and context
- Retrieved sources (regulatory documents, internal policies, knowledge bases)
- Model draft output (untrusted artifact)
- Validator findings and transformations (redactions, downgrades, remediations)
- Final output and citations
- Safety Report (audit artifact)
- Telemetry and metrics

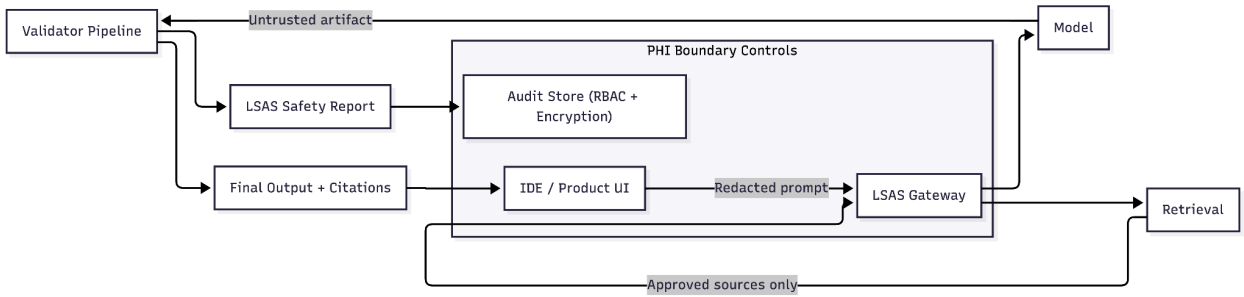
4.2 Data retention philosophy (default posture)

- Do not store raw PHI in Safety Reports by default.
- Prefer hashes, metadata, and evidence pointers.
- Store only what is needed for investigation, reproducibility, and audit readiness.
- Enforce role-based access and encryption for audit artifacts and sensitive evidence.

Artifact	Default PHI posture	Retention recommendation	Primary purpose
Raw prompt	May contain PHI	Do not store by default; store hash	Replay pointer only

Redacted prompt	PHI minimized	Store short-term with RBAC if needed	Investigation and QA
Retrieved passages	Usually non-PHI	Store hashes and source IDs	Evidence ledger
Embeddings / vector entries	Avoid PHI by default	No PHI unless justified + bounded	Retrieval performance
Untrusted model draft	May contain PHI	Do not store by default	Intermediate artifact
Final output	May contain PHI	Prefer hash; store only if required	User-facing response
LSAS Safety Report	Should be PHI-minimal	Store with RBAC + encryption; long-term	Auditability and traceability
Telemetry metrics	Non-PHI only	Store aggregated	Monitoring and drift detection

4.3 Data boundary reference diagram (conceptual)



4.4 Observability and analytics guardrails

Enterprises routinely leak sensitive data through logs and analytics events. LSAS enforces:

- No PHI in telemetry by default.
- Sanitization of error logs and traces.
- Aggregated metrics rather than raw payload retention.

- Explicit "debug mode" policies restricted to non-production or de-identified datasets.

5. LSAS Definition and Design Goals

LSAS governs AI-generated text and code through five layers: 1) Input understanding: intent classification and risk tagging. 2) Knowledge grounding: authoritative sources, curated corpora, optional live retrieval. 3) Real-time validation: validators for accuracy, compliance posture, security, and accessibility. 4) Escalation: deterministic safe behavior under uncertainty and high-risk conditions. 5) Continuous learning: policy versioning, feedback loops, regression harnesses, and regulatory watchers.

Layer	**Purpose**	**Key enforcement behaviors**	**Primary artifacts**
1. Input understanding	Classify intent and risk	Capability gating; conservative tie-breaks	Risk tags; capability decision
2. Knowledge grounding	Anchor to authoritative sources	Allowlist; evidence contract; live retrieval when required	Evidence ledger; citation map
3. Real-time validation	Validate text and code	Accuracy, compliance posture, security, a11y; transform or fail	Findings; transforms; evidence
4. Escalation	Safe behavior under uncertainty	Clarify; degrade; HITL; remediation checklists	Escalation record; rationale
5. Continuous learning	Keep controls current	Policy versioning; regression harness; watcher alerts	Releases; tests; telemetry

