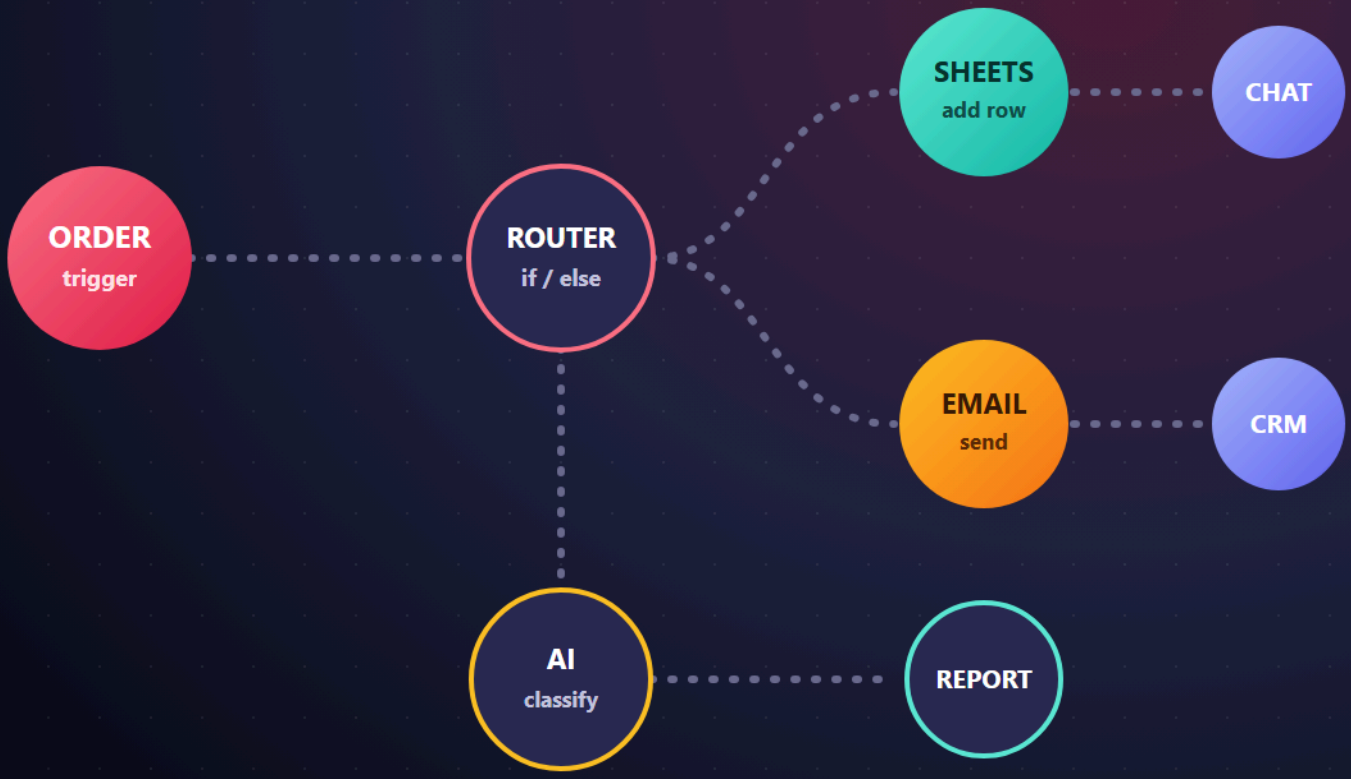



```
{{formatDate(1.created_at; "YYYY-MM-DD")}}
```

A PRACTICAL NO-CODE GUIDE

Make.com Automation for Small Business

Automate Sales, Operations, and Customer Work — Without Code

From your first scenario to AI-assisted workflows you can trust

SCENARIOS · WEBHOOKS · ROUTERS · AI · REPORTING

THENEXBOOKS

Make.com Automation for Small Business

A Practical Guide to Automating Sales, Operations, and Customer Work Without Code
From Your First Scenario to AI-Assisted Workflows You Can Trust

First Edition

THENEXBOOKS

Make.com Automation for Small Business
A Practical Guide to Automating Sales, Operations, and Customer Work Without Code

First Edition, 2026

Copyright © 2026 thenexbooks (thenexbooks.gumroad.com). All rights reserved.

