STRETCHLEARN · WORKBOOK

Building Custom GPTs (No-Code) — Workbook

This workbook turns the course into a finished, shareable custom GPT. Work through one section per module: scope the GPT, write structured instructions, prepare and ground knowledge files, then add a safe action and launch with the right privacy settings. Fill the worksheets, run the tests in the Builder's Preview pane, and by the end you will have at least one GPT live for your team or customers, plus a reusable instruction template and a test set you keep.

Custom GPT Foundations and Setup

Decide whether a custom GPT is the right tool, learn the Builder, and scope your GPT to one job and one audience.

Worksheet: GPT Scoping Sheet

Fill every field before you open the GPT Builder. Keep it to one line each. This sheet becomes the source material for your instructions in the next module, so be specific about the audience and the single job.

Working name for the GPT

Exact audience (who opens it)

The one job, in a single sentence

What a good output looks like

Three things that are out of scope

Knowledge files it will need

Tools it needs (web search, image, data analysis) and which to turn off

Intended sharing level (private, link, workspace, public)

Exercise: Tool-Fit Decision

Confirm a custom GPT is actually the right tool before you build. Write a one-line answer to each prompt. If the honest answer to the first prompt is no, consider a plain chat, a Project, or an automation instead.

• Will this task repeat, or will someone other than you need the result? Explain in one line.

• Could a ChatGPT Project or a plain chat do this just as well? Why or why not?

• Does the task need to run on a schedule or move data between apps (a sign you want Zapier or Make

instead)?

Checklist: Builder Setup Checklist

[ ] Confirm you have a paid plan (Plus, Team, or Enterprise) that includes the GPT Builder
[ ] Open chatgpt.com, go to Explore GPTs, and click Create to reach the Builder
[ ] Use the Create tab to generate a rough first draft of name and instructions
[ ] Switch to the Configure tab and locate Instructions, Knowledge, Capabilities, and Actions
[ ] Find the live Preview pane and run one test question to confirm the loop works

Writing Instructions That Behave

Write structured instructions using the four-part pattern, then test and debug them against a small test set.

Worksheet: Four-Part Instruction Builder

Draft each section, then paste the combined result into the Instructions box in Configure. Use short imperatives, put the most important rule first, and keep the total well under 8,000 characters. Edit the auto-generated draft from the Create tab rather than trusting it as final.

Role and goal (1 to 2 sentences naming who it is and the one job)

Rules (the non-negotiables as short imperatives)

Workflow (the step-by-step of a normal request)

Refusals and edge cases (what to decline and the exact wording)

Tone (3 adjectives and a reading level)

Default output format (shape and length limit)

One example output to pattern-match

Exercise: Write Four Conversation Starters

Draft four starter buttons that show off the GPT's real strengths, phrased the way a user would actually ask. Paste them into the Conversation starters fields, then test that each leads to a strong answer in Preview.

• Write one starter for the single most common question your audience will ask.

• Write one starter for a harder or edge-case question that proves the GPT is capable.

• Write one starter that triggers a task the GPT can perform (draft, summarize, quiz, look up).

Exercise: Build and Run Your Test Set

Create a small test set and run it in the Preview pane after every meaningful edit. Change one instruction at a time so you know which change caused which effect. Use the prompts below to generate and stress your tests.

• List 8 realistic questions a real user would ask, including 2 that should be refused.

• Run each question in Preview and note for each: correct, wrong, off-tone, or wrongly answered out-of-scope.

• For the worst failure, ask the GPT in Preview: Which instruction or file did you use for that answer?

Checklist: Instruction Quality Gate

[ ] Instructions are split into Role, Rules, Workflow, and Refusals with clear labels
[ ] The single most important rule appears first and is phrased as a blunt imperative
[ ] Tone, format, and length are specified, with one example output included
[ ] Refusal wording is explicit, so the GPT declines out-of-scope questions the same way every time
[ ] No contradictory rules remain, and the full test set passes after the latest edit

Knowledge Files and Retrieval

Choose and prepare clean knowledge files, force the GPT to cite and stay grounded, and set up a maintenance routine.

Worksheet: Knowledge File Inventory

List every document you plan to upload (up to 20). For each, confirm it is the current version, text is selectable, and it covers a distinct topic. Use descriptive filenames with a date. Delete superseded drafts before uploading.