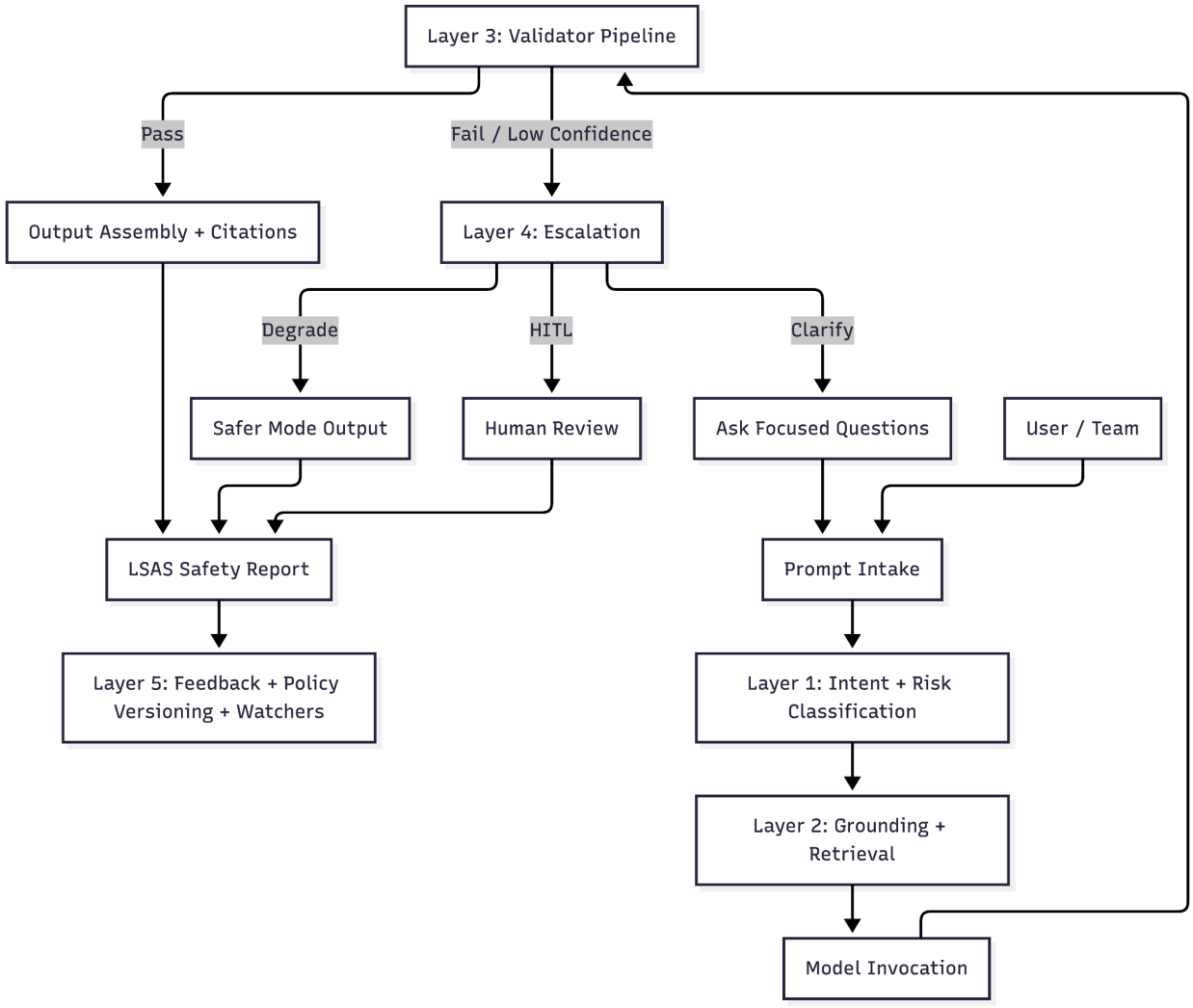
5.1 Core design goals

- Output is untrusted until validated.

- High-risk claims require evidence and citations.
- Controls are policy-driven, versioned, and testable (policy as code).
- Minimal sensitive retention with maximum auditability.
- Pluggable validators and policy packs.
- Practical latency budgets with safe fallbacks.
- Explicit alignment with healthcare and medtech lifecycle realities.

