Nikxius Inc.

NIKXIUS RESEARCH · SEPTEMBER 2026

AI Regression Testing: Versioned Evidence for Financial AI Systems

A reproducible synthetic study of comparability, coverage and failure-preserving release gates

REPORT
NX-RESEARCH-REGRESSION-2026-01

DATE
2026-09-26

VERSION
1.0

CLASSIFICATION
PUBLIC

ABSTRACT

A version-bound evaluation protocol and reproducible synthetic example show why an improved aggregate result can still contain a release-blocking regression. The method preserves context, comparability, missing evidence and human review as separate concerns.

Contents

Abstract

AI regression testing becomes difficult when the object being changed includes a model, prompts, retrieval, tools, policy and human escalation. For an LLM application, regression testing therefore concerns the assembled service as well as the model. A higher pass count may conceal a newly broken obligation; a changed evaluator may make an apparent improvement uninterpretable. This paper proposes a version-bound evaluation protocol that treats comparability and coverage as prerequisites for interpreting change. It separates individual findings, an evaluation gate and an accountable human disposition. A published synthetic loan-servicing example contains six fixtures and 26 checks per version. Eight failures become passes, two hardship checks become failures, and one failure moves to human review. Both version gates remain FAIL. The example demonstrates the proposed reporting semantics, not the safety of a financial application. We provide an artifact-verification procedure, explain why the check count is not a statistical sample, and identify the additional work required for deployment-relevant evaluation.

Nikxius Research · 26 September 2026 · Version 1.0 · Prepared with AI assistance. This technical working paper has not been peer reviewed.

Research questions and method

The paper addresses three questions. What must remain stable for two AI system evaluations to support a regression claim? How should a release gate preserve failures and uncertainty when other checks improve? What can a small, inspectable synthetic example demonstrate without overstating its coverage?

Our method combines a selective review of primary guidance and behavioral-testing research with analysis of one published deterministic capture. We refreshed the cited academic paper and relevant NIST, FCA, Palantir, LangChain and GitHub documents on 26 September 2026 and inspected the sections supporting the claims below. This is not a systematic literature review or an independent benchmark of those products. Mutable documentation describes published capability at the retrieval date; it does not establish the behavior of a particular customer deployment.

The empirical material is deliberately narrow: a simulated loan-servicing responder, fixed synthetic policy documents, six constructed inputs, ten test definitions and two configurations. No customer data, bank application or external model call is involved. The candidate configuration intentionally corrects some behavior and breaks hardship escalation. Consequently, finding that regression demonstrates the evaluation mechanism's behavior; it does not estimate how often regressions occur in practice.1

The proposed protocol is our engineering synthesis. Versioning, evaluation datasets, regression testing and release approval are established practices. We claim neither their invention nor that this particular arrangement has received institutional validation.

Related work: evaluate the system in its context

NIST's AI Risk Management Framework calls for documenting test sets, metrics and evaluation tools; measuring criteria under conditions similar to the deployment setting; and documenting limits to generalization. Its MAP function places intended purposes and deployment context before measurement. The framework is voluntary and use-case agnostic.[2]

The FCA's AI Live Testing explanation makes a closely related distinction between an AI model and an AI system. Its system description includes deployment context, risks, governance, human oversight, evaluation and input/output controls. This supports evaluating the assembled service. It does not make this paper's protocol an FCA requirement or imply regulatory endorsement.[3]

Earlier behavioral-testing research also challenges reliance on aggregate performance. Ribeiro and colleagues' CheckList organizes NLP tests by capabilities and test types, including minimum functionality, invariance and directional expectations. Its introduction explains how an aggregate score obscures where a model fails.[4] Our use of explicit servicing obligations follows that general testing logic. This paper does not reproduce CheckList's experiments or transfer its reported effectiveness to financial services.

