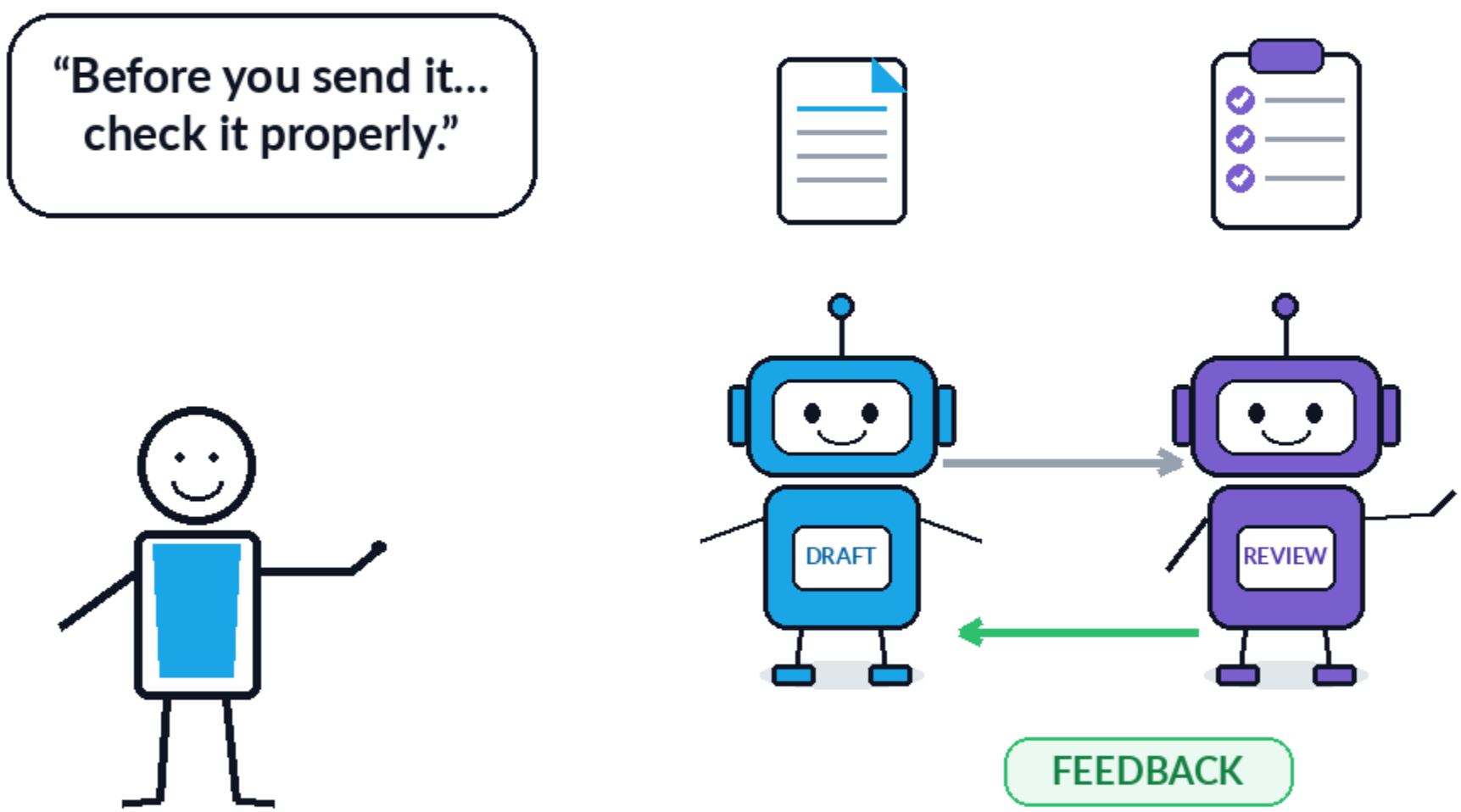
CAN AN AI AGENT CHECK ITS OWN WORK?