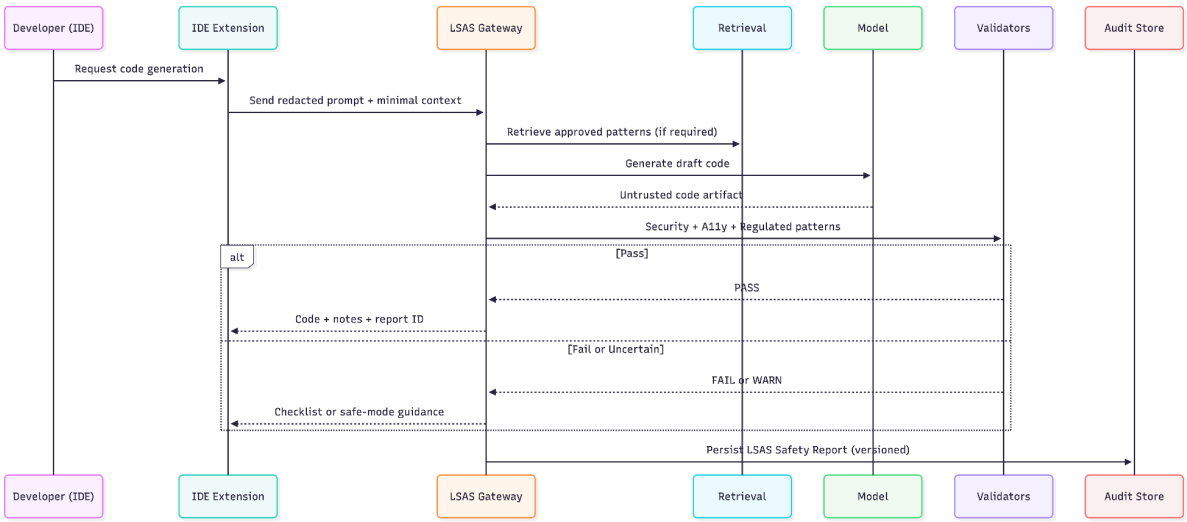
6. LSAS Reference Architectures

6.1 End-to-end pipeline



6.2 Control plane vs data plane (enterprise requirement)

6.3 IDE code generation integration pattern



7. Layer 1: Risk Classification and Capability Gating

In large organizations, risk must translate into deterministic system behavior. LSAS uses risk tags to control capabilities and validation requirements.

7.1 Practical risk taxonomy (baseline)

- Low risk: general engineering guidance, non-sensitive refactoring, generic UI patterns, no PHI, no clinical advice.
- Medium risk: security-sensitive guidance, authentication patterns, billing flows, accessibility recommendations, standards interpretation with citations.
- High risk: PHI handling, HIPAA interpretation, clinical workflow guidance that may influence care decisions, regulated device functions, production security posture, tool use against patient data.

7.2 Capability gating matrix (baseline)

Capability	Low risk	Medium risk	High risk
Citations required	Optional	Required for key claims	Required for all critical claims

Live retrieval	Optional	When standards may be current-sensitive	Required for regulatory claims unless pre-approved corpus
Code generation	Allowed	Allowed with security/a11y validators	Allowed only with full validators; may degrade to checklist
Tool use / external actions	Allowlisted only	Restricted; allowlist + evidence required	Disabled by default; HITL to enable
PHI in prompts	Disallowed	Disallowed; synthetic/de-identified only	Disallowed; strict redaction + minimum necessary
Store raw outputs	Per policy	Prefer hashed storage	Disallowed by default; store hashes + evidence metadata
Human review	Optional	Recommended for certain domains	Default for clinical/regulatory interpretation and patient-impact decisions

7.3 Reference classification object

Code block:

```
{
  "risk_level": "high",
  "domains": ["hipaa", "security", "accessibility"],
  "required_policy_packs": ["hipaa", "security", "a11y"],
  "capabilities": {
    "citations_required": true,
    "live_retrieval": "required_for_regulatory_claims",
    "code_generation": "allowed_with_full_validators",
    "tool_use": "disabled_by_default",
    "store_raw_content": false
```

```
 }
}
```

7.4 Conservative tie-breaking and failure philosophy

If classification uncertainty exists, LSAS must bias upward (more restrictive). This reduces the probability of unsafe behavior at the cost of occasional friction, which is an acceptable trade in healthcare and medtech.

8. Layer 2: Knowledge Grounding and Evidence Contracts

Grounding is not optional for high-risk claims. It is the mechanism that converts probabilistic output into defensible output.

8.1 Trusted-source allowlist (control plane)

The allowlist should store:

- Publisher authority (HHS, FDA, NIST, W3C, etc.)
- Document version and date
- Applicability and jurisdiction
- Stability expectations (fast-changing vs stable)
- Access restrictions (public vs internal policies)

8.2 Live retrieval option (staleness mitigation)

When regulatory or standard currency matters, LSAS can enforce "live retrieval required" before finalizing an answer. This is relevant for:

- HIPAA Security Rule NPRM updates [10][11]
- FDA cybersecurity guidance updates [13]
- FDA PCCP guidance and AI-enabled device change management [15]
- Accessibility standards updates, including WCAG 2.2 [19]

8.3 Evidence contract (citation gating)
For high-risk topics:

- Every critical claim must be tied to an authoritative citation.
- Unsupported claims must be downgraded or escalated.
- The Safety Report must store a citation map and evidence ledger (URL, retrieval timestamp, hash).

