02

AI & AUTOMATION

CHOOSING THE RIGHT TOOL

(AND WHY)

Greg@Automationwithgreg.com

A note before you start

If you've read Start Here, you've already done the hardest bit. You've started to look at your own working week and ask whether bits of it could be done smarter. This playbook picks up where that one left off. Start Here was about the why. This one is about the how.

Specifically: how to look at a task and recognise which kind of tool fits, what the trade-offs are, and what to watch out for when the tool you've been given isn't actually the right one for the job.

You probably didn't pick the tools on your desk. Most readers of this playbook had their tools chosen for them. An AI assistant rolled out across the company. A workflow tool you were told to learn. Maybe both. That's normal. The job isn't to use everything; it's to recognise what each kind of tool is good at, so the one on your desk gets pointed at the right work.

There's no homework, no quiz, and nothing to install. About twenty minutes of reading, a cup of tea, and a slightly clearer view of the toolkit you've inherited.

> The one idea in this playbook: get the task right, and the tool usually picks itself.

Section 1: It starts with the task, not the tool

Most conversations about AI and automation start in the wrong place. They start with the tool. “We're rolling out Gemini.” “We've bought licences for n8n.” “We need an agent strategy.” The tool is the noun, and everything else has to arrange itself around it.

That's the wrong way round.

The tool is a means. The work is the end. When something is taking too long, feels fiddly, or you dread it, the useful first question isn't “which tool should I use?” It's “what kind of task is this?”

Get that right and the tool usually picks itself. Get it wrong, and you'll spend an afternoon trying to coax an LLM into doing something a five-line email rule could have handled in a minute.

This isn't a complicated discipline. It's a habit. The four tools you'll meet in this playbook all fit on a desk; what changes is the kind of task each one's good at. Once you can recognise the task, the choice is mostly obvious.

> Judge the task, not the tool.

Section 2: Every tool has the same shape

Before we look at the individual tools, here's a frame that makes all four easier to recognise. Whether it's an email rule, a workflow, an LLM, or an agent, they all fit the same simple shape:

Trigger → Steps → Output.

Trigger

Something starts the process. It might be a time of day (“every Monday at 7am”), an event (“when a form is submitted”), a message (“when this arrives in the inbox”), or you typing a prompt. The trigger is the moment the tool wakes up and starts doing something.

Steps

What happens next. In a rule, the steps are fixed. In a workflow, they're predefined but more complex. In a script, they're custom code. In an LLM, they're a single generation. In an agent, they're chosen on the fly. The steps are where the tool earns its keep.

Output

What the tool leaves behind. A file. A message. A summary. An action taken. A row added to a spreadsheet. The output is the thing the rest of your work picks up from.

This frame is useful because it strips the marketing off. “Agentic AI” sounds mystical. “Something triggers it, it does some steps, and it produces an output” doesn't. When somebody tells you about a tool, you can ask the same three questions every time: what triggers it, what are the steps, and what's the output? If they can't answer crisply, you've learned something useful before you've spent a penny.

> If you can describe the trigger and the steps, you can spot the tool.

Section 3: The four tools

Four broad categories. Each has its own shape, its own strengths, and its own ways of failing. You'll probably have access to one or two of these at work. You might have access to all of them. Either way, knowing what each one is for makes the decisions easier.

Rules & Workflows: the reliable one

Rules and workflows are the oldest, dullest, and most underrated tool in the toolkit. They predefine a sequence of steps that happen the same way every time. An email rule that files messages into the right folder. A workflow that sends a reminder three days before a deadline. A formula in a spreadsheet that updates a total whenever you type a number.

Where it fits

Routing, filing, notifications, copy-paste between systems, anything where the steps don't change. If you could write down the rules clearly enough for a new starter to follow, a workflow can probably follow them too.

Where it doesn't

Judgement, interpretation, anything where the answer depends on reading something fuzzy. A workflow can't decide whether an email is angry or polite. It can only check whether the sender is on a list.