File name (descriptive, with date, e.g. returns-policy-2025.pdf)

Topic it covers (one per file where possible)

Source of truth and owner

Text selectable, not a scanned image? (yes or no)

Current version confirmed? (yes or no)

Contains sensitive data unsafe for the audience? (yes or no)

Exercise: Force Grounding and Citations

Add the grounding rules to your instructions, then deliberately test that the GPT answers only from files and refuses when a topic is not covered. If it invents an answer, strengthen the rule and re-test.

• Add a rule: Answer only from the uploaded knowledge files and always name the file or section used.

• Ask a question you know is answered in a file, and confirm the GPT quotes it and cites the source.

• Ask a question that is deliberately not in any file, and confirm it refuses instead of guessing.

Checklist: Retrieval Health Check

[ ] Large documents are split by topic with clear headings, not one giant file
[ ] The GPT cites the file or section it used for each grounded answer
[ ] Numbers and table answers are spot-checked against the source document
[ ] No overlapping or duplicate files that could make the GPT cite contradictory rules
[ ] Every uploaded file is current, and stale versions have been deleted

Worksheet: Knowledge Maintenance Plan

Define how each file stays current. Tie updates to the underlying source of truth so the GPT never quotes outdated policy. Replace files cleanly in Configure rather than uploading a second version alongside the old one.

Owner responsible for each file

Trigger that prompts an update (e.g. policy change, new price sheet)

Filename versioning convention (e.g. pricing-2025-q3.csv)

Review cadence (e.g. monthly)

Post-update step: which test questions to re-run after each swap

Actions, Sharing, and Going Live

Add a safe read-only action with no code, choose the right privacy setting, and launch with a pilot, a rollout note, and a feedback loop.

Exercise: Add a Read-Only Action

Connect one read-only outside service so a mistake cannot change or delete anything. Get a schema with no code, paste it into the Actions area in Configure, set authentication securely, and test in Preview that the GPT calls the action and returns real data.

• Identify a read-only service and get an OpenAPI schema (published file, or ask ChatGPT to draft it from the docs).

• In Configure, create a new action, paste the schema, and set Authentication to None or API Key in the secure field.

• Add an instruction telling the GPT exactly when to use the action, then test the triggering question in Preview.

Worksheet: Sharing and Privacy Decision

Decide the visibility before you publish. Match the setting to the sensitivity of the instructions and files, defaulting to the most private option that still lets your intended users reach the GPT.

Sensitivity of the knowledge files (internal, customer-safe, fully public)

Chosen visibility (Only me, Anyone with the link, workspace-only, or Public)

Who specifically needs access and how they will get the link

Confirmed no secrets in instructions (keys live only in the Action field)?

Builder identity shown (your name or workspace)?

Checklist: Pre-Launch Privacy and Safety Check

[ ] Every uploaded document is safe for the intended audience to see
[ ] No credentials or secrets appear in the instructions; they live only in the secure Action field
[ ] The action is read-only, or fully trusted if it writes, sends, or pays
[ ] You tried to make the GPT reveal its files and instructions, and tightened anything that leaked

[ ] Visibility matches sensitivity and defaults to the most private setting that still works

Exercise: Pilot and Write the Rollout Note

Share the link with two or three real pilot users before publishing widely. Watch how they actually use it, fix the obvious gaps, update the starters, then draft a short rollout note for the full audience.

• Recruit 2 to 3 pilot users who match the audience and collect the questions they actually ask.

• Draft a rollout note covering name and purpose, how to access it, what it can and cannot do, where to report problems, and who owns it.

• List the top 3 fixes from the pilot and re-run the full test set before widening access.

Your Action Plan

1. Complete the GPT Scoping Sheet: audience, one job, success, and out-of-scope, on one line each
2. Confirm tool fit, then open the Builder, draft in Create, and move to Configure for control
3. Write structured instructions using the Role, Rules, Workflow, Refusals pattern, plus tone and format
4. Add four conversation starters phrased the way real users ask
5. Build a test set of 8 to 10 questions (including refusals) and run it in the Preview pane
6. Prepare clean, single-topic knowledge files with selectable text and descriptive names, and upload them
7. Add grounding rules so the GPT answers only from files and cites sources, then test grounding deliberately
8. Add one read-only action with a no-code schema, keeping any key in the secure Authentication field
9. Choose the right visibility, run the pre-launch privacy check, and try to make the GPT leak its instructions
10. Pilot with 2 to 3 users, write a rollout note, fix the gaps, and set up a feedback and review cadence