8.4 Retrieval poisoning and corpus governance
Large organizations will ask how LSAS prevents malicious or low-quality sources from steering output:

- Use allowlists, not open-web browsing by default.
- Treat retrieval content as untrusted until validated for provenance and integrity.
- Hash and version authoritative documents in the corpus to prevent silent drift.
- Monitor for corpus changes through the watcher and approval workflow.

9. Layer 3: Validator Architecture (Practical Engineering)
Validators are the practical core of LSAS. They must be composable, testable, and explainable.

9.1 Validator design principles
- Validators emit structured findings, not prose.
- Validators can transform artifacts (redaction, downgrading language, producing checklists).
- Policy packs decide which validators run and thresholds.
- Validators run with explicit budgets and fail-safe fallbacks.

9.2 Validator interface (reference)
Code block:

```
Validator.execute(artifact, context) -> result


artifact: { type: "text"|"code", content, citations?, metadata }
```


context: { riskLevel, domains, policyPackVersions, tenant, environment, budgets }

result: { status: PASS|FAIL|WARN, findings[], transformedArtifact?, evidence[] }

9.3 Validator catalog (enterprise baseline)

Output	Validator	Trigger	Failure behavior	Evidence captured
Text	Evidence / citation gate	Regulatory or standards claims	Downgrade language or escalate	Citation map; evidence ledger
Text	Compliance posture (tone)	High-risk domains	Transform language; escalate on ambiguity	Findings with rule IDs
Text	PHI/PII detector + redaction	All prompts/outputs	Redact; fail closed for high risk	Redaction record; hashes
Text	Scope gate (clinical advice)	Patient-impact / clinical requests	Escalate to HITL; refuse patient-specific advice	Escalation record
Code	SAST / insecure patterns scan	Medium/high risk code gen	Fail or require remediation	Findings (CWE/OWASP tags)
Code	Secret scanning	All code artifacts	Fail closed	Secret detector findings
Code	Dependency/SCA check	When dependencies suggested	Fail or block high CVEs	CVE list; SBOM pointers
Code	A11y baseline (WCAG)	UI code generation	Warn + checklist; block severe issues	A11y findings; checklist
Both	Insecure output handling gate	Tool use or execution paths	Block privileged use; require sanitization	Gate decision record
Both	Prompt injection heuristics	Untrusted user input	Strip instructions; constrain tool use	Injection flags; sanitized prompt hash

9.4 Latency budgets and deterministic fallbacks

Healthcare workflows require predictable interaction. LSAS should define time budgets per layer (example):

- Classification: 50 to 150 ms
- Retrieval: 200 to 800 ms (cacheable)
- Fast validators: 500 to 1500 ms
- Deep validators: 2 to 10 s (use only when required by policy)

If deep validation exceeds budget, LSAS should degrade output (checklist, clarifying questions, or HITL) rather than silently skipping controls.

9.5 Validators for information outputs (text)

Minimum practical validators:

- Evidence validator: require citations for regulatory and standards claims.
- Compliance posture validator: disallow overconfident language without authority; enforce scoped language.
- PHI/PII detector: redact or block leakage; enforce minimum necessary [3].
- Scope validator: block patient-specific advice and clinical recommendations unless explicitly permitted and routed through clinical governance.

9.6 Validators for code outputs (IDE)

Minimum practical validators:

- Security scan: detect OWASP-class issues, secret leakage, auth mistakes, insecure defaults.
- Insecure output handling gate: prevent generated output from being executed or used as privileged tool input without sanitization [18].
- Accessibility baseline checks: ARIA and semantic structure, keyboard navigation, focus management, error prevention patterns aligned to WCAG 2.2 [19].
- Regulated design pattern validator: enforce inclusion of audit trail hooks, consent capture points, and secure transport notes for PHI workflows.

9.7 Implementation guidance: enforce the pipeline

LSAS does not require inventing new scanners. It requires enforcing a pipeline contract:

- SAST and code scanning can be integrated into IDE and CI workflows.
- Accessibility checks can be automated and complemented by human review for complex UI flows.
- Privacy detection should be deterministic; do not "guess" what might be sensitive.
- Prefer providing remediation checklists over auto-repair in high-risk contexts unless changes are fully attributable and reviewable.

10. Layer 4: Escalation Protocols (Safe Behavior Under Uncertainty)

Escalation is deterministic behavior, not defensive disclaimers.

10.1 Escalation actions

- Ask focused clarifying questions when essential context is missing.
- Degrade to safe mode (general guidance only, no code, no PHI, no tool use).
- Provide remediation checklists instead of executable code.
- Route to human review (security, compliance, clinical) when required by policy.