Watch-outs

- A workflow only knows about the cases you designed it for. The first time a different kind of input arrives, you'll discover what “catch-all” was missing.
- Workflow platforms vary. Power Automate, Zapier, n8n, Make all do similar jobs but feel different. Some are visual and friendly. Others assume comfort with code. Pick one that fits your skill level, not just the marketing.
- Workflows are only as reliable as the systems they connect to. If the source system changes a field name, your workflow may fail silently. Build in a way of knowing when it breaks.

> If a temp could follow the instructions, this is your tool.

Scripts: the custom-built one

A script is custom code that does a specific thing. Pulling data out of a CSV, renaming a thousand files, transforming one report format into another. Scripts are powerful because they can do almost anything. They're also fragile, because someone wrote them and someone has to look after them.

Where it fits

Bespoke jobs that the off-the-shelf tools can't quite handle. Data wrangling. Bulk operations. One-off transformations that would take hours by hand.

Where it doesn't

Anything you need a non-technical colleague to maintain. Anything where the requirements keep changing. Anything where you'd rather not be the only person who understands how it works.

Watch-outs

- Maintenance debt. A script written in a hurry today is a problem six months from now when the person who wrote it has moved on and the requirements have changed.
- Visibility. Scripts often run quietly. Make sure someone knows what they do, where they live, and how to tell when they've failed.
- Version control and review. A script that operates on live data is doing things in the real world. It deserves the same scrutiny as any other code: stored somewhere shared, reviewed by another pair of eyes, and tested before it runs against anything important.

> Powerful, but it owes its life to whoever wrote it.

LLMs: the fluent one

Large language models are the tool everyone's talking about online. You type a prompt, the model generates a response. It can summarise a document, draft an email, explain a concept, brainstorm options, reformat a piece of writing, and a hundred other things. It's brilliant. It's also confident about things it doesn't actually know.

Where it fits

Drafting, summarising, reformatting, brainstorming, explaining, translating, and extracting structure from messy text. Anything where the answer is fluent language and the cost of a small error is low.

Where it doesn't

Facts you can't verify. Numbers. Current events. Legal or clinical questions where being wrong has consequences. Anything where precision matters more than fluency.

Watch-outs

- Confident bluffing. The model can produce a paragraph that sounds authoritative and is also wrong. Fluent doesn't mean correct. Verify everything that matters.
- Not all LLMs are the same. Some models are tuned for code, some for long-form writing, some for concise structured output. The one your organisation has rolled out may not be the best fit for the task in front of you. Knowing the differences helps you ask better questions when the answer doesn't land.
- Where the data goes. When you paste text into an LLM, that text leaves your local environment. Consumer AI tools and enterprise AI tools treat that data very differently. Know which model your organisation has approved for what kinds of information, and never paste sensitive data into a tool that hasn't been sanctioned for it.
- The reviewer is you. Whatever the model produces, your name goes on what you send. Read it. Adjust it. Question it. The model is a junior assistant, not a final author.

> Brilliant junior assistant. Verify everything.

Agents: the autonomous one

An agent is an LLM with tools and the ability to decide what to do next. Where a workflow follows fixed steps, an agent looks at the situation, picks an action, sees what happened, and picks the next

one. It's the newest tool, the one most likely to be misrepresented in marketing, and the one that most rewards a healthy scepticism.

Where it fits

Tasks where the steps can't be fully scripted in advance. Researching a topic, triaging an incoming request, investigating an anomaly. Anything where the next action depends on what you find.

Where it doesn't

Anything you need guaranteed. Anything where mistakes compound across multiple steps. Anything you couldn't afford to have go subtly wrong without you noticing.

Watch-outs

- Probabilistic decisions stacked across steps. Each individual decision the agent makes might be 90% right. Across five steps, the chance of the whole thing being right drops considerably. Agents need supervision, especially early on.
- Visibility matters. You should be able to see what the agent did, in what order, and why. If a tool can't explain itself, you can't trust it for anything important.
- Marketing vs reality. Many things called “agents” today are simpler than the word suggests. Some are LLMs with a single tool. Some are workflows with a chatbot front-end. Ask what it actually does. Ask what triggers it and what its steps are. The same three questions as everything else.
- You may not have one yet. That's fine. The first time you actually need autonomous decision-making is when the case for an agent becomes real. Until then, the simpler tools are usually doing the work.

> Think of it as an LLM let off the leash. Useful, but supervise.