Reflection, verification and the evaluator–optimizer loop — explained simply.



First answer → feedback → improved answer.

THE FIRST ANSWER IS NOT ALWAYS THE BEST

Agents can be fast and capable — but they can still miss important details.

WRITING

Clear idea, but the tone is wrong.

RESEARCH

Useful sources, but one claim is unsupported.

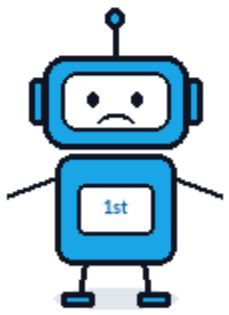
CODE

The function works, but fails an edge case.

SUPPORT

The answer is correct, but policy approval is missing.

A second pass can catch what the first pass missed.



Improvement needs feedback — not just another random attempt.

THE HUMAN ANALOGY: WRITER + EDITOR

Good work often improves through a simple review cycle.

1

DRAFT

Create the first version.

2

REVIEW

Compare it with clear criteria.

3

REVISE

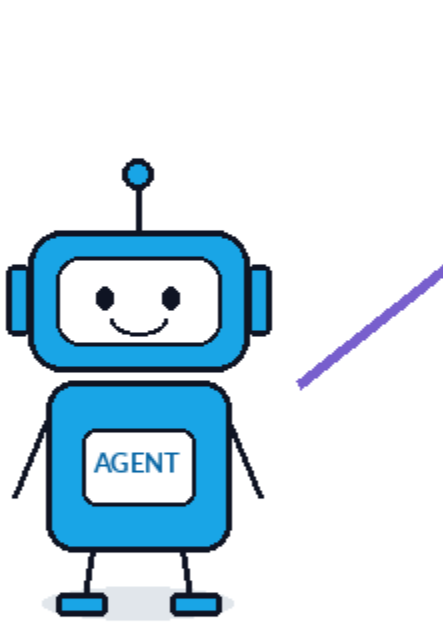
Fix the specific problems.

Repeat only when the feedback can improve the result.

An AI evaluator plays the role of the editor.

WHAT DOES “REFLECTION” MEAN?

It is a system design pattern — not self-awareness.



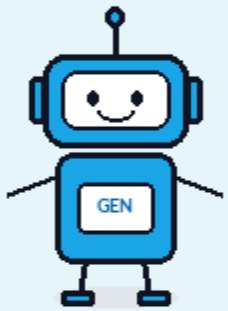
Reflection means the agent:

- checks its output or action result
- compares it with clear criteria
- turns problems into feedback
- uses feedback for the next attempt

It does not “feel regret.” It follows a review process you designed.

THREE DIFFERENT JOBS

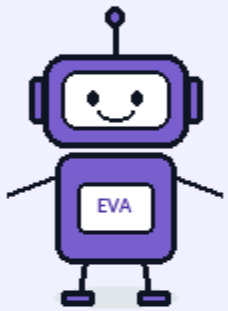
Keeping the roles clear makes the loop easier to trust and debug.



GENERATOR

Creates the draft or takes the action.

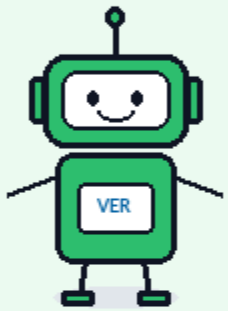
"Here is my answer."



EVALUATOR

Judges quality against a rubric.

"Two criteria are not met."