10.2 Soft qualifiers for high-risk topics

High-risk outputs should include scoped qualifiers tied to evidence:

- "Based on the cited sources, here are preliminary guidelines. Confirm final interpretations with your compliance team."

Avoid blanket disclaimers. Tie uncertainty to the specific claim or missing context.

11. Layer 5: Continuous Learning, Policy Versioning, and the Regulatory Watcher

Healthcare safety posture drifts if policies are not versioned and tested.

11.1 Policy as code lifecycle

- Policy pack changes require review and approval.
- Each release is tested against golden suites and adversarial prompts.
- Releases are versioned with changelogs.
- Safety Reports capture the policy versions used for each output.

11.2 Regulatory watcher design

The watcher monitors authoritative sources and triggers review:

- HHS HIPAA guidance changes and NPRM developments [10][11]
- FDA cybersecurity and PCCP guidance updates [13][15]
- FDA CDS guidance updates [14]
- NIST updates (CSF, AI RMF) [8][9][20]
- OWASP LLM risk updates [17][18]
- WCAG updates and related accessibility guidance [19]

Watcher output should trigger review workflows, not silently change runtime behavior without approval.

12. MedTech Alignment: QMS, Risk Management, and Device Lifecycle

For medtech, LSAS is valuable only if it integrates with real quality system practices.

12.1 QMSR and controlled records

FDA's QMSR final rule incorporates ISO 13485 by reference and is enforceable on February 2, 2026 [12]. LSAS artifacts can be designed as controlled records:

- Policy packs as controlled documents (document control and approvals).

- Safety Reports as controlled records supporting validation evidence.
- Regression harness results as verification artifacts.
- CAPA hooks based on repeated validator failures.

12.2 Cybersecurity guidance and premarket documentation posture

FDA cybersecurity guidance emphasizes secure design, labeling, and documentation expected in premarket submissions for devices with cybersecurity risk [13]. LSAS contributes by gating insecure generated code and producing evidence that cybersecurity validation controls are consistently applied.

12.3 AI-enabled device change management and PCCPs

FDA provides recommendations for PCCPs to support iterative improvements to AI-enabled device software functions while maintaining safety and effectiveness [15]. LSAS supports PCCP-ready posture by versioning policy packs and validator configurations and maintaining regression harnesses that demonstrate continued safety after changes.

12.4 CDS scope boundaries

FDA CDS guidance clarifies which CDS functions may be excluded from device regulation under non-device criteria and what functions remain within device scope [14]. LSAS provides a governance boundary so that outputs related to CDS are escalated and controlled.

13. Healthcare-Specific Design Patterns LSAS Should Enforce

This section identifies practical patterns that are repeatedly missed in real products.

13.1 Minimum necessary by default

When users request patient examples or realistic datasets, LSAS should:

- recommend synthetic or properly de-identified data [5]

- block inclusion of PHI in prompts and logs
- enforce redaction in Safety Reports

13.2 Audit trails as a product design primitive

LSAS should:

- identify audit trail events in generated designs ("consent captured," "record accessed," "order submitted")
- produce audit event templates as part of output
- ensure the Safety Report captures evidence for audit readiness

13.3 Telehealth: consent, audit, and secure transport notes

For telehealth workflows, LSAS should ensure outputs include consent capture points, audit event templates, secure transport notes, and secure storage considerations for recordings and transcripts.

13.4 EHR-integrated workflows: retrieval boundaries

For EHR-integrated tools:

- tool calls must be permissioned and logged
- retrieval must be scoped to minimum necessary
- outputs must not echo sensitive fields unless explicitly required and permitted
- Safety Reports should store hashes and metadata rather than raw PHI

14. Evaluation Methodology: How to Prove LSAS Works

Top institutions will not accept "it seems safer." LSAS must be measurable.

14.1 Core metrics (recommended baseline)

Metric	Definition	Why it matters
Evidence coverage rate	% of high-risk claims with authoritative citations	Reduces unsupported assertions; improves auditability

Unsafe output rate	% outputs failing validators or requiring escalation	Measures safety posture and policy effectiveness
False positive cost	Time/blocks introduced by validators	Ensures usability remains viable for teams
Latency overhead	Added time per validator set	Controls operational cost and developer experience
Security defect rate	Findings per generated code artifact	Reduces breach and exploit risk
Accessibility defect rate	A11y findings per generated UI artifact	Reduces legal risk and improves patient access
Policy drift indicators	Changes in failure modes over time	Detects emerging risks and regressions

14.2 Testing strategy

- Golden prompts by risk category.
- Red-team prompts mapped to OWASP LLM risks [17][18].
- Regression tests per policy pack version.
- Before/after comparisons for evidence coverage, unsafe output rate, and code defect rate.