Section 4: Choosing well within a category

Picking the right kind of tool is most of the battle. But within each category, the specific tool you reach for still matters, and the data flowing through it matters even more. This section is about the things that the deck doesn't have room for.

Tools within a category aren't identical

“AI” doesn't tell you which AI. “Workflow platform” doesn't tell you which one. The variation within a category is often as large as the variation between categories.

Among LLMs, some models are stronger at writing code, some at long-form essays, some at structured output, some at concise summaries. The same prompt to two different models can produce noticeably different answers. If the tool your organisation has rolled out doesn't seem to be giving you what you need, the problem isn't always your prompt. Sometimes it's the model.

Among workflow platforms, the differences are more practical. Power Automate sits comfortably inside Microsoft estates and integrates with everything you'd expect. Zapier and Make are friendly visual builders aimed at non-developers. n8n is more flexible and more technical. Which one you reach for depends partly on the task and partly on how much complexity you're willing to take on.

None of this means you need to become a tool reviewer. It means that when something isn't working, “this specific tool isn't right for this specific job” is a legitimate answer, and worth raising rather than blaming yourself.

Where does the data go?

Every time you use one of these tools, information flows. You paste something into a chat window; that text leaves your computer and goes to a server somewhere. A workflow connects two systems; data passes between them, and possibly through a third-party service in the middle.

The places that data goes determine what kinds of information are safe to use the tool for. The same prompt that's fine to send to your organisation's sanctioned enterprise AI might be a serious breach if sent to a free consumer chatbot. The same workflow that's fine for an internal newsletter could be a problem if it's quietly routing customer data through an external service.

You don't need to become a privacy lawyer. You do need to know two things: what kinds of data your organisation considers sensitive (personal information, financial data, anything covered by contracts or NDAs), and which tools have been approved for which kinds. If you don't know either of those, that's a five-minute conversation with someone in your AI, automation, or information governance team.

What guardrails are (and why they're not the enemy)

Guardrails are the rules, policies, and technical controls that decide what kinds of data can go where, what kinds of actions a tool is allowed to take, and how those uses get logged and reviewed. In practice they look like approved tool lists, sanctioned models, data classification policies, approval workflows for new use cases, and audit trails.

It's tempting to see guardrails as friction. They're usually doing the opposite. Without them, every use of an AI tool is a judgement call, and the cost of getting it wrong falls on the individual. With them, most uses are safe by default, and you only have to think when something's unusual. They're what lets you use these tools confidently, rather than worrying every time you reach for one.

Most organisations have at least some guardrails already, even if nobody has told you what they are. Finding out is one of the better-value conversations you can have. “What's the policy for using AI on customer data?” “Which AI tools have been approved for what?” “Who do I ask before I build a workflow that touches the CRM?” Three questions, a clearer picture, and you've stopped guessing.

Always verify

Whatever tool you use, the work that goes out has your name on it. That's the principle that sits underneath everything in this section.

In practical terms: nothing produced by an LLM should be sent, submitted, published, or acted on without a human reading it and being prepared to defend it. The model doesn't know what it doesn't know. It produces plausible answers, not necessarily correct ones, and the difference between those is invisible from the inside.

This isn't paranoia. It's professional good practice, and it's the same standard you'd apply to anything produced by anyone junior. If a colleague drafted a reply for you to send under your name, you'd read it before clicking send. The same applies to anything a tool produces.

A short verification check before anything goes out:

1. Does it actually answer what I asked?
2. Are the facts, names, numbers, and references real and correct?
3. Would I sign my name to this?

If the answer to any of those is unclear, the work isn't ready.

> Tools differ. Data flows. The reviewer is you.

Section 5: Real work is usually a combination of pieces

Once you can recognise the four tools, the next thing to notice is that most interesting tasks don't fit into just one. They're combinations. A trigger here, a workflow there, an LLM in the middle, sometimes an agent doing the deciding.

Think of it as a puzzle. Each tool is a piece with its own shape. The trick isn't picking a single piece; it's recognising which pieces a particular task needs and how they fit together.

Three worked examples, deliberately unglamorous.