No part of this publication may be reproduced, distributed, or transmitted in any form or by any means without the prior written permission of the publisher, except for brief quotations in reviews and certain other non-commercial uses permitted by copyright law. Scenario designs and examples may be used freely in your own business.

The information in this book is provided "as is" without warranty of any kind. While every precaution has been taken in its preparation, the publisher assumes no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein. Make and the apps it connects change frequently: interfaces, module names, plan features, limits, and pricing may differ from what is described here. Always check the current documentation and your plan. Nothing in this book is legal, tax, or financial advice.

Make is a trademark of its owner. Shopify, Stripe, Google, Gmail, Google Sheets, Slack, HubSpot, Airtable, Mailchimp, Calendly, Typeform, QuickBooks, Xero, Trello, Asana, and other product names are trademarks of their respective owners. Their use in this book is for identification only and does not imply endorsement or affiliation. This book is independent and is not affiliated with or endorsed by Make or Celonis.

Larkspur Home, the running example in this book, and its team are fictional. Any resemblance to real businesses or people is coincidental.

Preface

Small businesses run on repetition. An order comes in and someone copies it into a spreadsheet. A customer fills in a form and someone types their details into the CRM. A booking is made and someone sends a reminder the day before. A payment arrives and someone creates an invoice. Every Friday, someone collects numbers from five different tools to see how the week went. None of these tasks is hard, but together they eat hours and invite mistakes.

Make lets you connect the tools you already use and have that work happen automatically. You build *scenarios* visually: a trigger starts the scenario, modules move and transform data, filters and routers make decisions, and error handlers keep things running when something goes wrong. You do not need to be a programmer. You do need to think clearly about your processes, understand how data flows, and build automations that are reliable, secure, and easy for your team to maintain.

This book shows you how. We start with finding the right things to automate and building your first scenario, then learn Make's building blocks properly. From there we automate real small-business processes — leads, orders, support, marketing, bookings, invoices, inventory, documents, approvals, and reporting — and add AI where it genuinely helps. Finally, we cover error handling, design patterns, cost control, security, and how to run automations in production without surprises.

Who This Book Is For

- **Small-business owners and managers** who want their team to spend less time on repetitive admin.
- **Operations, marketing, and support staff** who are "good with tools" and have been asked to automate things.
- **Freelancers and consultants** who build automations for clients and want a solid, professional approach.
- **Beginners to no-code automation** — no programming experience is needed; technical topics are explained from the ground up.

What You Will Learn

- How to find, prioritize, and plan automations that are worth building.
- How Make works: scenarios, bundles, mapping, triggers, webhooks, scheduling, filters, routers, and functions.
- How to handle lists, store data between runs, and connect any app with the HTTP module and JSON.
- Complete automations for sales, customers, operations, and finance, built step by step.
- How to add AI safely: classifying, summarizing, drafting, and when an AI agent is (and is not) the right tool.

- How to make automations reliable, affordable, secure, documented, and maintainable.

How This Book Is Organized

Part I — Automation Foundations (Chapters 1–5) covers what to automate, getting started, your first scenario, how data flows, and how scenarios start. **Part II — Core Building Blocks** (Chapters 6–10) teaches logic, functions, lists, stored data, and APIs. **Part III — Automations for Sales and Customers** (Chapters 11–15) and **Part IV — Automations for Operations and Finance** (Chapters 16–20) build complete business automations. **Part V — AI and Advanced Techniques** (Chapters 21–25) adds AI, error handling, design patterns, and cost control. **Part VI — Running Automations in Production** (Chapters 26–28) covers security, maintenance, and a full case study. The appendices provide a cheat sheet, templates and a function reference, a launch checklist, and a glossary.

Each chapter ends with a **Key Takeaways** box for quick review.

The Running Example: Larkspur Home

Throughout the book we automate **Larkspur Home**, a fictional home-decor boutique with a shop, an online store, paid design consultations, and monthly DIY workshops. Nora (owner), Theo (operations), Mia (marketing), and Sam (bookkeeping) use common tools — Shopify, Stripe, Google Workspace, Slack, a CRM, an email-marketing app, and Calendly. Their problems are ones most small businesses share, so you can adapt every scenario to your own.