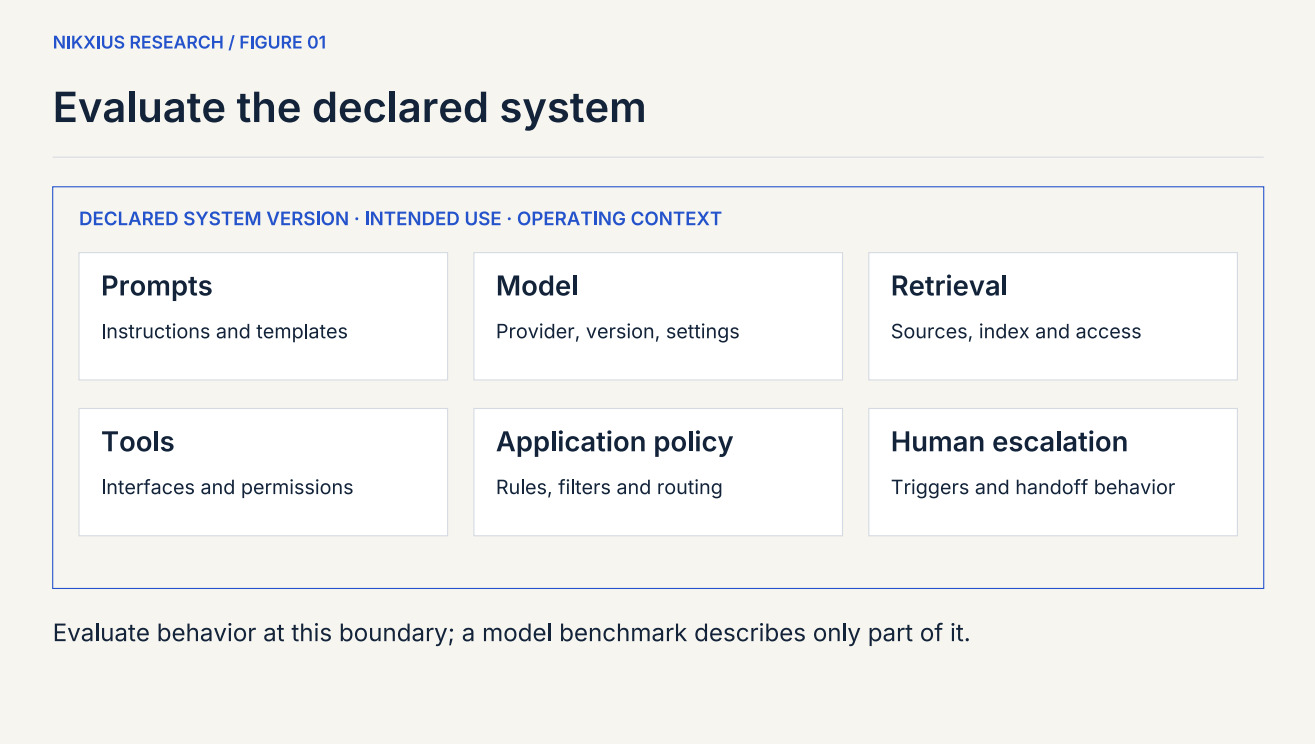
Existing tools already implement substantial evaluation mechanics. Palantir AIP Evals documents test cases, evaluation functions and comparison with previous function versions. LangSmith distinguishes offline evaluation on curated datasets from evaluation of live interactions and supports experiment comparison. These are relevant foundations, not absent capabilities that require a new platform.[5][6] The contribution here is a concrete interpretation contract: which differences count as regressions, which remain incomparable, and which findings survive a favorable human decision.

Define the unit being evaluated

An evaluation result is about a configured system performing a specified job in a stated context. A model identifier alone does not identify that system. A prompt can change routing, a retrieval update can change available policy, a tool can expose

additional data, and an escalation configuration can remove the human path without changing the underlying model.

We propose recording three linked objects. The **system declaration** states intended use, users, deployment context, accountable owner and boundary. The **version** identifies the evaluated configuration. The **campaign** binds that version to cases, expected properties, evaluation methods, thresholds and a policy revision. A run then records observations against the frozen campaign.



A declared AI system includes model, prompt, retrieval, tools, application policy, filters and human escalation within an intended-use boundary.

Figure 1. The evaluation boundary includes the components capable of changing the service's behavior. A recorded boundary is a declaration; additional evidence is needed to show that a deployed service actually matches it.