VERIFIER

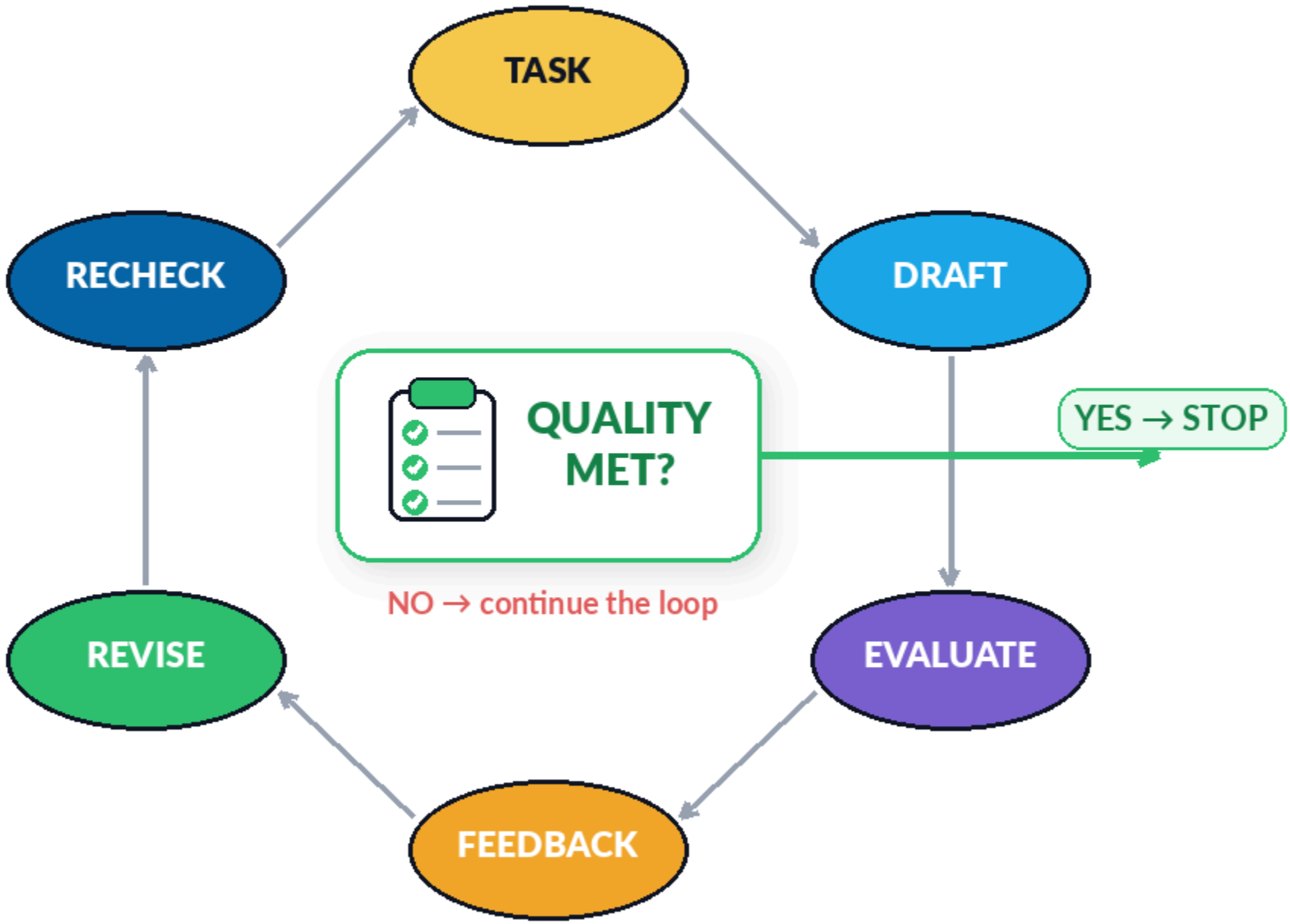
Checks facts, rules or results using evidence.

"The test failed."

One model may perform all three jobs — but they are still different jobs.

THE EVALUATOR–OPTIMIZER LOOP

One part creates. Another evaluates. Feedback drives the next version.



A loop without a clear stop rule can run forever.

EXAMPLE 1: A CUSTOMER EMAIL

The evaluator turns vague dissatisfaction into specific corrections.