Versions and Conventions

Make evolves quickly. We focus on concepts that stay stable and describe features by what they do; where a feature is new, varies by plan, or may have changed, we say so. We never quote prices or plan limits — check your own plan. Usage is described in operations (counted as credits on recent plans). All figures for Larkspur Home are hypothetical.

> **TIP**
> A recommended practice or a technique that makes your work easier.

> **NOTE**
> Background information, version details, or a useful aside.

> **WARNING**
> A mistake that can lose data, annoy customers, or cost money.

Contents

Preface

PART I — AUTOMATION FOUNDATIONS

1 Why Automation Matters for Small Businesses: Finding What to Automate

2 Getting Started with Make: Accounts, Organizations, Teams, and Plans

3 Your First Scenario: Triggers, Modules, and Running It

4 How Make Thinks: Bundles, Operations, Data Flow, and Mapping

5 Starting Scenarios: Connections, Webhooks, Polling, and Scheduling

PART II — CORE BUILDING BLOCKS

6 Filters and Routers: Adding Decisions to Your Scenarios

7 Working with Data: Functions for Text, Numbers, Dates, and Arrays

8 Iterators, Aggregators, and Handling Lists

9 Data Stores, Data Structures, and Variables

10 The HTTP Module, APIs, and JSON: Connecting Almost Anything

PART III — AUTOMATIONS FOR SALES AND CUSTOMERS

11 Lead Capture: From Forms to CRM to Follow-Up

12 E-commerce Order Automation: Orders, Customers, and Fulfillment

13 Customer Support and Communication: Email, Chat, and Ticket Routing

14 Email Marketing and Customer Lists: Syncing, Segmenting, and Staying Compliant

15 Appointments and Bookings: Calendars, Reminders, and Workshop Sign-Ups

PART IV — AUTOMATIONS FOR OPERATIONS AND FINANCE

16 Invoicing and Payments: Stripe, Accounting Apps, and Payment Follow-Ups

17 Inventory, Suppliers, and Purchase Orders

18 Documents and Files: Generating PDFs, Contracts, and Organizing Drive

19 Team Productivity: Tasks, Approvals, and Internal Notifications

20 Reporting and Dashboards: Pulling KPIs from Many Tools into One Place

PART V — AI AND ADVANCED TECHNIQUES

21 Adding AI to Scenarios: Classifying, Summarizing, and Drafting with LLM Modules

22 AI Agents and Assistants in Make: What They Are and When to Use Them

23 Error Handling and Reliability: Error Handlers, Retries, and Incomplete Executions

24 Scenario Design Patterns: Modular Scenarios, Webhook Responses, and Avoiding Duplicates

25 Usage, Performance, and Cost Control

PART VI — RUNNING AUTOMATIONS IN PRODUCTION

26 Security, Privacy, and Compliance for Small-Business Automations

27 Documenting, Monitoring, and Maintaining Your Automations

28 Case Study: Automating Larkspur Home End to End

APPENDICES

A Make Cheat Sheet

B Scenario Templates and Function Reference

C Automation Launch Checklist

D Glossary

PART I

Automation Foundations

What to automate and why, getting set up in Make, building and running your first scenario, understanding how data flows through it, and the different ways a scenario can start.

1 · Finding What to Automate
2 · Getting Started with Make
3 · Your First Scenario
4 · How Make Thinks
5 · Starting Scenarios

CHAPTER 1 · PART I — AUTOMATION FOUNDATIONS

Why Automation Matters for Small Businesses: Finding What to Automate

Before you build a single scenario, you need to know which work is worth handing to a machine. This chapter explains what automation really is, where small teams quietly lose their hours, and gives you a simple, repeatable method for choosing your first automations with confidence.

What Automation Is (and What It Isn't)

In this book, **automation** means getting software to carry out a repeatable piece of work for you, following rules you define, whenever a specific event happens or at a specific time. A customer places an order, and a row appears in your spreadsheet. A form is submitted, and the enquiry lands in your CRM with a friendly reply sent within a minute. Every Monday at 8:00, last week's sales figures are pulled together and posted to your team chat. None of these tasks is difficult. They are simply repetitive, and repetition is exactly what software does well.