14.3 Reproducibility in academic settings

To support university-grade credibility:

- publish the LSAS specification and Safety Report schema
- publish representative (non-sensitive) test suites
- disclose assumptions and limits
- version all artifacts to enable reproducible evaluation

15. Operationalization: Governance, SDLC Integration, and Deployment

LSAS is designed to be deployed as a reusable platform boundary.

15.1 Deployment models

- Central gateway (recommended): enforce LSAS across product UIs and IDE workflows.
- Embedded library: for disconnected, offline, or constrained environments.
- Hybrid: local redaction and classification plus centralized validation and evidence storage.

15.2 Multi-tenant controls

- Tenant-scoped policy packs and allowlists.
- Data residency and logging controls by tenant.
- Per-tenant audit artifacts and access controls.
- Separation of dev, test, and prod policy packs and corpora.

15.3 Governance roles and approval paths

Role	Responsibilities	Approval scope	Typical participants
LSAS Platform Owner	Owns gateway runtime, SLOs, deployments, incident response	Runtime changes; infra controls	Platform engineering, SRE
Policy Pack Owner	Owns policy pack content and versioning	Policy releases; thresholds; allowlists	Security lead, privacy lead
Security Lead	Security validators; tool boundaries; vuln policy	Security validator updates; exceptions	AppSec, security engineering
Privacy/Compliance Lead	Minimum necessary rules; PHI handling; audit retention	PHI-related policies; retention	Privacy officer, compliance
Clinical Safety Governance	Clinical boundary decisions; HITL criteria	Clinical escalation policies	Clinician reviewers, safety committee
Product/Engineering Leads	Adoption, workflow integration, training	Project-level usage and rollout	Product, engineering managers

15.4 CI/CD and SDLC integration

LSAS becomes strongest when it is enforced as part of the development lifecycle:

- Policy pack changes are treated as code changes: reviewed, tested, and released.
- Validator pipelines run in CI to prevent regression and drift.
- IDE integrations provide early feedback while CI provides enforcement gates.
- Security and accessibility scans are treated as required quality gates for high-risk code outputs.

15.5 Incident response integration

Safety Reports should support incident investigation without amplifying exposure:

- minimal sensitive retention by default
- replay pointers and evidence hashes
- alignment with breach readiness workflows [1][4]

16. Extensions: PCI DSS, SOC 2, HITRUST CSF

LSAS is healthcare-first but should integrate cleanly into enterprise assurance demands.

16.1 PCI DSS (payments)

PCI DSS provides a baseline of technical and operational requirements designed to protect payment account data [22]. LSAS can extend via a payments policy pack to prevent account data from entering prompts/logs and to apply stricter security validators to payment flows.

16.2 SOC 2 and Trust Services Criteria

SOC 2 reports evaluate controls relevant to security, availability, processing integrity, confidentiality, and privacy [23]. LSAS provides evidence artifacts that can support such evaluations. The Trust Services Criteria provide the underlying control categories and points of focus [24].

16.3 HITRUST CSF

HITRUST CSF harmonizes multiple regulations and frameworks into a certifiable control set used widely in healthcare [21]. LSAS policy packs can map validator evidence to HITRUST-aligned controls when an organization pursues certification.

17. Common Anti-Patterns and How LSAS Prevents Them

- "Copy-paste AI code into production": LSAS enforces security and accessibility validators before code is accepted.
- "RAG as compliance": LSAS requires evidence contracts and escalates when citations are missing.
- "Logging everything": LSAS treats minimum necessary as a system property and discourages raw PHI retention.
- "Tool use without boundaries": LSAS gates tool use, requires allowlists, and logs tool actions in Safety Reports.
- "Silent drift": LSAS versions policies, runs regressions, and uses a watcher to trigger reviews.

18. Conclusion

LSAS reframes generative AI in healthcare and medtech as a governed output pipeline rather than a raw generative feature. By separating control plane policy/versioning from runtime execution, enforcing grounding and validation before finalization, and producing audit artifacts per interaction, LSAS provides a practical path for adopting GenAI without relying on informal prompt etiquette.

The next step for organizations is not to debate whether models are safe. It is to implement enforceable system boundaries that make safety measurable and repeatable. LSAS is one such boundary, designed for production.

Appendix A: LSAS Safety Report (Audit Artifact)

The Safety Report is the primary artifact that enables auditability and reproducibility while minimizing sensitive retention.

A.1 Schema goals

- Replayability: sufficient metadata to reproduce decisions without storing PHI.
- Traceability: policy versions, validator results, evidence ledger.
- Audit readiness: consistent structure across outputs.

