

PRACTITIONER · 6 MINUTE READ

Evaluations: testing AI before you trust a change

An evaluation is a fixed set of example tasks you rerun before and after any change, to see whether the AI improved or got worse.

What it is

An **evaluation** (often shortened to **eval**) is a fixed set of example tasks with known good answers, or with a checklist for scoring them. You run the same set before and after you change something, then compare the results.

The change might be a new prompt, an edited skill (see guide 11), a different model (see guide 03), a new connector or a new tool. Any of these can improve one thing and quietly break another. An eval shows you which.

Think of a driving test route. Every learner drives the same roads, and the examiner uses the same marking sheet. That makes results fair and comparable. An eval does the same for your AI setup: same tasks, same scoring, every time.

Why it matters

- A change that looks better on one example can be worse on others.
- Model updates and new tools arrive often; an eval tells you whether to switch.
- Saved results give you evidence, not opinions, when deciding what to keep.
- Each failure you find becomes a new test, so the same mistake is caught next time.
- It keeps a person's judgement at the centre of decisions about quality.

How to do it

1. **Collect** 10 to 20 real examples of the task, removing names and confidential details first, or use made-up ones.
2. **Write** what good looks like for each: the right answer, or the points a good answer must contain.
3. **Choose** simple scoring: a pass or fail checklist, or a 1 to 5 **rubric**: a scale with a sentence describing each score.
4. **Run** the current setup on every example and record the scores. This is your **baseline**.
5. **Make** one change at a time, run the full set again and compare with your baseline.
6. **Spot-check** the results yourself; a named person reads a sample of answers, not just the scores.
7. **Add** every new failure you meet in real work as a new test case.
8. **Track** results over time in a simple table: date, change, score, notes.

Example

A team uses a saved instruction to draft replies to customer questions about deliveries. They want to try a shorter, clearer version.

Test set: 15 past customer questions, with names and order numbers replaced.

Checklist for each reply:

```
[ ] Answers the actual question asked
[ ] Gives the correct delivery policy from [policy document]
[ ] Promises nothing the policy does not allow
[ ] Polite tone, no jargon
[ ] Under [150] words
Pass = all five ticked
```

Results table:

Date	Change	Passed	Notes
[date]	Current instruction	11/15	Two too long
[date]	Shorter instruction	13/15	One new error on refunds

The shorter version scores higher overall, but it introduced a regression: a wrong statement about refunds that the old version got right. The team fixes the instruction, adds the refund question as a permanent test, reruns all 15, and the reply owner reads a sample before switching.

Common mistakes

- **Testing on one or two examples.** Fix: use at least 10, covering easy, typical and awkward cases.
- **Changing several things at once.** Fix: change one thing per run, so you know what made the difference.
- **Letting AI grade AI with no human check.** Fix: an assistant can help score against your checklist, but a person reviews a sample and makes the final call.
- **Vague scoring ("is it good?").** Fix: write specific checks a colleague could apply the same way.
- **Using real customer data in an unapproved tool.** Fix: anonymise examples and use only approved tools.

Try it this week

1. Pick one task you regularly give to an AI assistant (1 minute).
2. Gather 10 past examples and remove any personal or confidential details (4 minutes).
3. Write a five-point pass or fail checklist for a good answer (3 minutes).
4. Run your current prompt on all 10 and score them (5 minutes).
5. Save the results with today's date, ready for your next change (2 minutes).

Going further

- Ask a second approved assistant, such as Claude, ChatGPT, Microsoft Copilot or Gemini, to score against your checklist, then compare its scores with yours on a sample to see where it disagrees.
- Keep a small "never again" set of past failures that every change must pass.
- Rerun the eval when your provider updates the model, even if you changed nothing.
- Store test cases, checklists and results together in a shared folder, alongside the harness (see guide 23).
- Feed lessons from failures into memory and learning loops (see guide 24).

Key words

- **Evaluation (eval):** a fixed set of tasks used to test AI before and after a change.
- **Test case:** one example task with a known good answer or checklist.
- **Rubric:** a scoring scale that describes what each score means.
- **Baseline:** the score of your current setup, used for comparison.
- **Regression:** when a change makes something worse that used to work.