FIRST DRAFT

"Your refund is delayed. Please wait."

→

EVALUATOR

- No apology
- No expected date
- No next step
- Tone feels dismissive

→

IMPROVED

"I'm sorry for the delay. Your refund is being processed and should arrive within 3–5 business days. I'll notify you if anything changes."

Good feedback names the problem and tells the optimizer what to change.

EXAMPLE 2: A RESEARCH REPORT

"Looks professional" is not the same as "supported by evidence."

REPORT DRAFT

Claim: "Demand grew 40% last year."

⚠ Citation points to a general news article — not the 40% figure.

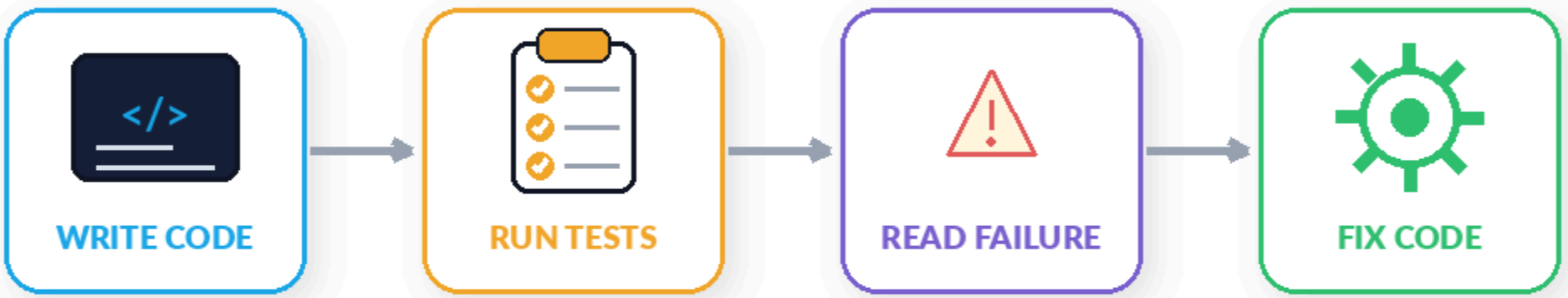
VERIFY THE CLAIM

- Open the cited source
- Find the exact supporting passage
- Check date, geography and sample
- Mark unsupported or uncertain claims

Verification checks the evidence — not only the writing quality.

EXAMPLE 3: A CODING AGENT

Software provides strong external feedback: tests either pass or fail.



TEST RESULT

FAIL: calculate_discount(-1)

Expected: validation error

Received: -0.10

ACTIONABLE FEEDBACK

"Add input validation for negative values."

External tests are usually more reliable than asking, "Does this code look correct?"

A GOOD EVALUATOR NEEDS A RUBRIC

A rubric converts “Is this good?” into answerable checks.

CORRECT — Are the facts, calculations or actions right?

COMPLETE — Did it cover every required part?

GROUNDED — Are important claims supported by evidence?

SAFE — Did it follow permissions and risk rules?

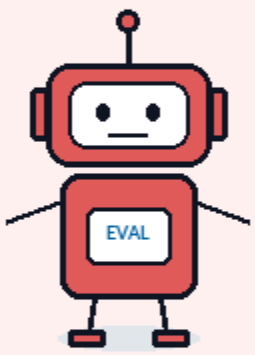
USEFUL — Is the result clear, relevant and usable for the person?

Use task-specific criteria. A legal memo and a birthday caption need different rubrics.

VAGUE FEEDBACK vs ACTIONABLE FEEDBACK

The optimizer can only fix what the evaluator explains clearly.

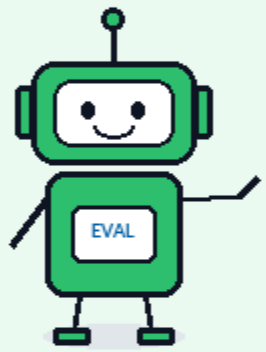
VAGUE



"Make it better."