A.2 Example JSON (illustrative)


```
{
  "lsas_version": "1.2",
  "request": {
    "id": "req_...",
    "timestamp_utc": "2026-01-23T17:02:11Z",
    "tenant_id": "tenant_...",
    "environment": "prod",
    "output_type": "code"
  },
  "classification": {
    "risk_level": "high",
    "domains": ["hipaa", "security", "accessibility"],
    "policy_packs": {"hipaa": "1.3.0", "security": "1.2.1", "a11y": "1.0.4"},
    "capabilities": {
      "code_generation": "allowed_with_validators",
      "tool_use": "disabled_by_default",
      "live_retrieval": "required_for_regulatory_claims",
      "store_raw_content": false
    }
  },
  "grounding": {
    "sources": [
      {"title": "HHS HIPAA Cloud Computing Guidance", "url":
"https://www.hhs.gov/hipaa/for-professionals/special-topics/health-information-technology/cloud
-computing/index.html", "retrieved_utc": "2026-01-23T17:01:55Z", "hash": "sha256:..."},
```



```
    {"title": "WCAG 2.2", "url": "https://www.w3.org/TR/WCAG22/", "retrieved_utc":
"2026-01-23T17:01:56Z", "hash": "sha256:..."}
   ],
   "citations_required": true
  },
  "validators": [
   {"name": "security_sast", "status": "PASS", "findings": []},
   {"name": "a11y_baseline", "status": "WARN", "findings": ["Missing focus management in modal."]},
   {"name": "insecure_output_handling_gate", "status": "PASS", "findings": []}
  ],
  "escalation": {
   "action": "remediation_checklist",
   "reason": "a11y_warn",
   "notes": "Provided checklist instead of auto-fixing high-risk UI behavior."
  },
  "output": {
   "artifact_hash": "sha256:...",
   "contains_phi": false
  }
}
```


Appendix B: Policy Pack Example (YAML Skeleton)


```
id: hipaa
version: 1.0.0
description: HIPAA-oriented governance for ePHI and PHI-sensitive outputs.
applicability:
  domains: ["hipaa", "privacy", "security"]
source_allowlist:
  - title: "HHS HIPAA Security Rule"
    url: "https://www.hhs.gov/hipaa/for-professionals/security/index.html"
  - title: "NIST SP 800-66 Rev. 2"
    url: "https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-66r2.pdf"
validators:
  - name: "phi_pii_detector"
```



```
    mode: "fail_closed"
  - name: "evidence_citation_gate"
    mode: "required_for_high_risk"
  - name: "compliance_posture_tone"
    mode: "transform_or_escalate"
thresholds:
  high_risk_requires_live_retrieval: true
escalation:
  high_risk_default: "clarify_or_hitl"
logging:
  store_raw_prompts: false
  store_raw_outputs: false
  store_hashes_and_metadata: true
```

Appendix C: Implementation Checklist (MVP to Production)

MVP (2-4 weeks):

- LSAS gateway/orchestrator.
- Risk classifier and capability gating.
- Trusted source allowlist and retrieval.
- Evidence gate and PHI detector.
- Security validator for obvious hazards.
- Safety Report generation.

Production (8-16 weeks):

- Policy as code lifecycle with approvals and releases.
- Regression harness and adversarial prompt suite.
- Full validator pipeline (security, accessibility, compliance posture).
- Multi-tenant enforcement and data residency controls.
- Incident response integration and audit readiness.
- Regulatory watcher and review workflow.

Appendix D: OWASP LLM Risk Mapping (Condensed)

OWASP LLM risk	Primary LSAS layers	Mitigation pattern	Test case example
LLM01 Prompt Injection	L1, L3, L4	Input delimiting; tool allowlists; injection heuristics; escalation	Injected instructions to exfiltrate data
LLM02 Insecure Output Handling	L3, L4	Sanitize output before execution; block privileged tool use	Model output used as a shell command
LLM03 Training Data Poisoning	L2, L5	Trusted corpus allowlists; integrity hashes; watcher alerts	Poisoned internal doc steering output
LLM04 Model DoS	L1, L3	Budgeting; rate limits; degrade behavior	Long prompts causing validator overload
LLM05 Supply Chain Vulnerabilities	L3, L5	Dependency scanning; SBOM; block risky packages	Suggested dependency with known CVE
LLM06 Sensitive Info Disclosure	L1, L3, L4	PHI detection; minimum necessary; redaction; no raw retention	Prompt containing PHI echoed in output
LLM07 Insecure Plugin Design	L1, L3, L4	Tool allowlists; scoped permissions; audit tool calls	Tool used to fetch unauthorized records
LLM08 Excessive Agency	L1, L4	Disable tool use by default; HITL for actions	Model attempts to change configs
LLM09 Overreliance	L4, L5	Escalation and evidence artifacts; require citations	User accepts unsupported claim as policy
LLM10 Model Theft	Outside LSAS core	Infra and access controls; rate limits; monitoring	Repeated probing of system prompts

