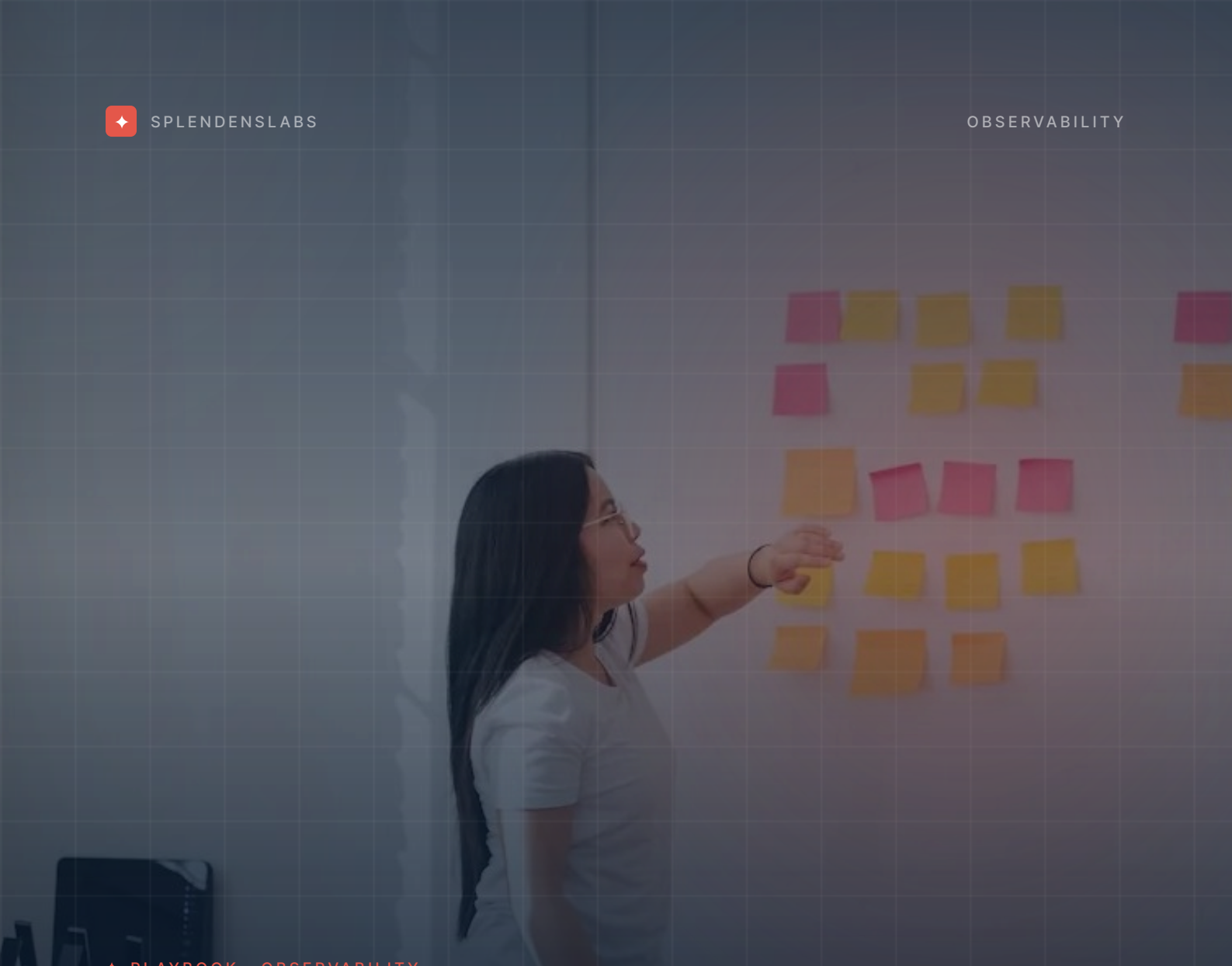


✦ PLAYBOOK · OBSERVABILITY

Evaluation Frameworks

Golden sets, calibrated LLM-as-judge rubrics, human review, offline and online evals, CI regression gates and trajectory evaluation for agents — with metrics by system type.

FORMAT	LEVEL	READ TIME	UPDATED
Playbook	Intermediate	16 min	Oct 2026

OVERVIEW

What this playbook covers

Without evaluation, every prompt change, model upgrade or retrieval tweak is a guess, and quality drifts without anyone noticing. This playbook lays out a practical evaluation framework: a golden set built from real usage, automated scoring with rubric-based judges calibrated against people, human review where judgment matters, regression gates in CI, and online evaluation in production — plus how to evaluate agents by their trajectory, not just their final answer.