Make (formerly called Integromat) is a visual automation platform. You build **scenarios**: chains of **modules**, each of which does one job in one app — watch for new orders in Shopify, add a row to Google Sheets, send a message in Slack. You connect those modules on a canvas, tell each one which pieces of data to use, and switch the scenario on. From then on, Make runs it for you. You do not need to write code, although later chapters show you how to reach further with JSON, webhooks, and direct API requests when you are ready.

It helps to be equally clear about what automation is *not*:

- **It is not a strategy.** Automating a confused process gives you a faster confused process. If nobody agrees on how a refund should be handled, a scenario will not settle the argument.
- **It is not "set and forget".** Apps change their fields, passwords expire, and business rules evolve. Every automation needs an owner and an occasional check, which is why Part VI of this book is devoted to running automations in production.
- **It is not a replacement for judgment.** Automation is excellent at moving data and following clear rules. It is poor at deciding whether an upset customer deserves a phone call. Even the AI features covered in Part V work best when a person reviews anything sensitive.
- **It is not free.** Your time spent building and maintaining, plus the subscription for the platform and any connected apps, is an investment. The method in this chapter helps you check that the investment pays back.

A useful mental model is to think of automation as a very reliable, very literal assistant. It never forgets, never gets bored, and never mistypes a number. But it does exactly what you told it to do, even when that turns out to be the wrong thing. Most of the skill in automation lies in telling it precisely what you mean.

Three kinds of automation you will build

Almost every scenario in this book falls into one of three broad patterns. Recognizing them early makes it easier to spot opportunities in your own business.

Pattern	What starts it	Larkspur example
Event-driven	Something happens in an app	New Shopify order is logged
Scheduled	A time or interval	Monday KPI report to Slack
On demand	A person clicks or calls it	Generate a workshop roster

Event-driven automations react to the outside world, so they keep your tools in sync as work arrives. Scheduled automations collect, summarize, and remind. On-demand automations package a multi-step chore into a single button press. You will meet all three in the next few chapters, and Chapter 5 examines how each kind of start works under the hood.

Where Small Businesses Lose Time

Large companies have operations departments whose job is to design processes. Small businesses rarely do. Work gets done by whoever is available, using whatever tool was handy when the need first appeared. Over a few years this produces a patchwork: the online store holds some customer data, the email inbox holds more, a spreadsheet holds the rest, and people move information between them by hand. None of it feels like a problem on any single day, which is exactly why it is so costly over a year.

When you look closely, the lost time clusters into four familiar categories.

1. Repetitive data entry

Typing the same information into a system that could have received it automatically. Order details re-entered into a tracking sheet. Workshop attendees added to a list one at a time. Supplier invoices keyed into the accounting app. Each entry takes only a minute or two, but the entries arrive every day, and each one is an opportunity for a typo.

2. Copy-paste between tools

A close cousin of data entry, but with a twist: the information already exists in digital form, and a person is acting as the bridge between two apps that do not talk to each other. Copying an enquiry from an email into the CRM. Pasting a customer's address from Shopify into a shipping tool. Moving a

booking from Calendly into the team calendar with extra notes. Copy-paste work is the single richest source of automation opportunities because the data is already structured — it just needs moving.

3. Reminders and follow-ups

Many tasks are easy to do but easy to forget: reminding a client about tomorrow's consultation, chasing an unpaid invoice after seven days, asking a customer for a review a week after delivery, telling the team that stock of a best-seller is running low. People try to remember these with sticky notes, calendar entries, and good intentions. When they slip, the business loses revenue or goodwill, and nobody notices until later.

4. Reports and status updates

Someone opens four tools every Monday morning, copies numbers into a spreadsheet, and writes a summary for the team. The report is useful; the assembly is not. Because it is tedious, it often gets skipped in busy weeks — precisely the weeks when the numbers matter most.

> **NOTE**
>
> Notice that none of these categories describes the core work of the business — designing rooms, choosing products, teaching workshops, talking to customers. Automation's promise for a small business is not to replace that work, but to protect the time available for it.

Meet Larkspur Home

Throughout this book we follow a single fictional business so that every example connects to the ones before it. **Larkspur Home** is a home-decor boutique with one physical shop, an online store, paid interior-design consultations, and monthly DIY workshops such as "Macramé Wall Hanging". It is small enough that everyone wears several hats, and busy enough that manual work has started to pile up. All names, figures, and data about Larkspur in this book are invented for teaching purposes.