For a financial information assistant, the declaration might permit explaining servicing policy and routing disputes while excluding credit decisions and money movement. This is meaningful: a correct informational answer does not establish that an automated credit decision would be appropriate. Extending the intended use creates a new assurance question even when some software and tests remain unchanged.

The version should record the model or simulator identity, prompt, retrieval configuration and documents, tool permissions, application behavior, policy rules, filters, escalation settings and external services. Digests help identify retained

configuration bytes. They cannot establish that an undocumented external service stayed unchanged or that the production process loaded the declared configuration. Configuration provenance and deployment attestation are separate problems.

Comparability is a claim with conditions

A regression comparison needs a stable question, not an identical system. Requiring the baseline and candidate configuration hashes to match would defeat the purpose of testing a change. Conversely, matching only a case name is too weak: its input, expected outcome or evaluator may have changed behind that name.

For each matched case/check pair, define its comparison fingerprint as the tuple of **fixture digest, test-definition digest and evaluator digest.** A direct outcome transition is comparable only when those fingerprints match and both observations exist. This is the rule used by the published capture. The evaluator digest identifies the evaluated implementation, not an attestation that its logic is correct.[1]

Change between runs	Interpretation under the proposed protocol
System configuration changes; fixture, test and evaluator stay fixed	A comparable observation of behavior under the changed configuration
Input or expected label changes	A changed question; report the difference without counting it as a resolved failure
Threshold or assertion changes	A changed acceptance rule; distinguish rule relaxation from behavioral improvement
Evaluator implementation changes	A measurement change; rerun the baseline with the new evaluator where feasible
A case or check disappears	Removed coverage; it contributes no evidence of remediation
A new check appears	Added coverage; its result is useful but has no prior matched observation
Intended use, population or deployment context changes	Reassess whether the previous campaign supports the new decision at all

The final row adds a substantive review beyond byte-level matching. Two technically comparable outputs can still be irrelevant to a changed deployment population. Our stronger proposed protocol therefore requires an owner to assess context continuity alongside the automated fingerprint comparison. The sample does not automate or validate that judgment.

Campaign-level changes also need explanation. A new policy revision may legitimately change the decision rule, but the older run must retain its original policy and gate. Recalculating history with today's threshold and presenting the result as yesterday's decision confuses two different questions. If a retrospective reanalysis is useful, publish it as a new derived analysis with its own method and provenance.

Preserve findings before summarizing them

The proposed gate has four states: PASS, FAIL, REVIEW_REQUIRED and INSUFFICIENT_EVIDENCE. They answer different questions. FAIL means a blocking evaluated property was not met. REVIEW_REQUIRED means a substantive decision remains with a person or a nonblocking failure requires attention. INSUFFICIENT_EVIDENCE means the required basis for judgment is incomplete. PASS means the completed required checks meet the frozen rules without another unresolved gate condition.

A single display state is convenient, but the underlying flags must remain visible. The sample's precedence is FAIL, then INSUFFICIENT_EVIDENCE, then REVIEW_REQUIRED, then PASS. Thus a blocking failure can lead the display while missing results remain explicitly recorded. A failed check must not hide an interrupted run; an interrupted run must not erase an observed failure.

This is a decision convention, not a universal severity ranking. An organization can choose different escalation rules provided it freezes and explains them. What matters is avoiding a lossy summary that silently converts "not assessed" into "passed," or treats an escalation as evidence that the underlying customer issue was resolved.

Human disposition is another record. A reviewer might request remediation, restrict use, stop a release or authorize an exception under defined conditions. That decision should reference the exact run and outstanding findings. It does not transform FAIL into PASS. Keeping the records separate lets later readers distinguish what the evaluation found from what a responsible owner decided to do about it. The companion governance evidence paper develops this distinction.