KEY TAKEAWAYS

- Build the golden set from real user inputs, including hard, ambiguous and unanswerable cases.
- Prefer deterministic checks wherever possible; use LLM judges for what code cannot score.
- Calibrate every LLM judge against human labels before trusting it, and re-check after judge or rubric changes.
- Gate releases in CI on eval results, with thresholds per metric and per critical slice.
- Evaluate agents on trajectory — tools chosen, order, arguments, efficiency — as well as outcome.
- Offline evals prevent regressions; online evals catch the cases you did not anticipate.

WHO IT'S FOR

AI and ML engineers

QA and evaluation engineers

Engineering managers owning AI releases

Product owners defining quality bars

HOW TO USE IT

Work through sections 01–03 to build your first eval suite, use 04 to wire it into CI, and refer to 05–06 for agents and system-specific metrics.

CONTENTS

01 The evaluation stack
02 Building the golden set
03 Scorers: checks, judges and humans
04 Regression gates in CI
05 Evaluating agents by trajectory
06 Metrics by system type

01 / FOUNDATIONS

The evaluation stack

An evaluation framework is a pipeline: datasets go in, scorers run, results are compared to a baseline, and a decision comes out.

LAYER 01 **Datasets**	Golden set · Regression cases · Adversarial set · Sampled production traffic
↓	
LAYER 02 **Runners**	Offline batch runs · CI on pull requests · Shadow / canary runs · Online sampling
↓	
LAYER 03 **Scorers**	Deterministic checks · Reference comparison · LLM-as-judge rubrics · Human review
↓	
LAYER 04 **Analysis**	Baseline comparison · Slice breakdowns · Variance & confidence · Failure clustering
↓	
LAYER 05 **Decisions**	Release gate · Rollback triggers · Review queues · Backlog of fixes

OFFLINE EVALUATION

- Fixed datasets, repeatable runs
- Runs before release, in CI
- Compares versions fairly
- Only as good as the dataset

ONLINE EVALUATION

- Real production traffic, sampled
- Runs continuously after release
- Catches new inputs and drift
- Needs redaction and privacy controls

✦ 02 / DATASETS

Building the golden set

The golden set defines what “good” means for your system. If it does not look like real usage, your scores will not predict real quality.

01 **Collect real inputs**

Pull from logs, support tickets, pilot users and subject-matter experts — not inputs the build team invented.

02 **Stratify by slice**

Tag each case by intent, difficulty, language, user type and risk level so results can be broken down.

03 **Write expected outcomes**

Reference answers, required facts, expected sources or expected tool calls, written or approved by domain experts.

04 **Include the hard cases**

Ambiguous questions, unanswerable questions, adversarial inputs and permission-restricted cases.

05 **Version and grow it**

Store it in version control; add every confirmed production failure as a regression case.

SIZE

Start small, grow steadily

A hundred or so well-chosen cases beats thousands of synthetic ones to begin with; expand as failures surface.

SYNTHETIC

Use with care

LLM-generated cases fill coverage gaps but skew toward easy phrasing; review and mix with real ones.

HYGIENE

Keep it out of prompts

Never use golden cases as few-shot examples, or scores will overstate quality.

✦ 03 / SCORING

Scorers: checks, judges and humans

Use the cheapest scorer that measures the property reliably. Escalate to LLM judges and people only for what code cannot decide.

SCORER	GOOD FOR	WATCH OUT FOR
Deterministic checks	Schema validity, exact fields, required citations, banned content	Too strict on correct paraphrases
Reference metrics	Extraction and classification accuracy, retrieval recall@k	Text-overlap metrics correlate poorly with quality for free text
LLM-as-judge	Faithfulness, relevance, tone, rubric adherence	Position and length bias, self-preference, drift when the judge model changes
Human review	Domain correctness, safety, nuanced judgment	Slow and costly; inconsistent without guidelines

Calibrating an LLM judge

01 **Write an explicit rubric**

Define each score level with concrete criteria and examples; prefer pass/fail or a short scale over fine-grained scores.

02 **Label a calibration set**

Have domain experts score a sample independently; measure agreement between experts first.

03 **Compare judge to humans**

Measure judge–human agreement per criterion; inspect disagreements and refine the rubric or prompt.

04 **Reduce known biases**

Randomize order in pairwise comparisons, ask for reasoning before the verdict, and avoid judging with the same model being evaluated.