The optimizer must guess what "better" means.

ACTIONABLE



"The answer omits the refund timeline and gives no next step. Add both in two sentences."

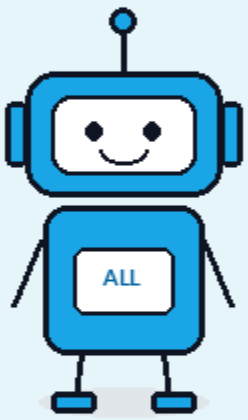
Specific issue + required change + useful constraint

Strong feedback is short, specific and tied to the rubric.

SAME MODEL OR A SEPARATE EVALUATOR?

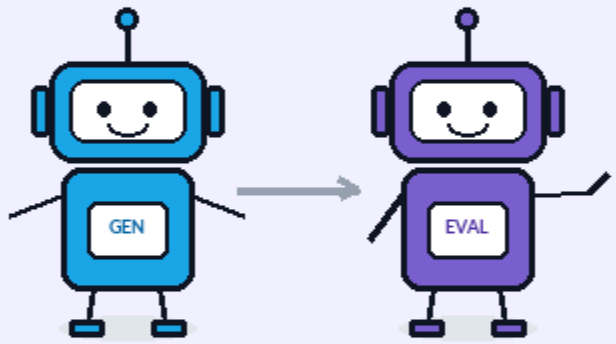
Both designs can work. Choose based on risk, cost and available checks.

SAME MODEL



- Simple to build
- Lower cost and latency
- Useful for low-risk refinement
- May repeat its own blind spots

SEPARATE EVALUATOR



- Clearer separation of roles
- Can use a different model or prompt
- May reduce shared bias
- Still needs external evidence

Independence helps — but no LLM judge is automatically objective.

VERIFICATION TOOLS BEAT "AI OPINIONS"

Whenever possible, check the result against the real world.

CALCULATOR

Recompute numbers

TEST SUITE

Run expected cases

SCHEMA CHECK

Validate structure

SOURCE LOOKUP

Confirm evidence

POLICY ENGINE

Check rules and permissions

DATABASE

Verify live status or records

Ask the model to interpret evidence — not invent the evidence.

WHEN SHOULD THE LOOP STOP?

A good agent knows when to improve, when to stop and when to ask for help.

CRITERIA MET

All required checks pass.

MAX ATTEMPTS

Retry limit reached.

NO IMPROVEMENT

Score is not getting better.

BUDGET LIMIT

Time or cost is exhausted.

HUMAN REVIEW

The decision needs judgment or approval.

"Try again forever" is not a production strategy.

COMMON FAILURE MODES

Reflection can improve quality — but it can also create new problems.

SHARED BLIND SPOT

Generator and evaluator make the same mistake.

RUBRIC GAMING

The output improves the score, not the real user outcome.

ENDLESS POLISHING

Each pass changes wording without adding value.

FALSE CONFIDENCE

A neat explanation makes an unverified answer look trustworthy.

More loops do not automatically mean more truth.

Use independent signals and measure whether the loop actually helps.

WHEN IS THIS PATTERN WORTH USING?

Reflection adds cost and latency, so use it where the extra pass matters.

GOOD FIT

- Quality can improve through revision
- Success criteria are clear
- Mistakes are expensive
- External checks are available
- The task produces a reviewable artifact

POOR FIT

- A deterministic rule already solves it
- There is no meaningful rubric
- The result cannot be verified
- Latency matters more than polish
- The action is high-stakes without human oversight

Use the simplest review mechanism that reliably catches the important mistakes.

A SAFER PRODUCTION PATTERN

Reliable improvement comes from the whole system — not one clever prompt.

1 **CLEAR RUBRIC** — Define what "good" means before generation.

2 **STRUCTURED FEEDBACK** — Return failed criteria and exact corrections.