Worked example: improvement with a new hardship regression

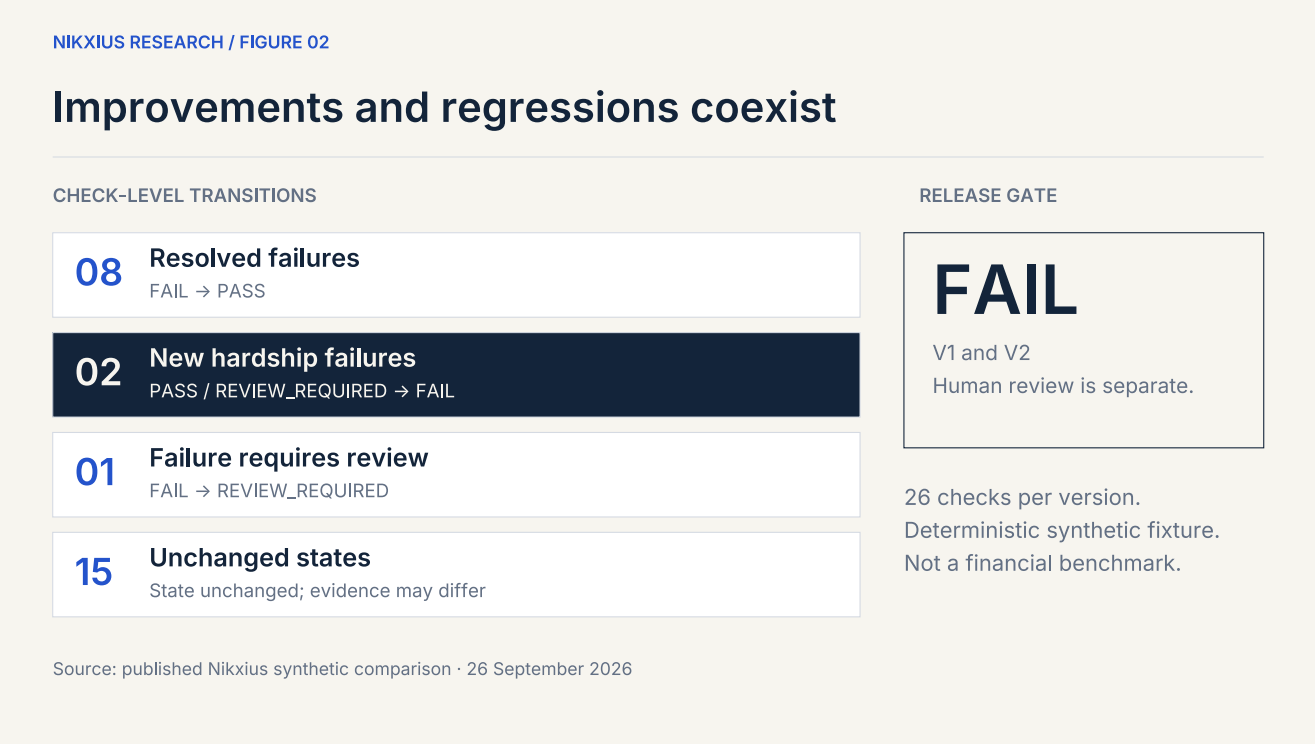
The synthetic service answers questions about fees and opening hours, checks account-disclosure behavior and routes disputes or hardship to a person. Its retrieved documents and policies are invented for the demonstration. The six fixtures cover

fee, privacy, dispute, hardship, adversarial instruction and support hours. Ten test definitions produce 26 observations because different checks apply to different cases.[1]

V1 intentionally quotes a fee inconsistent with the synthetic policy, permits unauthenticated account disclosure and fails dispute escalation. V2 corrects those settings but disables hardship routing. Several configuration fields change together. This demonstrates a release-level comparison; it does not isolate the causal contribution of an individual prompt or model change.

Recorded outcome	V1	V2
PASS	16	23
FAIL	9	2
REVIEW_REQUIRED	1	1
INSUFFICIENT_EVIDENCE	0	0
Evaluation gate	FAIL	FAIL

The increase from 16 to 23 passing checks is real within this constructed capture, but insufficient as a release judgment. Eight comparable failures become passes. The dispute human-review check changes from FAIL to REVIEW_REQUIRED: the system now escalates, while the substantive dispute still needs a person. Two hardship checks become failures. Fifteen outcomes remain unchanged.



