STRETCHLEARN · WORKBOOK

Zapier Workflow Automation — Workbook

This workbook turns the course into hands-on practice. Each section maps to one module and mixes exercises, fill-in worksheets, and checklists so you build real, tested Zaps as you go. Work in a free or trial Zapier account connected to apps you actually use, and by the end you will have a multi-step Zap connecting Gmail, Slack, Airtable, and your CRM, complete with filters, clean data, and error notifications.

How Zapier Thinks: Triggers, Actions, and Your First Zap

Map a real process, build a trigger-action Zap, and learn to estimate task usage before you commit.

Worksheet: Process-to-Zap blueprint

Pick one manual process you repeat. Write it as a single when-this-happens sentence, then break it into trigger, data, and actions. This blueprint becomes the documentation you keep beside the Zap.

Process in one sentence (when X happens, do Y and Z)

Trigger app and exact event (e.g. Gmail, New Labeled Email)

Polling or Instant trigger? (check the trigger label)

Data needed from the trigger (name, email, amount, date)

Action 1 — app and what it creates or updates

Action 2 — app and what it creates or updates

Action 3 — app and what it creates or updates

Records that should NOT trigger it (becomes a Filter)

Exercise: Build a Gmail-to-Slack alert Zap

In the Zap editor, build the worked example: trigger on Gmail New Labeled Email for a label like Urgent, action Slack Send Channel Message. Map the Subject and From fields into the message, test the step so a real message posts, then publish.

• Which Gmail label did you trigger on, and why that one?

• Which trigger fields did you insert into the Slack message?

• Did the test step actually post a real Slack message? Note what it said.

• What error, if any, did the first test return, and how did you fix it?

Exercise: Estimate task usage before building

For the Zap in your blueprint, calculate the monthly task cost. Count the action steps, estimate how often the trigger fires per month, and multiply. Compare the result to common plan tiers (Starter 750, Professional 2,000 tasks).

• How many action steps does the Zap have? (Filters that stop it are free.)

• How many times do you expect the trigger to fire per month?

• Action steps multiplied by trigger fires = estimated tasks. What is the number?

• Which plan tier does that fit, and would an early Filter lower it?

Checklist: First-Zap readiness

[ ] Process is written as a clear when-this-then-that sentence
[ ] Trigger app and exact event are chosen, and polling vs instant is noted
[ ] Every action step lists its app and what it creates or updates
[ ] Each step was tested with real data and the output was read
[ ] Estimated monthly tasks were calculated and matched to a plan

Controlling the Flow: Filters, Paths, and Timing

Decide which records run and when, using Filters, Paths, Delay, Schedule, and Digest to control behavior and cost.

Exercise: Add a money-saving Filter

Take a Zap with a noisy trigger and add a Filter by Zapier step immediately after the trigger. Set rules that exclude internal or test records, for example Email does not contain your company domain AND Message does not contain test. Send a test record that should be blocked and confirm the Zap stops.

• Write out your Filter rules exactly, noting which are AND and which are OR.

• Read the rules aloud as a sentence. Does the sentence match your intent?

• Send a record that should be blocked. Did the Zap stop and skip the actions?

• Roughly what percentage of records does this Filter now stop (and save in tasks)?

Worksheet: Paths routing plan

Plan a single Zap that branches with Paths instead of building near-duplicate Zaps. Define each path's rule and actions, and make sure the rules are mutually exclusive so each record belongs to exactly one path.

Trigger event the paths share

Path A name and rule (e.g. High-value: Amount > 10000)

Path A actions

Path B name and rule

Path B actions

Are the rules mutually exclusive? (no record matches two paths)

Worksheet: Timing tool selector

For each timing need in your Zaps, pick the right built-in tool. Match a pause-after-event to Delay, a recurring clock job to the Schedule trigger, and a batch-the-noise need to Digest.

Timing need described (pause, recurring, or batch)

Chosen tool (Delay / Schedule / Digest)

Delay mode if used (For / Until / After Queue) and duration

Schedule frequency if used (hourly / daily / weekly) and time

Digest release schedule if used and what gets batched

Checklist: Flow-control review

[ ] A Filter sits immediately after the trigger to stop unwanted records
[ ] Filter AND/OR logic reads correctly as a plain-English sentence
[ ] Path rules are mutually exclusive and each path is descriptively named
[ ] Delay, Schedule, or Digest is matched to the actual timing need
[ ] Digest is used where a stream of alerts would otherwise cause fatigue

Reshaping Data and Connecting Real Apps

Clean data with Formatter, prevent duplicates with search steps, and connect Gmail, Slack, Airtable, and your CRM into one Zap.