References

[1] HHS. The Security Rule. https://www.hhs.gov/hipaa/for-professionals/security/index.html (accessed 2026-01-23)

[2] HHS. HIPAA Privacy Rule. https://www.hhs.gov/hipaa/for-professionals/privacy/index.html (accessed 2026-01-23)

[3] HHS. Minimum Necessary Requirement. https://www.hhs.gov/hipaa/for-professionals/privacy/guidance/minimum-necessary-requirement/index.html (accessed 2026-01-23)

[4] HHS. Guidance on HIPAA and Cloud Computing. https://www.hhs.gov/hipaa/for-professionals/special-topics/health-information-technology/cloud-computing/index.html (accessed 2026-01-23)

[5] HHS. Guidance Regarding Methods for De-identification of PHI. https://www.hhs.gov/hipaa/for-professionals/special-topics/de-identification/index.html (accessed 2026-01-23)

[6] HHS. HIPAA Security Series: Technical Safeguards (PDF). https://www.hhs.gov/sites/default/files/ocr/privacy/hipaa/administrative/securityrule/techsafeguards.pdf (accessed 2026-01-23)

[7] NIST. SP 800-66 Rev. 2 (PDF). https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-66r2.pdf (accessed 2026-01-23)

[8] NIST. Cybersecurity Framework (CSF) 2.0 (PDF). https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.29.pdf (accessed 2026-01-23)

[9] NIST. AI RMF 1.0 (PDF). https://nvlpubs.nist.gov/nistpubs/ai/nist.ai.100-1.pdf (accessed 2026-01-23)

[10] HHS. HIPAA Security Rule NPRM landing page. https://www.hhs.gov/hipaa/for-professionals/security/hipaa-security-rule-nprm/index.html (accessed 2026-01-23)

[11] Federal Register. HIPAA Security Rule NPRM (Jan 6, 2025). https://www.federalregister.gov/documents/2025/01/06/2024-30983/hipaa-security-rule-to-strengthen-the-cybersecurity-of-electronic-protected-health-information (accessed 2026-01-23)

[12] FDA. QMSR Final Rule FAQ. https://www.fda.gov/medical-devices/quality-system-qs-regulationmedical-device-current-good-manufacturing-practices-cgmp/quality-management-system-regulation-final-rule-amending-quality-system-regulation-frequently-asked (accessed 2026-01-23)

[13] FDA. Cybersecurity in Medical Devices Guidance (PDF). https://www.fda.gov/media/119933/download (accessed 2026-01-23)

[14] FDA. Clinical Decision Support Software Guidance (Jan 2026 revision). https://www.fda.gov/regulatory-information/search-fda-guidance-documents/clinical-decision-support-software (accessed 2026-01-23)

[15] FDA. PCCP Guidance (PDF). https://www.fda.gov/media/166704/download (accessed 2026-01-23)

[16] ASTP/ONC. HTI-1 Final Rule.
https://healthit.gov/regulations/hti-rules/hti-1-final-rule/ (accessed 2026-01-23)

[17] OWASP. Top 10 for Large Language Model Applications.
https://owasp.org/www-project-top-10-for-large-language-model-applications/
(accessed 2026-01-23)

[18] OWASP. LLM02: Insecure Output Handling.
https://genai.owasp.org/llmrisk2023-24/llm02-insecure-output-handling/
(accessed 2026-01-23)

[19] W3C. WCAG 2.2. https://www.w3.org/TR/WCAG22/ (accessed 2026-01-23)

[20] NIST. AI 600-1: Generative AI Profile (PDF).
https://nvlpubs.nist.gov/nistpubs/ai/NIST.AI.600-1.pdf (accessed 2026-01-23)

[21] HITRUST. HITRUST CSF Framework overview.
https://hitrustalliance.net/hitrust-framework (accessed 2026-01-23)

[22] PCI SSC. PCI DSS Document Library.
https://www.pcisecuritystandards.org/document_library/ (accessed 2026-01-23)

[23] AICPA. SOC 2 overview.
https://www.aicpa-cima.com/topic/audit-assurance/audit-and-assurance-greater-than-soc-2 (accessed 2026-01-23)

[24] AICPA. Trust Services Criteria (2022 points of focus). https://www.aicpa-cima.com/resources/download/2017-trust-services-criteria-with-revised-points-of-focus-2022 (accessed 2026-01-23)