The team

Person	Role	Main pain point
Nora	Owner	No single view of the week
Theo	Operations and fulfillment	Retyping orders and stock
Mia	Marketing and social	Leads scattered across inboxes
Sam	Part-time bookkeeper	Matching payments to invoices
Two assistants	Shop floor	Workshop sign-up questions

The tools

Larkspur uses the kind of widely available apps you probably use too. The exact choices matter less than the pattern: several good tools, none of which talk to each other without help.

Job	Tool
Online store	Shopify
Consultation and workshop payments	Stripe
Invoices and accounting	QuickBooks Online or Xero
Email, files, calendar, docs	Google Workspace
Team chat	Slack
Customer records	HubSpot free CRM or Airtable
Email marketing	Mailchimp
Consultation bookings	Calendly
Enquiry forms	Typeform or Google Forms
Tasks	Trello or Asana

At the center of Larkspur's operations sits a Google Sheets spreadsheet called **Larkspur Ops**, with tabs named *Orders*, *Leads*, *Inventory*, *Workshops*, and *KPIs*. Many businesses have a spreadsheet like this: part database, part to-do list, part report. It will be the landing place for many of our early scenarios because it is easy to inspect — you can see exactly what an automation wrote and fix it by hand if needed.

Larkspur Ops — spreadsheet layout

```
Larkspur Ops (Google Sheets)
├── Orders      Order no. | Date | Customer | Email | Items | Total | Status
├── Leads       Received | Name | Email | Phone | Interest | Source | Owner
├── Inventory   SKU | Product | On hand | Reorder point | Supplier
├── Workshops   Date | Workshop | Attendee | Email | Paid | Reminder sent
└── KPIs        Week | Orders | Revenue | New leads | Bookings | Notes
```

A typical Larkspur week, before automation

Consider what happens when a customer buys a "Linen Throw – Sage" (SKU LH-TH-014) from the online store. Shopify records the order and emails the customer a confirmation — that part is already automatic. Then the manual work starts. Theo copies the order into the *Orders* tab so the team can track fulfillment. He checks the *Inventory* tab and lowers the on-hand count. If stock looks low, he messages Nora in Slack. On Friday, Sam cross-checks the week's orders against payouts. On Monday, Nora opens Shopify, Stripe, and the spreadsheet to write a short summary for the team.

Meanwhile, Mia fields enquiries from the website form about consultations. Each one arrives as an email. She copies the details into the *Leads* tab, types a personal reply, and — when she remembers — adds the person to the CRM. Enquiries that arrive on Friday evening sometimes wait until Tuesday for a response.

A Method for Finding Automation Candidates

The most common mistake beginners make is to automate whatever they happen to be annoyed by today. Sometimes that works out, but often it leads to a clever scenario that saves five minutes a month while a genuinely expensive task stays manual. A short, structured review avoids this. The method below takes an hour or two for a small team and produces a ranked list you can work through for months.

Step 1: Build a task inventory

Ask each person to list the recurring tasks they do that involve moving, checking, or copying information between tools. Keep the descriptions concrete: not "admin", but "copy new web orders into the Orders tab". A shared spreadsheet works well, with one row per task. Record who does it, which tools are involved, and what triggers it — an email arriving, a time of day, a customer action.

Encourage people to include small tasks. A two-minute task done thirty times a week is often a better candidate than a two-hour task done once a quarter. Also encourage them to note tasks that are frequently *forgotten*, since those carry hidden costs that do not show up as time spent.

Step 2: Estimate frequency and minutes

For each task, estimate two numbers: how many times it happens in a typical week, and how many minutes each occurrence takes, including switching between apps. Multiply them to get **weekly minutes**. Round generously; you are looking for orders of magnitude, not precision. If a task is monthly, divide by four or so to get a weekly figure.

Time-cost formula

```
weekly minutes = occurrences per week × minutes per occurrence
yearly hours   = weekly minutes × working weeks per year ÷ 60

Example (hypothetical): 40 orders/week × 3 min = 120 min/week
                        120 × 48 weeks ÷ 60   = 96 hours/year
```

These are your own estimates, not benchmarks. The point of the arithmetic is comparison: it lets you put an enquiry-logging task next to a reporting task and see which one actually consumes more of your team's week.

