Teamulate

A PRACTICAL GUIDE FROM TEAMULATE

The Automation Health Check Kit

Why does automation stop working after launch, and who notices?

A practical guide from Teamulate. Write down what one automation promises to deliver, check the result and not just the run, plan for the run that never starts, make retries safe, and name who gets the signal and what they do next.

Five checks for working

Received | Processed | Correct | In place | On time

Who this is for

Owners and marketing leads of small businesses who have one or more automations or AI workflows running, such as routing inquiries, updating the CRM or drafting replies, and who would only find out about a problem when a client complains.

You need the automation's run history, access to the system where its result should appear, and the person who built it.

The idea in one paragraph

> A green run in the automation tool is one layer of proof, not the result. An automation can finish without an error and still miss an inquiry, update the wrong record, create a duplicate or leave a task with no owner. Worse, a run that never started may produce no error at all. "Working" should mean a confirmed result, on time, in the right place, with a known path back when something fails.

PART 1

What the documentation says, and what it does not

Source	What it says	What it does not show
1 **Google SRE book, monitoring**	Black-box monitoring is symptom-oriented ("the system isn't working correctly, right now"); white-box monitoring can show failures masked by retries; the four golden signals are latency, traffic, errors and saturation	How often small business automations break
2 **Same Google SRE chapter, on paging**	Every page should be actionable; if a page merits only a robotic response, it should not be a page	A response time that suits your business
3 **Google Identity, OAuth 2.0**	A refresh token can stop working, for example when the user revokes access, the token is unused for six months, or time-based access expires	That your connections are about to fail
4 **Amazon Builders' Library, retries**	Retries can be ineffective if all clients retry at once; jitter spreads them out; implementations cap backoff and limit the number of retries	A retry policy that fits every tool
5 **Amazon Builders' Library, idempotent APIs**	A client request identifier lets a service recognize a repeated request so a retry does not cause a second side effect	That your tools support this
6 **Stripe documentation, webhooks**	Endpoints might receive the same event more than once, and Stripe does not guarantee events arrive in the order they were generated	How other providers behave; check each one
7 **n8n documentation, error handling**	An error workflow runs when an execution fails; some error details are missing when the error is in the trigger node, because the workflow does not execute	That an error workflow catches a run that never started
8 **Google Cloud, generative AI evaluation**	Describes computation-based metrics against a reference answer and rubric-based, model-based evaluation	That an AI judge's approval proves an answer is true

The method in this guide is our proposal. It is built from this documentation.

PART 2

What "working" means: five checks

Check	Question	Evidence
1 **Input received**	Did the expected event, record or task reach the process?	A record in the source with a matching ID
2 **Processing done**	Did the needed technical steps run?	The run history for that ID
3 **Result correct**	Does the record, label, draft or calculation meet the agreed criteria?	A check against written rules or a sample review
4 **Result in the right place**	Is the result in the right system, with the right owner, findable?	The record in the target system
5 **On time**	Did the work finish while it still mattered?	The time from input to confirmed result

If the last check is out of date or monitoring has stopped, the status is Unknown, not Working. No error is not a reason for a green status.

PART 3

Three kinds of problem, three kinds of check

Problem	What it looks like	How you catch it
1 **Visible failure**	The run returns an error: access denied, service down, invalid input	Error alerts from the tool
2 **Silent miss**	No error, but the expected work is missing. A scheduled run never started	Compare expected work with work actually done
3 **Wrong result**	The run finished and wrote a record, but the record is wrong	Data and quality checks on the result

One general "workflow failed" alert does not cover all three. A run that never started has no failed run to report.

Why a working process breaks. Access changes, a field is renamed, a business rule changes while the logic stays the same, an outside service slows down or rate-limits, the volume grows, the AI model, prompt or source changes, or tasks keep going to someone who left. These are hypotheses to check against the records, not verdicts.

PART 4

Worksheet: the operating contract for one automation

Fill in one copy for each automation. Keep it where the person who checks the automation will find it.

Automation: _____ Business owner: _____ Technical owner: _____ Backup: _____

Field	What to write down	Filled in?
1. Trigger	What starts it, and when it is expected to run	Yes / No / Unknown
2. Input	What it accepts, and what it must reject	Yes / No / Unknown
3. Output	What it must produce, and where that result appears	Yes / No / Unknown
4. Proof	How you confirm the result for one ID	Yes / No / Unknown
5. Time limit	How long is acceptable from input to result	Yes / No / Unknown
6. Approvals	Which actions need a person to approve, such as sending	Yes / No / Unknown
7. On failure	What happens to the input, and who is told	Yes / No / Unknown
8. Exceptions	Who handles cases the automation cannot	Yes / No / Unknown

If the automation prepares a draft, success is "draft ready for review", not "reply sent". Show that difference in the tool too.

PART 5

Catching the run that never started

- An error handler reacts to a recorded error. If a scheduled run does not start, or the event never reaches it, there may be no error to react to.
- For a scheduled automation, check: when should it have started, is there a start record, is there a confirmed result, and has the allowed window passed?
- For an automation triggered by incoming work, compare accepted items in the source with matching records in the target, and explain any gap.
- Zero incoming items is not proof of a failure. There may simply have been no demand that day.
- Clearly labelled test records can help when volume is low. Keep them separate from real results, send them only to permitted test targets, and remember they cover only part of real cases.