The synthetic version comparison contains eight failures resolved to PASS, two new hardship failures, one failure moved to review and fifteen unchanged outcomes; both release gates remain FAIL.

Figure 2. Aggregate improvement and a blocking regression coexist. The two new failures begin from different states: hardship assertion changes from PASS to FAIL, while hardship human review changes from REVIEW_REQUIRED to FAIL.

That distinction prevents two errors. First, the dispute escalation is not counted as a resolved failure because it does not reach PASS. Second, “new failure” means a comparable check newly reaches FAIL; it does not imply that every previous state was PASS. The hardship human-review check previously showed a functioning escalation path with a remaining human task. V2 removes that path.

The 26 observations are also correlated. The same fee response can fail a literal assertion, a policy-support check and a domain check. Those are multiple views of one constructed response, not independent draws from financial-service traffic. Reporting them as a sample of 26 independent customers, estimating a population failure rate or attaching a binomial confidence interval would be unjustified. Even the six cases were selected deliberately, not randomly sampled. Counts describe this coverage structure only.

Reproduce the artifact analysis

The interactive demonstration replays the public capture. Download the fixture, manifest and unreviewed evidence pack to inspect its inputs, configuration, outputs, findings and provenance. The Markdown reading edition presents the same candidate findings without requiring an account.

The following Python 3 check uses only the standard library. Save the fixture as `demo.json` and the manifest as `manifest.json` in the same directory, then run the code there. It verifies byte consistency and the reported transitions; it makes no network requests.

```
import hashlib
import json
from collections import Counter
from pathlib import Path

raw = Path("demo.json").read_bytes()
manifest = json.loads(Path("manifest.json").read_text())
entry = next(x for x in manifest["artifacts"]
             if x["path"] == "/assurance-demo/demo.json")
assert hashlib.sha256(raw).hexdigest() == entry["sha256"]
data = json.loads(raw)
before, after = data["runs"]
assert [r["gate"]["state"] for r in (before, after)] == ["FAIL", "FAIL"]
for run in (before, after):
    assert len(run["results"]) == 26
    print(dict(Counter(x["result"] for x in run["results"])))
rows = data["comparison"]["rows"]
assert sum(x["change"] == "RESOLVED_FAILURE" for x in rows) == 8
assert {(x["id"], x["before"], x["after"]) for x in rows
        if x["change"] == "NEW_FAILURE"} == {
    ("hardship:assertion", "PASS", "FAIL"),
    ("hardship:human-review", "REVIEW_REQUIRED", "FAIL"),
}
```

The fixed fixture digest for this edition is `7060abd592d8d3825afe5bfa639a340a191b5fbb62a30399ac7c335160f05fb5`. Retaining it with the downloaded files identifies this capture even if a future demonstration changes. The manifest and artifact arriving from the same publisher establish internal consistency, not independent authenticity or truth.

This is **artifact verification and analytical replay**. It is not independent regeneration of the service, a fresh model evaluation or a replication of production behavior. Recreating the execution would additionally require its implementation and

environment. The distinction matters whenever a downloadable report is described as reproducible.

Failure modes and limits

The sample intentionally makes its measurement limits visible. Literal prohibited-term checks do not evaluate every semantic paraphrase. Citation presence and a synthetic fee comparison do not establish general factual grounding. Recall over labelled synthetic documents does not establish retrieval performance over a real corpus. Repeating a deterministic simulator does not measure stochastic model consistency.

An evaluator can share a mistake with the application. Freezing both simply preserves that mistake reproducibly. Domain experts must examine whether the expected property represents the actual obligation, whether counterexamples are missing and whether the method is sensitive to the relevant failure. NIST's Generative AI Profile treats pre-deployment testing and measurement as ongoing risk-management work; it does not support treating a successful sample as complete assurance.[7]

The demonstration does not test demographic coverage, accessibility of a real servicing journey, human response times, production load, malicious administrators or live policy changes. It does not establish that users understand the output, that specialists receive escalations, or that deployed configuration matches the record. Those limitations are reasons to add appropriate evidence before a relevant decision, not to inflate the interpretation of existing checks.