Step 3: Rate error risk and customer impact

Time is only part of the story. Some tasks are quick but costly when they go wrong. Add two ratings on a simple 1–3 scale:

- **Error risk** — how likely a mistake is and how much it hurts. Retyping prices, quantities, and email addresses scores high; posting a routine reminder scores low.
- **Customer impact** — how directly the task affects what customers experience. A slow reply to a consultation enquiry scores high; updating an internal report scores low.

Keep the scale coarse. Arguing over whether a task deserves 2.5 or 3 wastes the time you are trying to save. If the team disagrees, take the higher score and move on.

Step 4: Score and rank

Combine the numbers into a single priority score. There is no official formula; the one below is deliberately simple and works well for small teams. It uses weekly minutes as the base and multiplies by a factor that grows with risk and customer impact.

Priority score (a simple house formula)

```
priority = weekly minutes × (1 + (error risk + customer impact) ÷ 4)

error risk:       1 = low   2 = medium   3 = high
customer impact:  1 = low   2 = medium   3 = high
```

With both ratings at 1, the multiplier is 1.5; with both at 3, it is 2.5. A task that is quick but risky and customer-facing can therefore outrank a slightly longer internal chore — which matches how most owners actually feel about their business.

Larkspur's inventory, scored

Here is the result of Larkspur's first review session, held on a quiet Tuesday morning with the whole team and a pot of coffee. The figures are hypothetical estimates made by the team, which is exactly how yours will look too.

Task	Per week	Min	Wk min	Risk	Cust.	Score
Log web orders in sheet	40	3	120	3	2	270
Log and reply to enquiries	15	8	120	2	3	270
Update stock counts	40	2	80	3	2	180
Match payments to invoices	10	6	60	3	1	120
Weekly KPI summary	1	60	60	2	1	105
Add workshop sign-ups	12	4	48	2	2	96
Consultation reminders	6	5	30	2	3	68

Low-stock messages	3	5	15	2	2	30

A few things jump out. Logging orders and handling enquiries tie for first place, but for different reasons: the order task is frequent and error-prone, while the enquiry task is slower and directly shapes a prospective client's first impression. The weekly KPI summary happens only once a week, yet it still ranks mid-table because each occurrence takes an hour. Low-stock messages come last on time, but notice that they depend on the stock-count task above them — if stock counts are automated, the alert becomes almost free to add.

That last observation is important. Tasks are rarely independent. When you look at your ranked list, draw arrows between tasks that share data or follow one another. Clusters of connected tasks are often best automated together as a single scenario or a small family of scenarios.

Step 5: Check feasibility

A high score tells you a task is worth automating; it does not tell you it is *easy* to automate. Before committing, run each top candidate through four quick questions:

1. **Is there a clear starting event or time?** "When a Shopify order is created" is clear. "When Theo feels the stock looks low" is not — but "when on-hand quantity falls below the reorder point" is.
2. **Is the data already digital and reasonably structured?** Form fields, order records, and calendar bookings are ideal. Handwritten notes and free-form phone calls are not, at least not without extra steps.
3. **Can the apps involved connect to Make?** Search Make's app directory for each tool. If an app has no ready-made module, it may still offer an API you can reach with the HTTP module (Chapter 10) — but that adds effort.
4. **Can the rules be written down?** If you cannot describe the decision in a few "if this, then that" sentences, the process needs clarifying before it can be automated.

Larkspur's top candidates pass comfortably. Shopify, Google Sheets, Gmail, Slack, and the form tools all have Make apps, the triggers are clear, and the rules are simple. Payment matching is trickier because it involves judgment when amounts do not line up exactly, so the team decides to automate the easy part (recording payments and creating invoices) and leave exceptions to Sam.

Step 6: Turn candidates into named scenarios

Finally, group the tasks into scenarios and give each a short, descriptive name. Good names follow the pattern *trigger → result*, which makes it obvious what a scenario does when you see it in a list months later. These are the names we use for Larkspur throughout the book:

Scenario name	Covers tasks	Chapter
Shopify order → Sheets + Slack	Orders, stock, alerts	12

Website enquiry → CRM + reply	Enquiries	11
Stripe payment → invoice	Payments	16
Weekly KPI report	KPI summary	20
Workshop sign-up flow	Workshops	15
Calendly booking → reminders	Reminders	15
Low stock alert	Stock alerts	17
AI email triage	Inbox sorting	21

The last entry, "AI email triage", was not on the scored list at all. It came up in discussion when Mia pointed out that the general inbox mixes supplier emails, customer questions, and spam. The inventory is a starting point for conversation, not a straitjacket; good ideas often surface once the team starts thinking in terms of triggers and results.

