Preventing Unapproved Changes: How to Build Compliance-Friendly Automated Workflows

A design guide for automating work without letting anything consequential happen unapproved

Direct answer: how do you stop an automated workflow making unapproved changes?

Design the automation so it does the preparation, checking, routing, and drafting, but gate any consequential action — a change, an approval, a send, a payment, a record update — behind an explicit human approval step. Log every automated step in an audit trail that records what was done, when, and under whose approval. And when the system is uncertain, have it stop and escalate to a person rather than push through. Nothing consequential should ever happen without a named human signing off first.

The principle: separate “prepare” from “commit”

Almost every workflow can be split into two halves. The **prepare** half is everything leading up to a decision: gathering the inputs, checking them against rules, drafting the document, working out who needs to approve, and assembling the context. The **commit** half is the single consequential moment — the action that actually changes something in the world or the record.

Automation earns its keep in the prepare half, where the work is repetitive and reversible. The commit half is where accountability lives, so it stays with an authorised person. Keeping those two halves distinct is what makes a workflow compliance-friendly: you can automate aggressively without ever changing who is accountable for the decision.

> **The core rule**
>
> Automate the preparation, checking, routing, and drafting around a decision. Keep the decision itself with the authorised person, and log everything.

The anatomy of an approval gate

An approval gate is the checkpoint between prepare and commit. A well-built gate has three parts, and each one should be explicit rather than assumed:

- **What triggers it.** The gate fires whenever the workflow reaches a consequential action — anything that writes, sends, pays, or changes a record. Reversible prep steps flow through freely; the gate only stands in front of the commit.
- **Who can approve.** The gate names the authorised person or role for that decision, so approval authority is defined up front rather than left to whoever happens to be online. Uncertain or out-of-policy cases route to the right person, not around them.
- **What's recorded.** When the gate is cleared, it captures what was approved, when, and by whom — so the approval itself becomes part of the audit trail, not a decision that happened invisibly.

The key property is that the gate **blocks**: the consequential action cannot proceed until a human clears it. A gate that merely notifies someone after the fact is not a gate.

Mapping a workflow: what the AI does vs what needs sign-off

For any workflow you want to automate, sort each step into one of two columns. If a step only prepares, checks, routes, or drafts, the AI can own it. If a step commits something consequential, it requires human sign-off. The table below shows the pattern across common steps.

Workflow step	AI does (prepare)	Requires human sign-off (commit)
Incoming request / document	Reads, extracts, and summarises the key details; flags anything unusual	Any decision to accept, reject, or action the request
Checking against rules	Compares inputs to policy, spots gaps or exceptions, and explains what it found	Approving an exception or overriding a rule
Drafting a response or document	Produces a draft with the right template, tone, and details filled in	Sending, publishing, or finalising the document
Routing and assignment	Works out who should approve and assembles the context for them	The approver's actual decision on the item
Updating a record or system	Prepares the exact change and shows a before/after preview	Committing the change to the live record
Payment or financial action	Assembles the payment details and checks them against the source documents	Authorising the payment

How to make it auditable

A compliance-friendly workflow is one you can reconstruct after the fact. That means every automated step is logged — not just the final action, but the preparation and checks that led up to it. A useful audit record answers three questions for each step:

- **What** was done — the specific action or draft the AI produced.
- **When** it happened — a timestamp for the step.
- **Under whose approval** it proceeded — the named person who cleared the gate for any consequential action.

The log should be exportable, so you can hand it to whoever needs it without screenshots or reconstruction. If you cannot export a record of what the system did, you cannot answer a public-records request, an auditor, or a regulator. In a regulated setting, the workflow also has to sit inside your existing obligations — public-records laws, HIPAA (where health data is involved), SOX (where financial controls apply), and your records-retention schedules — rather than route around them.

> **Why the log matters**
>
> The audit trail is what turns "the automation did something" into "on this date, this step ran, and this person approved it." That distinction is the whole difference between a system you can defend and one you cannot.

Failure behaviour: what happens when the AI is unsure

The most important behaviour in a compliance-friendly workflow is what it does when it is *not* confident. A well-designed system does not guess and push through; it stops and hands the case to a person. Three mechanisms make that reliable:

- **Confidence thresholds.** When the AI's certainty about a step falls below the bar you set, that step does not proceed automatically — it is held for review rather than committed on a guess.
- **Exception queues.** Items that do not fit the normal path — malformed inputs, missing information, out-of-policy cases — land in a queue for a human to handle, instead of being forced through the happy path.
- **Stop-and-escalate.** When something is genuinely uncertain or unusual, the workflow pauses and escalates to the right person, rather than choosing the least-bad option on its own.

> **Fail safe, not fail silent**
>
> A workflow that stops and asks when it is unsure is behaving correctly. The failure mode to design out is the one where the system quietly pushes an uncertain action through because nothing told it to stop.

Checklist: what a compliance-friendly workflow must have

- **Prepare and commit are separated** — the automation handles the reversible prep; consequential actions are gated.
- **Every consequential action has an approval gate** that blocks until a named, authorised person signs off.
- **Every automated step is logged** with what was done, when, and under whose approval — and the log is exportable.
- **Uncertain cases stop and escalate** via confidence thresholds and an exception queue, rather than pushing through.
- **The workflow runs inside your existing environment** rather than copying records out to a separate system.
- **Value is proven on a scoped pilot** against your own baseline before you widen it.

How SG1 builds this

These are the defaults we build to. Consequential actions go through human-in-the-loop approval checkpoints; every AI action is logged in an audit trail; the AI runs on Microsoft Azure and is never trained on your data; it works inside your existing Microsoft 365 tenant rather than copying records out; and when the system is uncertain, it stops and escalates instead of pushing through.

We start with a scoped pilot on one workflow, measured against your own baseline, so you can see the prepare/commit split and the audit trail working on real work before widening it.