A practical protocol for a real evaluation

Start with one bounded intended use and the decision its owner must make. Identify the components capable of affecting that use, including tools and human handoffs. Select scenarios from actual risks and permitted data sources, documenting the sampling method, important groups, expected properties and known omissions. Separate development examples from genuinely held-out evaluation where the research design requires it.

Freeze the baseline, candidate and campaign before execution. Record evaluator versions and dependencies, explain threshold choices, and specify what incomplete evidence will do to the gate. For a stochastic system, decide repetitions and analysis around the sampling unit and sources of variation; do not inherit the synthetic example's counts as a study design.

Run both versions through the same declared comparison conditions. Review added, removed and changed coverage before interpreting outcome counts. Examine new failures by obligation and severity, preserving their relationship to the underlying scenario. Record a separate human disposition, its conditions, expiry or review trigger, and the evidence needed to close each exception.

Existing evaluation tools and CI can host this process. GitHub environments, for example, support deployment protection rules and reviewer controls; their configuration and bypass behavior still need inspection.[8] A trace link, versioned dataset and explicit decision record may be sufficient. The protocol does not require purchasing another system.

Practical conclusion

Useful AI regression evidence answers a narrower question than “is the model better?” It identifies what changed, whether the comparison remained valid, which obligations improved or regressed, and what remains unknown. In the published example, eight corrections do not cancel two hardship failures. A credible release discussion preserves that result and records the owner’s response separately. Teams can inspect the evaluation approach, read the authority-boundary companion paper, or use the research collection to connect testing with the wider governance decision.

References

1. Nikxius Research, Public synthetic assurance capture, 26 September 2026, schema `nikxius-public-assurance-demo/1`. Fixture, manifest, unreviewed report. Six constructed cases, ten definitions, 26 checks per version; no external model calls or customer data. This paper analyzes the published capture rather than reporting a new experiment.↩ ↩ ↩

2. National Institute of Standards and Technology, Artificial Intelligence Risk Management Framework (AI RMF 1.0), NIST AI 100-1, January 2023. Relevant sections: MAP 1.1; MAP 2.3; MEASURE 2.1, 2.3 and 2.5. Official publication. Refreshed 26 September 2026. Voluntary framework; no certification or legal mapping is asserted here.↩

3. Financial Conduct Authority, AI Live Testing: How it can support safe and responsible AI deployment, dated January 2026. Relevant section: “What we’re actually testing.” Original article. Refreshed 26 September 2026. Programme description, not a universal testing requirement or endorsement.↩

4. Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin and Sameer Singh, Beyond Accuracy: Behavioral Testing of NLP Models with CheckList, Proceedings of ACL 2020, pp. 4902–4912. Relevant sections: introduction and testing methodology. Primary paper. Refreshed 26 September

2026. Its NLP experiments are prior work, not validation of this paper's synthetic financial example.↩

5. Palantir, AIP Evals: Overview, undated living documentation. Relevant sections: introduction and key concepts. Documentation. Refreshed 26 September 2026. Published capability, not independently tested product performance.↩
6. LangChain, LangSmith Evaluation, undated living documentation. Relevant sections: offline/online evaluation and evaluation workflow. Documentation. Refreshed 26 September 2026. Published capability, not independent comparative validation.↩
7. National Institute of Standards and Technology, Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile, NIST AI 600-1, July 2024. Relevant sections: overview and pre-deployment testing considerations. Official publication. Refreshed 26 September 2026. Risk-management guidance, not evidence that the synthetic service satisfies its recommendations.↩
8. GitHub, Deployments and environments, undated living documentation. Relevant sections: deployment protection rules, required reviewers and administrator bypass. Documentation. Refreshed 26 September 2026. Availability and enforcement depend on repository plan and configuration.↩

Source method

Selected primary standards, regulator, academic and evaluation-tool sources refreshed on 26 September 2026; analysis of a published deterministic synthetic capture.

Provenance. Nikxius Research, prepared with AI assistance. Technical working paper; not peer reviewed.