

AI Agent Evaluation Checklist

A release gate for enterprise AI agents, adapted from the ReadyForRole guide “Evals for Agents: Measuring Enterprise AI Agents Beyond Accuracy”.

Quick answer. An agent should be judged by whether it reached the right end state in your systems, by a path you would approve, at a cost you can afford — measured continuously, not once at UAT.

1. The eight-point evaluation gate

Score each item before approving an agent for production or promoting it to a new release. Treat every “No” or “Partial” as a blocker to resolve or a risk to document.

GATE	QUESTION	STATUS
1. Task measurability	Is success defined as a verifiable end state for each task class the agent handles?	☐ Yes ☐ Partial ☐ No
2. Scenario coverage	Does the suite cover routine, edge, adversarial and known-failure cases, owned by a business expert?	☐ Yes ☐ Partial ☐ No
3. Trajectory checks	Are tool choice, arguments, step limits and policy adherence asserted, not just final answers?	☐ Yes ☐ Partial ☐ No
4. Groundedness	Are the agent's claims checked against retrieved sources, with citations verified?	☐ Yes ☐ Partial ☐ No
5. Consistency	Is each scenario run multiple times, with reliability across runs reported?	☐ Yes ☐ Partial ☐ No
6. Judge calibration	Are LLM-as-judge scores validated against human labels, with agreement tracked over time?	☐ Yes ☐ Partial ☐ No
7. Release gating	Does every change to prompt, model, tool or index trigger the suite with pass/fail thresholds, including cost and latency?	☐ Yes ☐ Partial ☐ No
8. GCC / regulatory fit	Are evals segmented by entity, language and jurisdiction, and packaged as evidence for model risk review?	☐ Yes ☐ Partial ☐ No

2. Six failure modes and the design decision that fixes each

FAILURE MODE	DESIGN DECISION
Grading the final answer only	Assert end states directly in sandboxed systems of record, and evaluate trajectories: tool choice, argument correctness, step count and policy adherence.
The static golden set	Turn every incident, escalation and reviewer rejection into a versioned regression case, and refresh coverage from sampled production traffic each release cycle.
Uncalibrated LLM judges	Write per-criterion rubrics, measure judge-human agreement on a labelled sample, and re-calibrate whenever the judge model changes.

Single-run scoring	Run each scenario several times and report both whether it can succeed and whether it succeeds consistently across every run.
Cost and latency as afterthoughts	Report cost and latency per successful task alongside success rate, set budgets per task class, and fail a release when either regresses beyond tolerance.
Silent dependency drift	Pin model versions, trigger the suite on every dependency change, and canary new versions against live traffic with online evals before full rollout.

3. Seven actionable takeaways

- Define task success as a verifiable end state in the system of record, not a well-worded answer.
- Build scenario suites with sandboxed tools, and grow them from every production failure.
- Evaluate trajectories: tool choice, argument correctness, step budgets and policy adherence.
- Run scenarios multiple times and report consistency, not a single lucky pass.
- Calibrate LLM judges against human labels before trusting their scores.
- Report cost and latency per successful task as release criteria.
- Gate every prompt, model, tool and index change on the evaluation pipeline.

4. Release sign-off

AGENT / RELEASE	BUSINESS OWNER	EVALUATION LEAD	DATE

Source: ReadyForRole, "Evals for Agents" (readyforrole.com/blog/evals-for-agents.html). This checklist is a general practice aid and does not constitute legal, audit or regulatory advice.