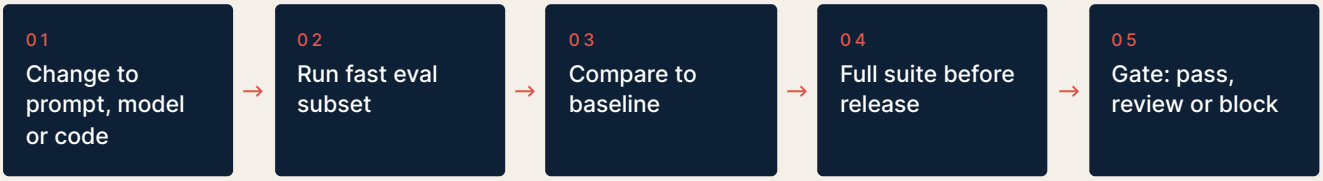
05 **Re-calibrate on change**

Re-run calibration whenever the judge model, rubric or task changes.

04 / RELEASE

Regression gates in CI

Evals earn their keep when they block a bad change automatically. Treat the eval suite like a test suite with budgets.



01

Tiered suites

A fast subset on every pull request; the full suite nightly and before each release.

02

Thresholds per metric and slice

Block if an aggregate falls below the bar, or if any critical slice — such as safety cases — regresses.

03

Account for variance

Model outputs vary between runs. Run multiple samples on key cases and treat small differences as noise.

04

Diff, don't just score

Show which cases flipped from pass to fail so reviewers can judge whether a regression is real.

FIELD NOTE

Pin everything the eval depends on

Record model version, prompt version, index snapshot, judge version and dataset version with every run. Otherwise a score change cannot be attributed to your change.

> If you cannot say how you would know it got worse, you are not ready to ship it.
>
> SPLENDENSLABS ENGINEERING PRINCIPLE

05 / AGENTS

Evaluating agents by trajectory

An agent can reach the right answer the wrong way — an unsafe call, a wasted loop, a lucky guess. Score the path as well as the destination.

DIMENSION	QUESTION	HOW TO SCORE
Outcome	Was the task completed correctly?	End-state checks against the environment or reference
Tool selection	Did it call the right tools?	Compare against expected tool set; flag forbidden tools
Arguments	Were tool arguments valid and correct?	Schema checks plus reference values
Ordering	Were dependent steps done in a sensible order?	Expected partial order, not an exact sequence
Efficiency	How many steps, tokens and retries did it take?	Budgets per task; compare with baseline
Safety & policy	Did it respect approvals and limits?	Policy-violation count must be zero

Practical notes

- Run agents against sandboxed or mocked tools with deterministic responses so runs are reproducible.
- Allow multiple valid trajectories; score against constraints rather than a single golden path.
- Use an LLM judge on the full trace for qualitative questions, such as whether the agent asked for clarification when it should have.

✦ 06 / METRICS

Metrics by system type

Pick a small set of metrics per system that map to what users care about, and report them by slice.

SYSTEM	CORE METRICS	SUPPORTING METRICS
RAG assistant	Faithfulness, answer correctness, refusal accuracy	Retrieval recall@k, citation accuracy, latency
Extraction / classification	Field-level precision and recall, accuracy per class	Schema validity, confusion by class
Summarization	Faithfulness, coverage of key points	Length compliance, readability
Conversational / support	Resolution, correctness, policy compliance	Escalation rate, tone, user feedback
Agents	Task success, policy violations	Steps, cost and time per task

DATASETS

- [ ] Golden set from real inputs, tagged by slice
- [ ] Unanswerable and adversarial cases included
- [ ] Production failures added as regression cases

SCORING

- [ ] Deterministic checks used wherever possible
- [ ] LLM judges calibrated against human labels
- [ ] Rubrics versioned alongside prompts

PROCESS

- [ ] Fast subset in CI; full suite before release
- [ ] Thresholds per metric and critical slice
- [ ] Online evals on sampled traffic with alerts

✦ GET IN TOUCH

Let's Talk

We set up evaluation frameworks that teams actually use to ship — golden sets, calibrated judges and CI gates. Bring us a system whose quality you cannot yet prove.

- Golden set design with your domain experts
- LLM-judge rubrics and calibration
- CI regression gates and online evals

BOOK AN AI STRATEGY SESSION →

EMAIL
hello@splendenslabs.com

WEB
splendenslabs.com

OFFICE
Sona Incubation Foundation, Salem