The inbox digest

Maya runs a small team and used to spend the first hour of every Monday morning catching up on what everyone had been doing the previous week. Reading through the team's emails, picking out the things that mattered, summarising for her one-to-ones. An hour a week, four hours a month.

What it became:

- Trigger: Monday, 7am.
- Workflow: filter the team's emails from the previous week against a set of criteria (anything mentioning a customer name, anything flagged as decisions, anything with attachments).
- LLM: summarise each into one line.
- Workflow: post the digest to the team channel.

Three pieces. None of them clever in isolation. Together, they do a job that nobody has to remember to do, every Monday at seven, before anybody's in the office.

The meeting prep

David is an account manager who often has back-to-back meetings with new prospects. He'd been doing the research himself: company website, recent news, LinkedIn, anything in the CRM. Twenty minutes per meeting. On busy days, two hours of prep before the day started.

This one was different because the answer depended on what the research turned up. A clean prospect gets a different treatment from one where something needs flagging. That's where an agent earns its place.

- Trigger: a meeting with a new prospect added to the calendar.
- Agent: reads the brief, researches the company, checks recent news, looks at what's in the CRM, decides what's relevant and whether anything needs raising.
- Workflow (if nothing to flag): post the pre-read to the team channel.
- Workflow (if something to raise): send it to the meeting lead for review first.

Two paths out of the agent depending on what it finds. That's the shape of work where an agent is actually the right tool: when the next step depends on what the research turns up. If every prospect were treated identically, a simpler combination would have done the job.

> Agents earn their place when the next step depends on what they find.

The brief no one writes

Aisha got a request at 8pm from her director: a one-pager summary of a forty-page strategy document for a 9am board meeting the next morning. No regular cadence. Not a recurring task. Just an awkward evening and a long document.

The shape:

- Trigger: Aisha, at 8pm, with the document in her hand.
- LLM: a summary in the format she specified.

That's it. No workflow, because this isn't repeated work. No agent, because the steps aren't ambiguous. One piece. Half an hour later she had a draft she could refine and submit.

Three examples, three different shapes. The first has three pieces chained. The second has an agent making a decision and routing the result. The third is a single piece because nothing more was needed. The skill isn't picking the most impressive combination; it's matching the combination to the task.

> Small puzzles add up to bigger pictures.

Section 6: Not every task warrants a tool

Before reaching for any of the four tools, there's one question that has to come first: Does this task need to happen at all?

Start Here covered this as the bottom rung of the ladder: streamlining. Stop making the report nobody bothers to read. Drop the approval step that never gets rejected. Cancel the meeting that could have been a message. No technology, just noticing something wasteful and stopping.

It's the rung most worth checking first. The cleverest workflow in the world can't beat deleting the step. The most sophisticated agent doing the wrong job is still doing the wrong job.

So before you build, ask:

- Does this task need to happen at all?
- Does it need to happen at this frequency?
- Does it need to be done by this many people?
- Does it need to produce this much output?

If the answer to any of those is no, you've found a streamline before you've found an automation. That's usually the cheapest, fastest, lowest-risk improvement in the building.

> Streamline first. Automate what's left.

Section 7: Ask the people who know the toolkit

Not every tool in this playbook is something you can pick up yourself. You probably can't spin up an agent or stand up a workflow platform on your own. Your organisation may already have the tools, or may be able to provide access. Your AI and automation team is the one to ask.

They're not there to gatekeep. They're there to tell you what's possible, what already exists, and what would take real work. Most adoption teams spend half their week wishing more people came to them with practical questions. “Is there a way to do this?” is the kind of conversation they want to have.

A five-minute conversation can save you a fortnight of trying to build the wrong thing. It might also save you from quietly recreating something that already exists. Most large organisations have more internal tooling than anybody realises.

What to bring to that conversation:

- The task. What you're trying to do, in plain English, without specifying a tool.
- How often it happens, and how long it currently takes.
- Where the data comes from and where it needs to go.
- Any constraints you already know about (“this touches customer data”, “it has to integrate with this system for this step”).

That's enough for someone who knows the landscape to either point you at the right tool, tell you something already exists, or honestly say “I don't know, but I'll find out.” Any of those answers is worth more than a fortnight of guesswork.

