STRETCHLEARN · WORKBOOK

Zapier Automation for Non-Coders — Workbook

This workbook turns the course into action. Each section maps to one course module and mixes guided exercises, fill-in worksheets, and checklists so you finish with a real, documented library of working Zaps. Keep it open beside the Zapier editor and complete each item as you go rather than all at once.

How Automation Thinking Works

Train your eye to spot automatable tasks and translate them into the trigger-action model before you build anything.

Exercise: Write Five When-Then Sentences

List five recurring tasks from your week and rewrite each as a single when-then sentence. The when must be a clear digital event and the then must be a concrete action. Do not build anything yet; the goal is to see the shape of each automation.

• What digital event marks the start of this task (an email, a form, a row, a time)?

• What single concrete action should happen in response?

• Could a non-expert follow the steps without judgement calls, making it rule-based?

• Roughly how many times per week does this happen?

Worksheet: 3-Test Rule and Payoff Scorecard

Score each candidate task against the 3-Test Rule, then estimate annual time saved using minutes per run times runs per week times 52. The task with the highest score and payoff becomes your first build.

Task name

Repetitive (yes / no)

Rule-based (yes / no)

Trigger-able (yes / no)

Minutes saved per run

Runs per week

Estimated hours saved per year

Build priority (1 = first)

Checklist: Account and Editor Readiness

[ ] Created a free Zapier account using the email tied to my main apps
[ ] Located the Zaps dashboard, the Create button, and Zap History
[ ] Opened the editor and identified the step list, config panel, and Test button
[ ] Connected at least one app I already use
[ ] Confirmed I understand that tasks, not Zaps, are what I pay for

Building Your First Working Zaps

Build, test, and publish your first real Zaps and learn to diagnose failures using Zap History.

Exercise: Build and Publish Your First Zap

Build the Gmail-attachment-to-Drive Zap or an equivalent two-step Zap from your scorecard. Test each step as you build it and only publish once both steps pass. Then trigger it for real and confirm the result.

• Which trigger app and event did you choose, and what action followed?

• Which value did you map from the trigger into the action's main field?

• Did the test produce a real, visible result (a file, a row, a message)?

• How long was the delay between the trigger event and the action running?

Worksheet: Form-to-Spreadsheet Mapping Plan

Before building the form-to-sheet Zap, plan the mapping. List each form field and the exact spreadsheet column header it will write to. A clean header row is what makes the mapping hold.

Form tool and form name

Form field 1 maps to column

Form field 2 maps to column

Form field 3 maps to column

Timestamp maps to column

Destination sheet name and tab

Header row confirmed clean (yes / no)

Checklist: Debugging a Broken Zap

[ ] Opened Zap History and found the errored run
[ ] Read the raw input the failing step received
[ ] Checked whether a required field was empty
[ ] Confirmed the data type matched what the action expected
[ ] Verified the app connection had not expired
[ ] Used Replay to re-run the fixed task instead of recreating data by hand

[ ] Turned on email notifications for Zap errors

Multi-Step Logic: Filters, Paths, and Formatting

Add conditional logic, branching, and data cleanup so your Zaps act only when they should and handle messy data.

Exercise: Add a Filter to an Existing Zap

Pick one live Zap that currently runs every time and add a Filter so it continues only on records that matter. Test with sample data that should pass and sample data that should stop, then confirm both in Zap History.

• Which field does your filter test, and what condition does it use?

• What value distinguishes the records worth acting on?

• Did a should-stop record correctly show as Filtered in Zap History?

• How many tasks per month do you estimate this filter will save?

Worksheet: Path Branch Design Sheet

Design a single Zap that forks based on one field. Define each branch's name, its condition, and the unique actions it runs. Keep conditions mutually exclusive and add a catch-all.

Shared trigger and any shared steps

Path A name and condition

Path A actions

Path B name and condition

Path B actions

Catch-all path actions

Conditions are mutually exclusive (yes / no)

Checklist: Formatter Cleanup Pass

[ ] Identified one Zap where data lands in the wrong shape
[ ] Chose the right Formatter category (Text, Numbers, or Date and Time)
[ ] Inserted the Formatter step between the messy source and the destination
[ ] Mapped the cleaned output forward into the next step
[ ] Tested with a real sample and confirmed the output is correct

Exercise: Split a Full Name Into First and Last

Practise the Text Formatter by splitting a full name into separate first and last name values, then map both into a destination that needs them separately. This is the single most reused Formatter pattern.

• What separator did you use to split the name?

• Which segment index gave you the first name, and which the last?

• Where did you map each cleaned value?

• What other fields in your workflows could use this same split-or-clean approach?

Reliability, Cost, and Scaling Your System

Control task spend, document every automation, and plan your path into more advanced steps.

Worksheet: Monthly Task Budget Estimator

Estimate your monthly task usage so you can choose the right plan. For each Zap, multiply runs per month by the number of action steps, add a buffer, and compare the total to your plan's allowance.

Zap name

Runs per month

Action steps in the Zap

Tasks per month (runs times action steps)

Total tasks across all Zaps

Buffer added (20 to 30 percent)

Current plan and its task allowance

Plan covers total with room (yes / no)

Exercise: Name and Register Every Zap

Apply a consistent naming convention to all your Zaps, then record each one in your automation register. Aim for names that explain a Zap without opening it, such as Trigger to Action, Purpose.

• What naming convention will you use across every Zap?

• Which Zap was hardest to describe in one line, and why?

• Who is the named owner for each automation?

• Which sheets, folders, or forms must never be renamed because a Zap depends on them?

Checklist: Monthly Maintenance Review

[ ] Opened Zap History and scanned for errored or paused Zaps
[ ] Confirmed all app connections are still active
[ ] Checked the task meter against the plan allowance
[ ] Updated the automation register for anything that changed
[ ] Identified one new task to automate next
[ ] Confirmed every live Zap still has a named owner

Exercise: Plan One Advanced Automation

Design one ambitious automation that combines at least two course skills, such as a Filter plus a Formatter, or a Path plus an AI step. Map it on paper first, then build it one tested step at a time.

• What is the when-then sentence for this automation?

_____

• Which two or more skills will it combine, and why each?

_____

• Where will you check or override any AI-generated output?

_____

• What is the first single step you will build and test?

_____

Your Action Plan

1. List your five most repetitive weekly tasks as when-then sentences
2. Score them with the 3-Test Rule and the time-saved formula, then pick the top one
3. Create a free Zapier account and connect your two or three core apps
4. Build, test step by step, and publish your first two-step Zap
5. Add the form-to-spreadsheet Zap with a clean header row and mapped fields
6. Insert a Filter so one Zap acts only on records that matter
7. Add a Path or a Formatter step to handle branching or messy data
8. Estimate your monthly task usage and confirm your plan covers it
9. Set up your automation register and record every Zap you have built
10. Book a recurring monthly ten-minute maintenance review in your calendar