3 **EXTERNAL CHECKS** — Use tests, sources, validators and live data.

4 **LIMITED RETRIES** — Set turn, time, token and cost budgets.

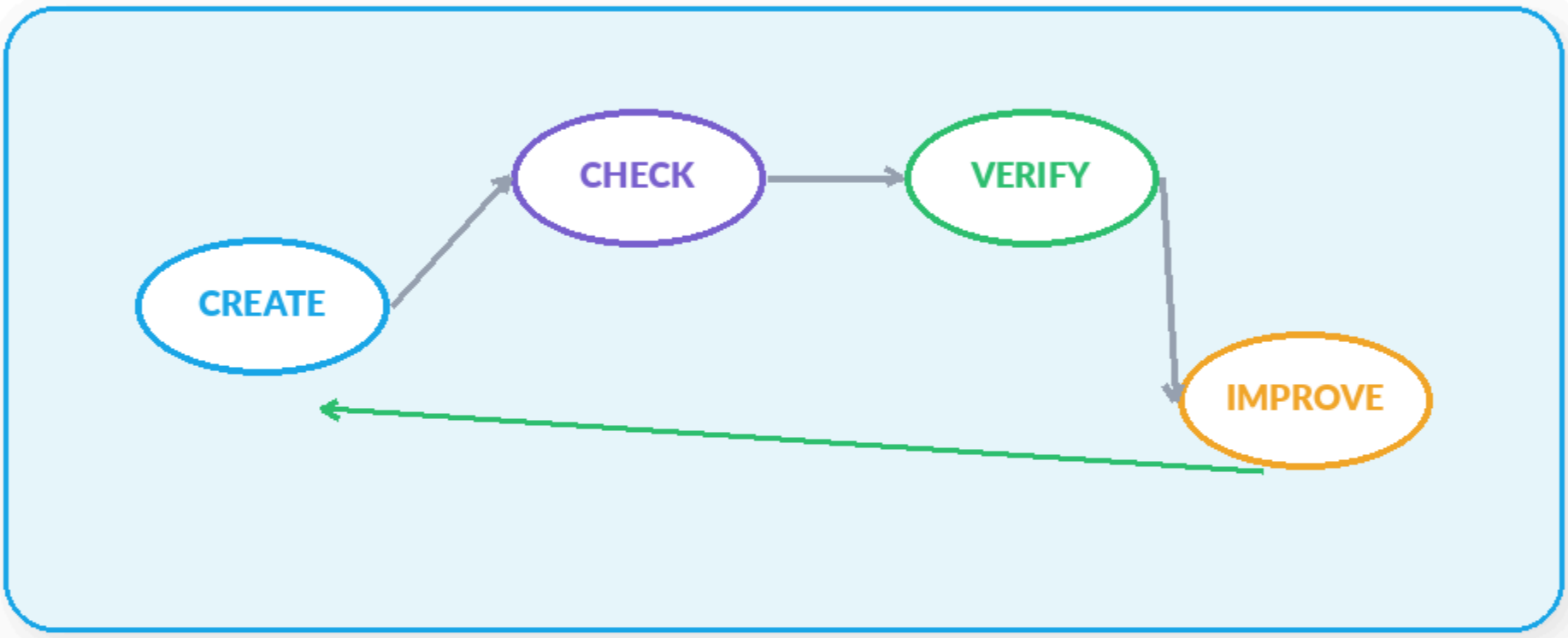
5 **TRACES + EVALS** — Record what happened and test across many examples.

6 **HUMAN APPROVAL** — Pause before consequential or irreversible actions.

Review the final answer and the path the agent took to produce it.

DAY 14 TAKEAWAY

Reliable agents do not only act — they check, learn and stop safely.



Evaluator–optimizer loops can improve an agent's output when they use clear criteria, useful feedback, external verification, limited retries and human oversight.

NEXT UP · DAY 15: Long-running agents, checkpoints and resumability

What task would you trust an agent to review and improve?