The CustomGPT Project Guide for Instructional Designers

Updated July 3, 2026

A custom GPT is one of the highest-leverage portfolio pieces an instructional designer can build right now. Candidates who show AI-enabled work in their portfolios are consistently seeing stronger job search outcomes, and hiring teams call these projects out by name in interviews. One of my community members built a custom GPT to support clinicians, and the hiring team explicitly referenced it as the reason her work stood out. Not because it was "AI", but because it demonstrated problem-solving, judgment, and an understanding of real constraints.

This guide walks you through building one that does the same for you.

What counts as a strong project

A custom GPT is a configured version of ChatGPT with its own instructions, knowledge files, and purpose. Anyone can make one in an afternoon, which is exactly why a generic one impresses nobody. The bar is the same as any portfolio piece: it must solve a real performance problem for a real audience.

Weak: "An ID assistant that answers questions about instructional design."

Strong: "A coaching bot that lets new pharmacy techs practice insurance-rejection conversations against realistic scenarios, with feedback rubrics drawn from the pharmacy's actual call guidelines."

The difference is specificity: a named audience, a task they struggle with, and source material that grounds the bot in reality.

Step 1: Pick a problem, not a topic

Start where you would start any project: with a performance gap. Good hunting grounds:

- A task from your current or former workplace that people constantly ask questions about
- A conversation people need to practice but rarely get to (feedback, sales objections, patient interactions)
- A dense policy or process document people search instead of read
- A decision people make often and inconsistently (escalation calls, tool selection, triage)

Write one sentence before you build anything: "This GPT helps [audience] do [task] better by [what it does]." If you cannot write that sentence, you have a topic, not a problem.

Step 2: Gather your source material

The knowledge files are what separate your GPT from vanilla ChatGPT. Collect the real documents the task depends on: process guides, policy excerpts, rubrics, example dialogues, FAQs. If your problem comes from a workplace, sanitize everything: strip names, numbers, and anything proprietary, or rebuild realistic stand-ins. A portfolio piece is public; treat it that way.

No workplace source? Build the source material yourself from public references and label it as a realistic simulation. Creating credible source docs is instructional design work too, and you can say so in your write-up.

Step 3: Write the instructions

The instructions field is where your ID skills show. Structure it like a design document:

1. **Role and audience.** Who the GPT is and who it serves. "You are a practice coach for newly hired pharmacy technicians."
2. **Task.** What it does, step by step. If it runs practice scenarios, spell out the loop: present the scenario, let the user respond, give feedback against the rubric, then escalate difficulty.
3. **Constraints.** What it must not do: give medical or legal advice, invent policy, reveal the full rubric up front, break character mid-scenario.
4. **Tone.** How it talks. Encouraging but honest beats relentlessly positive.
5. **Feedback rules.** The rubric it scores against, and the format of its feedback. This is where learning science earns its keep: make feedback specific, behavior-focused, and tied to the criteria.
6. **Openers.** Write the conversation starters users will see, one per core use case.

Keep instructions in plain, direct language. Number the steps. Test every rule you write, because the model will follow some instructions more reliably than others and you only find out by trying.

Step 4: Test it like an assessment

Do not ship after two happy-path conversations. Run it through:

- The intended flow, five or more times, checking that the loop holds
- Wrong answers and half-answers, to see if feedback stays useful
- Attempts to break character or extract the rubric
- Questions outside its scope, where it should decline and redirect
- A genuine novice user, watched silently while they use it

Log what fails, revise the instructions, and retest. Keep the log. Your iteration notes become the strongest part of your portfolio write-up because they prove you evaluate your own work.

Step 5: Package it for your portfolio

The GPT alone is not the portfolio piece. The write-up is. Publish:

1. **The problem.** The audience, the performance gap, and why existing resources were not enough.
2. **Your design decisions.** Why a conversational tool fit this problem, how you structured the practice loop, how you designed the feedback.
3. **The build.** Your instruction architecture and knowledge sources, with a screenshot or excerpt.
4. **Testing and iteration.** What broke, what you changed, what you measured.
5. **A way to try it.** A public link to the GPT, plus a short screen recording for reviewers who will not click through.

Frame it in the same problem-solution-result shape as any other portfolio project. The AI is the medium; the instructional design is the story.

Interview talking points

When the project comes up, and it will, be ready to speak to:

- Why this problem suited an AI tool, and what you would not use one for
- How you kept the bot from inventing facts (constrained scope, grounded knowledge files, refusal rules)
- What your testing revealed and how the design changed
- Where a tool like this fits in a full learning ecosystem: practice and performance support, not a replacement for analysis or assessment

Those four answers demonstrate judgment, and judgment is what gets senior consideration. The designers standing out in 2026 are not the ones who mention AI; they are the ones who can show a working tool, explain every design decision behind it, and name its limits.

Have any questions? Email me at devlin@devlinpeck.com