PART 6

Retries and duplicates

- A retry can help with a temporary failure. It does not fix wrong permissions, a missing required field, a wrong rule, a wrong recipient, an outdated source or a poor AI result.
- A timeout does not prove the outside action failed. A blind retry can create a second record or a second message.
- Retry only suitable error types, with a limit on attempts and total time, spread out over time, and within the provider's limits.
- Give each business task one ID, and reuse it on every retry. Use the provider's duplicate protection where it exists.
- If the result is unknown and the outside system offers no protection, send the task to a person to check. Endless retrying is not recovery.

Partial success. A process can create the CRM record, add the task and then fail to send the notice. Starting everything again may create a second record and a second task. Keep a log of each step: confirmed, failed, unknown, safe to repeat, needs a check. Resume from the right step, and remember that rolling back the automation does not undo what it already did.

PART 7

Who gets the signal, and what it says

Role	What they do
1 **Business owner**	Knows what a correct result is and what a delay costs
2 **Technical owner**	Checks the connection, settings and runs
3 **Backup**	Takes the case when the main owner is away
4 **Decision owner**	Approves restarting, changing rules or any action outside the automatic scope

- In a small team one person can hold several roles. What matters is that no responsibility is unnamed.
- A useful signal says: which automation, which result is missing or doubtful, since when, how many tasks are affected, what was already checked, whether any outside action happened, who should act and the next safe step.
- Never put keys, tokens or unnecessary client message text in shared alerts.
- Three proposed levels: urgent (risk of a wrong outside action, lost important work or ongoing data damage: stop the affected part and find the owner), respond within an agreed window (tasks blocked or late, input kept, no dangerous action), and for the record (one temporary error recovered by itself, with the result confirmed).
- Group repeated errors from one incident into one case. At low volume, show counts, ages and importance, not just percentages.

PART 8

AI quality is its own check

Check	Question
1 **Structure**	Are the needed fields, types and allowed values present?
2 **Grounding**	Are the claims supported by the permitted sources?
3 **Business fit**	Are the category, next step and limits right?
4 **Permission**	Is the proposed action allowed?

- AI can return a valid format with the wrong meaning: the right fields with the wrong category, a readable summary that drops a deadline, a draft with an old price.
- Keep a set of realistic, permitted examples with expected results, and run the same checks whenever the model, prompt or source changes.
- An AI judge can support review, but its agreement is not independent proof. Critical cases need checkable rules or a person.

PART 9

Ten test cases to run before you trust it

Run each in a separate test copy, with test data, one fault at a time, after one normal control run.

Test	Fault to introduce	What you should see
1 **Access removed**	A test connection loses permission	Error recognized, no endless retries, input kept, a task to restore access
2 **Run never starts**	The scheduled trigger is switched off	The missing run is noticed when the window passes
3 **Field renamed**	A required field gets a new name	Input rejected with a clear reason, nothing half written
4 **Same event twice**	The same test inquiry is delivered twice	One record, not two
5 **Timeout**	The outside step answers too late	Result checked before any retry
6 **Partial success**	The notice step fails after the record is made	Resume from the notice, no second record
7 **Wrong AI label**	A test case the AI should label as a fit	The quality check flags the wrong label
8 **Owner gone**	The assigned person is inactive	The task goes to the backup
9 **Volume spike**	Many test items at once	A queue with visible age, nothing silently dropped
10 **Rule changed**	A service is no longer offered	Old rule caught by the contract review

Ten single tests do not cover every combination of failures. Combinations are the next step.

PART 10

Checklist before you call it running

- [ ] The automation has a written operating contract with all eight fields.
- [ ] "Working" is defined by the five checks, not by a green run.
- [ ] There is a check for the run that never started.
- [ ] Retries are limited, spread out and reuse one task ID.
- [ ] Partial success has a log and a resume point.
- [ ] A business owner, a technical owner, a backup and a decision owner are named.
- [ ] AI output has structure, grounding, fit and permission checks.
- [ ] At least the normal control run and one fault test have been run in a test copy.

Limits of this guide

- We read official documentation and engineering guidance from Google, Amazon, Stripe and n8n.
- How often your automations break, and what response time fits, show in your own run logs.
- Nothing here promises that any platform guarantees the business result, or that any setup delivers each action exactly once across connected systems.

Want a second pair of eyes?

If you would rather not set up these checks alone, Teamulate uses this method to watch results, handle exceptions and recover safely as part of the marketing operations our AI marketing team runs with you, with every step visible. Request a demo at https://teamulate.ca/request-demo/

Sources

- Google SRE book: Monitoring Distributed Systems. https://sre.google/sre-book/monitoring-distributed-systems/
- Google Identity: Using OAuth 2.0 to Access Google APIs (refresh token expiration). https://developers.google.com/identity/protocols/oauth2
- Amazon Builders' Library: Timeouts, retries, and backoff with jitter. https://aws.amazon.com/builders-library/timeouts-retries-and-backoff-with-jitter/
- Amazon Builders' Library: Making retries safe with idempotent APIs. https://aws.amazon.com/builders-library/making-retries-safe-with-idempotent-APIs/
- Stripe documentation: Receive Stripe events in your webhook endpoint. https://docs.stripe.com/webhooks
- n8n documentation: Error handling. https://docs.n8n.io/flow-logic/error-handling/
- Google Cloud: Gen AI evaluation service overview. https://cloud.google.com/vertex-ai/generative-ai/docs/models/evaluation-overview