> **TIP**
>
> Keep your task inventory as a living document. Revisit it every quarter: re-estimate the numbers, cross off what you have automated, and add the new chores that have appeared as the business has grown. The ranking will change, and that is healthy.

Estimating Value Without Fooling Yourself

It is tempting to take the weekly minutes, multiply by an hourly rate, and announce a large annual saving. Be cautious. Automation rarely eliminates a task completely: someone still checks the results, handles exceptions, and maintains the scenario. A more honest estimate subtracts those costs.

A more honest value estimate (hypothetical Larkspur figures)

```
Task: Log web orders in sheet
  Manual time now              120 min/week
  Remaining manual checks       15 min/week  (spot-checking rows)
  Maintenance (averaged)        10 min/week  (fixes, updates)
  ------------------------------------------
  Net time freed                95 min/week

  Build time (one-off)          about 3 hours
  Weeks to recover build time   180 min ÷ 95 min ≈ 2 weeks
```

The payback calculation above ignores the subscription cost of Make and any connected apps, because those depend on your plan and change over time. Check your plan's pricing and allowance, and include the relevant share in your own estimate. Chapter 25 returns to usage and cost control in depth, including how the way you design a scenario affects how many operations (counted as credits on recent plans) it consumes.

Also consider benefits that do not show up as minutes: fewer typos in customer emails, faster replies to enquiries, reports that actually arrive every week, and a business that can take a holiday without everything stalling. These are real, but they are hard to measure, so describe them in words rather than inventing numbers for them.

When Not to Automate

Knowing when to leave a task manual is as valuable as knowing how to automate it. The following situations should make you pause.

The process is still changing

If Larkspur has run workshops for only two months and is still experimenting with pricing, sign-up questions, and payment timing, an automated sign-up flow would need rebuilding every few weeks. Let the process settle first. A rule of thumb: if you have changed how you do a task in the last month, wait.

The task is rare and simple

A task done a handful of times a year, taking a few minutes, will almost never repay the effort of building, testing, and maintaining a scenario. A checklist is usually the better tool.

Every case is different

Some work is inherently judgment-based: responding to a complaint, pricing a custom design project, deciding whether to extend credit to a trade customer. Automation can still help around the edges — gathering information, drafting a starting point, routing the case to the right person — but the decision itself should stay with a human.

The cost of a silent failure is severe

If a mistake would move money incorrectly, delete important records, or send the wrong message to your whole customer list, you need either strong safeguards or a human approval step. Chapters 19 and 23 show how to add approvals and error handling; until you are comfortable with those, keep high-stakes actions manual.

The real problem is elsewhere

Sometimes a painful manual task is a symptom. If Theo spends hours reconciling stock because the shop and the website sell from separate counts, the fix may be to change how inventory is managed, not to build a clever synchronization scenario on top of a broken setup.

> **WARNING**
>
> Be especially careful with automations that send messages to customers or take payment-related actions. A scenario that fires twice, or fires on test data, can email real people or create duplicate invoices within seconds. Always test with your own email address and test records first, and read Chapter 24 on avoiding duplicates before automating anything customer-facing at scale.

Make and Other Automation Tools

Make is one of several tools that connect apps without code. You may already have heard of others, and some of your apps may include their own built-in automation features. It is worth understanding the landscape so you can make an informed choice, without treating any one tool as the universal answer.

Tool	General approach
Make	Visual canvas; branching flows
Zapier	Step-by-step trigger and actions
n8n	Node-based; cloud or self-hosted
Power Automate	Microsoft 365 oriented flows
Built-in app rules	Automation inside one product

Make presents a scenario as a visual diagram on a canvas. Modules appear as circles joined by lines, and you can add branches, filters, loops over lists, and error-handling paths directly on the canvas. Many people find this helpful for understanding how data moves, especially as scenarios grow beyond a few steps. Make offers a large library of app integrations plus general-purpose tools such as the HTTP module and JSON modules.