Exercise: Clean a lead with Formatter

Add Formatter steps to split a Full Name into first and last using a space, titlecase a name, and reformat a date for your CRM. Then confirm you map the Formatter outputs, not the raw trigger fields, into the next action.

• Which Formatter transformations did you use (Split, Titlecase, Date Format, etc.)?

• What did the raw value look like, and what did the cleaned output look like?

• Confirm the action step uses the Formatter output, not the raw trigger field. How did you check?

• What formatting problem would have broken the action without Formatter?

Exercise: Prevent duplicates with a search step

Replace a plain Create action with a Find or Create action (for example, Find or Create Contact in HubSpot or Pipedrive) keyed on email. Then send the same record twice and confirm only one record results.

• Which app and which unique key did you search on (email, order ID, etc.)?

• Did you use Find or Create, or a search followed by a Path? Why?

• Send the same record twice. How many records were created?

• How many tasks did the find-or-create version use per run?

Worksheet: Four-app Zap field map

Plan the multi-step Zap connecting Gmail, a CRM, Airtable, and Slack. For each step, note the source of every field it needs, so you can build it without getting lost in the dropdowns.

Trigger (Gmail label) and fields it captures

Formatter outputs (cleaned name, formatted date)

CRM step: search key and fields mapped in

Airtable step: base/table and fields mapped in

Slack step: channel and which fields go into the message

Any line items needed (e.g. multiple products per order)?

Checklist: Connected-Zap quality check

[ ] Formatter cleans names, dates, and numbers before they reach any destination
[ ] Action steps map Formatter outputs rather than raw trigger fields
[ ] A search step runs before any create, keyed on a genuinely unique value
[ ] Sending the same record twice produces only one record
[ ] Each step does one clear job and is named so the Zap reads top to bottom

Keeping Zaps Alive: Errors, Monitoring, and Scale

Find and fix failures in Zap History, set up alerts and Autoreplay, and apply habits that keep a growing fleet of Zaps efficient.

Exercise: Diagnose a failure in Zap History

Open Zap History, filter by status Error, and open a failed run (cause one deliberately by removing a required field if needed). Read the error, open the step data to see the values that arrived, fix the cause, then use Replay to confirm it succeeds.

• What did the error message say, literally?

• What did the step's incoming data reveal (a blank or malformed field)?

• What was the fix — a Filter, a Formatter, or a reconnected account?

• After Replay, did the corrected run succeed?

Checklist: Safety-net setup

[ ] Error notification emails are enabled and reach a named Zap owner
[ ] Autoreplay is on (Professional+) so transient errors retry automatically
[ ] You know where the Versions panel is and how to restore a working version
[ ] Changes are made one at a time and tested, so a bad edit is easy to roll back
[ ] Critical Zaps have a monitoring alert if a key process has not run

Worksheet: Zap fleet register

Document each Zap you build so a colleague could maintain it without you. Keep this register updated as your collection grows.

Zap name (trigger and outcome)

Folder / team it belongs to

What it does in one line

Action step count (task cost per run)

Owner who receives error alerts

Last reviewed date

Exercise: Run a monthly efficiency review

Open your Zapier dashboard task usage and Zap History. Identify your highest-task Zaps and check for any runaway trigger. Decide one change that would cut cost, such as an earlier Filter, a Digest, or fewer steps.

- Which Zap used the most tasks this month, and why?
- Did any trigger fire far more often than you expected?
- What one change (earlier Filter, Digest, fewer steps) would cut the cost?
- Are you within your plan's task tier, or do you need to optimize or upgrade?

Your Action Plan

1. Spend a week listing every manual copy-paste-between-apps task you do and how often.
2. Write each one as a when-this-then-that sentence and pick a single one to automate first.
3. Build that Zap in the editor, testing the output of every step before publishing.
4. Add a Filter immediately after the trigger to stop test, internal, and junk records for free.
5. Insert Formatter steps to clean names, dates, and numbers before any destination app.
6. Replace every Create action with a Find or Create search keyed on a unique value to prevent duplicates.
7. Branch with Paths or batch with Digest where one trigger needs different handling or quieter alerts.
8. Turn on error notification emails and Autoreplay, and assign a named owner to the Zap.
9. Fill in the Zap fleet register and estimate the Zap's monthly task usage against your plan.
10. Book a monthly review of Zap History and task usage to catch runaway or broken Zaps early.