> You don't have to know the whole toolkit. They do.

Section 8: Start with one piece

Here's the permission slip again, in case Start Here didn't make it stick: you do not have to assemble the whole puzzle. You do not need a strategy, a roadmap, or a transformation programme. You need one annoying task and one small experiment.

The one-piece-this-week approach:

4. Pick the most annoying repetitive task on your desk this week. Not the biggest problem you have, the most annoying small one.
5. Work out the shape. What's the trigger? What are the steps? What's the output?
6. Pick one piece. Maybe it's a workflow. Maybe it's an LLM. Maybe it's just a tweak to how you're doing it. One piece.
7. Try it. See what happens.
8. If it works, you've got twenty minutes back. If it doesn't, you've lost nothing and learned something.

That's the whole method. The reason it works is that small experiments succeed quietly. They don't need a sign-off. They don't need a business case. They don't require explaining yourself to anyone. They just need ten minutes and a willingness to try.

The bigger combinations from Section 5 are what this builds up to. They're not where you start. Maya's inbox digest started life as one piece (a manual filter rule). The LLM and the team channel post came later, after the rule had been useful enough to be worth improving.

Start with one piece. The rest follows on its own.

> One piece placed beats a perfect plan never built.

Glossary

A short reference for the terms used in this playbook. Definitions are practical, not exhaustive.

Agent

A tool, usually built on an LLM, that can take a high-level instruction, decide what to do next, and use other tools to do it. Distinguished from a workflow by the fact that its steps aren't fully predefined. Distinguished from an LLM by its ability to take actions, not just produce text.

Consumer AI vs Enterprise AI

Consumer AI tools (the free or personal-subscription versions of ChatGPT, Gemini, Claude and similar) treat your data according to one set of rules. Enterprise versions of the same tools, paid for by your organisation, treat it according to another. The differences include where the data is stored, whether it's used for training, and who can see it. Always know which kind you're using.

Guardrails

The combination of policies, technical controls, and processes that decide what kinds of tasks a tool can be used for, what data it can access, and how its use is monitored. Not a brake; a safety net.

LLM (Large Language Model)

A model trained on large amounts of text that can generate fluent language in response to a prompt. Good at drafting, summarising, reformatting, brainstorming. Less good at facts and precision. ChatGPT, Claude, Gemini, and Copilot are all powered by LLMs.

Prompt

The instruction or question you give an LLM. Better prompts get better responses; the work of writing a clear prompt is most of the skill of using an LLM well.

Rule-based automation

Simple automation based on “if this, then that” logic. Email rules are a familiar example. Reliable, predictable, narrow in scope.

Script

Custom code that performs a specific task. More flexible than a rule or workflow but requires someone with technical skills to write and maintain.

Trigger

The event or condition that starts a process. Could be a time, an event, a message, a file appearing, or a person taking an action.

Workflow

A predefined sequence of steps that happen automatically when triggered. More complex than a rule, less flexible than a script. Built using platforms like Power Automate, Zapier, n8n, and Make.

Workflow orchestration

A more sophisticated form of workflow that coordinates between multiple systems or services. The same underlying idea as a workflow; the term tends to be used when the complexity gets serious.

Where to go next

This playbook gave you four tools, a frame for recognising them, and the watch-outs that matter. The next step isn't to memorise it. It's to look at your own working week and notice which tasks now have a shape you can describe.

Pick the most annoying small task you did this week. Work out its trigger, its steps, its output. Match one tool to one step. See what happens. If it works, do another. If it doesn't, you've still learned something.

More playbooks are on the way. Build & Develop will go deeper into the practical side: actually building workflows and prompts that hold up over time. Governance & Safe Use will cover the data, privacy, and policy side in more depth than this playbook could. Project Delivery will look at what changes when adoption stops being one-person-trying-something and becomes a deliberate piece of work across a team.

None of them is a prerequisite for the others. Read whichever one fits the question you're sitting with.

If you have questions, suggestions, or examples from your own work that might help someone else, get in touch. The playbooks are unbranded by design. Drop your logo on them, share them inside your organisation, send them to a colleague who'd find them useful. The more they're used, the more useful they get.

Good luck. You've got this.