Zapier is a widely used automation service that presents workflows as a linear list of steps beginning with a trigger. It supports a very large number of apps and also offers branching and multi-step features. People often describe it as quick to start with.

n8n is a workflow automation tool with a node-based editor. It can be used as a hosted cloud service or installed on your own server, which appeals to teams with technical staff who want more control over where data runs.

Microsoft Power Automate is Microsoft's automation service. It is a natural fit for organizations that already live in Microsoft 365, SharePoint, Teams, and related products.

Built-in automations inside your existing apps — Shopify's own workflow features, your CRM's sequences, your email tool's automations — are often the simplest option when a task stays entirely within one product. There is no point building a Make scenario to do something a single app already does well on its own.

All of these tools change frequently. Rather than looking for a winner, ask practical questions: Does it connect to your apps? Can your team follow how it presents workflows? Does its pricing model fit your volume? Where does it process your data? The method in this chapter applies to any tool.

> **NOTE**
>
> It is perfectly normal to use more than one approach. Larkspur keeps Shopify's own order-confirmation emails and Mailchimp's built-in welcome sequence, and uses Make for the work that crosses between tools. The question is always "where does this task naturally live?", not "which tool is best?".

A First Look at a Scenario

To make the ideas in this chapter concrete, here is a sketch of the top-ranked Larkspur candidate as it will eventually appear in Make. You do not need to understand every detail yet; Chapter 3 walks you through building a simpler version step by step, and Chapter 12 builds the full version.

Scenario: Shopify order → Sheets + Slack (sketch)

```
[Shopify: Watch orders]
        │   one bundle per new order
        ▼
[Google Sheets: Add a row]       → Larkspur Ops / Orders
        │
        ▼
[Slack: Create a message]        → #orders
    "New order #1047 from Priya Shah – $86.00"
```

Read the diagram from top to bottom. The first module is the **trigger**: it notices new orders. Each order becomes a package of data that Make calls a **bundle**. The bundle flows to the next module, which writes selected fields into a new row of the *Orders* tab, and then to a module that posts a short message to the team's *#orders* channel. Chapter 4 explains bundles and data flow in detail, because understanding them is the key to building scenarios that behave the way you expect.

Preparing a Task for Automation

Once you have chosen a candidate, spend fifteen minutes writing a short **process sketch** before you open Make. It saves far more time than it costs, because it forces the decisions you would otherwise make half-consciously while clicking around the editor.

Process sketch: Website enquiry → CRM + reply

```
Trigger:   New submission on the "Book a consultation" form
Input:     Name, email, phone (optional), interest, message
Steps:     1. Add or update the contact in the CRM
           2. Add a row to Larkspur Ops / Leads
           3. Send a reply from hello@larkspurhome.example
           4. Post a note in Slack #leads for Mia
```

```
Rules:     If interest = "Workshop", send workshop reply instead
Owner:     Mia (content), Theo (scenario maintenance)
Test data: Mia's own email with the name "Test Enquiry"
Done when: Reply arrives within 5 min; row and CRM record match
```

Notice what the sketch captures: the starting event, the exact data available, each step in order, the rules for decisions, who owns the scenario, how to test it safely, and what success looks like. If you cannot fill in a line, you have found a question to answer with your team before building. The "Owner" line is especially important — an automation without an owner is an automation that will quietly break one day with nobody noticing.

Larkspur's Automation Roadmap

With the review complete, Nora and the team agree on a simple three-stage roadmap. It balances quick wins with the foundations that later scenarios will depend on.

Stage	Scenarios	Goal
1. Learn	Enquiry → Sheets + Gmail	Learn Make safely
2. Core flows	Orders, enquiries, bookings	Free up daily time
3. Oversight	KPI report, stock, payments	See the business weekly

Stage 1 is a deliberately small scenario that you will build yourself in Chapter 3: a website enquiry is written to the *Leads* tab and an email notification is sent. It is low risk, uses tools almost everyone has, and teaches every basic skill. Stage 2 tackles the highest-scoring daily tasks in Parts II and III. Stage 3 adds the reporting and oversight scenarios in Part IV, and Part V layers in AI and advanced reliability techniques once the core is stable.

Before any of that, you need a Make account set up properly for a team — with the right organization, the right people invited, and a clear sense of how usage is measured. That is the subject of Chapter 2.