AI Agents 101: Exercises

AI Agents 101: Exercises

The views expressed are the author's own and do not necessarily represent the views of the U.S. Census Bureau or the U.S. Department of Commerce.

These exercises let you try the concepts yourself. All you need is access to Claude, ChatGPT, or another capable chat interface.

Exercise 1: Run the Recipe Workflow

Copy this prompt into your chat interface. Watch what happens. Then try breaking it.

```
You are a recipe assistant that helps users plan meals and create grocery lists.

When the user requests a dish:

1. SEARCH for highly-rated recipes (4+ stars, reputable sources like Serious Eats, NYT Cooking, Bon Appe
reviewed AllRecipes submissions). Prefer recipes with clear ingredient lists and reasonable prep times.

2. SELECT one recipe. Briefly explain why you chose it (rating, source credibility, matches the dish re

3. EXTRACT the full ingredient list. Standardize quantities where reasonable (e.g., "1 28-
oz can crushed tomatoes" not "one large can tomatoes").

4. CHECK against user's stated dietary restrictions or allergies. If there's a conflict:
   - STOP
   - Explain the conflict clearly
   - Ask if they want a substitution or a different recipe
   - Do not proceed until they respond

5. GENERATE a grocery list organized by store section (produce, meat, dairy, pantry, etc.). Include quan

6. ASK: "Anything on this list you already have? Give me the numbers and I'll remove them."

If the user provides items they have, regenerate the list without those items.
```

```
---

User dietary restrictions: [none stated yet - ask if unclear]

User request: I want to make chicken cacciatore
```

What to notice:

- Decision points with criteria (step 2)
- Hard stop with human checkpoint (step 4)
- Iteration loop (step 6)
- Transparency requirement ("explain why you chose it")

Try breaking it:

- Request a dish with an ingredient you're allergic to
- Ask for something obscure with limited search results
- Give a vague request like "something Italian"
- Ask for two dishes at once

What happens? Where does it handle well? Where does it struggle?

Try different models:

Run the exact same prompt on different model tiers. In testing, this produced dramatically different results:

Behavior	Frontier (Sonnet 4)	Small (Haiku 3.5)
No user input → wait?	Asked what to cook	Built a React app
Tool selection	Used web search	Searched ML papers for meatballs
Followed step sequence	Correct order	Skipped/hallucinated
Stayed in role	Recipe assistant	Became a software engineer

Why this matters: Modern frontier models are reasoning agents with built-in capabilities for intent inference, tool routing, and conditional logic. Smaller/older models lack this. The same prompt, the same framework, wildly different results. The "agent" lives in the interaction between prompt design and model capability — not in either one alone.

This is why the Jobs doctrine applies: don't cling too tightly to yesterday's model assumptions. The capability floor is rising fast, and what broke on Haiku 3.5 six months ago may work on the next generation's smallest model. Design for the trajectory, not the snapshot.

Exercise 2: The Generic Skeleton

This is the template structure underneath the recipe prompt. Use it to build your own workflows.

```
You are a [ROLE] that helps users [PRIMARY GOAL].

When the user [TRIGGER CONDITION]:

1. [OBSERVE] - Gather information
   - What inputs do you need?
   - What context matters?

2. [DECIDE] - Apply criteria
   - What are the selection/filtering rules?
   - What makes a good vs. bad choice?
   - Explain your reasoning.

3. [ACT] - Execute the task
   - What are the concrete steps?
   - What format should outputs take?

4. [CHECK] - Validate before proceeding
   - What could go wrong?
   - What conflicts or edge cases require a STOP?
   - If uncertain or high-stakes: STOP and ask.

5. [ITERATE] - Refine based on feedback
   - How does the user confirm or adjust?
   - What's the exit condition?

---

Constraints: [What the agent must never do]

User context: [Relevant background, if any]

User request: [The actual task]
```

The key additions to any workflow:

- **Explicit criteria** — Don't let the agent guess what "good" means
- **Hard stops** — Define when it must pause and ask
- **Transparency** — Require reasoning, not just answers
- **Iteration** — Build in refinement, not just one-shot output

---

Exercise 3: Adapt for Your Work

Pick one of these scenarios (or invent your own) and modify the skeleton.

Option A: Document Intake Assistant

A workflow that takes an uploaded document, extracts key information, and generates a structured summary.

Think about: - What document types? What's the scope? - What counts as "key information"? - When should it stop and ask for clarification? - What format should the summary take?

Option B: Meeting Prep Assistant

A workflow that takes a meeting topic and attendee list, then generates an agenda, relevant background, and suggested questions.

Think about: - What inputs does it need? - How does it know what background is "relevant"? - When should it flag that it doesn't have enough context? - What's the deliverable format?

Option C: Research Synthesis Assistant

A workflow that takes a topic, finds relevant sources, and produces a structured brief with key findings and open questions.

Think about: - What makes a source credible for your domain? - How does it handle conflicting information? - When should it stop and ask for direction? - How long should the output be?

Reflection Questions

After running these exercises, consider:

1. **Where did you need to add constraints?** What did the system do wrong when left underspecified?
2. **Where did you add a STOP?** What situations required human judgment?
3. **What did transparency reveal?** When you required reasoning, did you learn something about how the system was deciding?
4. **Would you sign your name under the outputs?** If not, what would need to change?

The Point

These exercises aren't about building production systems. They're about developing the skill of **specification** — being precise about what you want, what constraints apply, and what should happen when things go wrong.

That skill transfers. Whether you're writing prompts, reviewing vendor proposals, or evaluating AI claims, the ability to think clearly about bounded agency is what matters.

A Note on Model Capability

The recipe prompt above was tested across model tiers with dramatically different results:

Behavior	Frontier (Sonnet 4)	Small (Haiku 3.5)
No user input → wait?	Asked what to cook	Built a React app
Tool selection	Used web search	Searched ML papers for meatballs
Followed step sequence	Correct order	Skipped/hallucinated
Stayed in role	Recipe assistant	Became a software engineer

Why this matters: Modern frontier models are reasoning agents — they have built-in capabilities for intent inference, tool routing, and conditional logic. Smaller or older models lack this. The same prompt, the same framework, wildly different results.

The "agent" lives in the interaction between prompt design and model capability — not in either one alone. This is why staying current matters more than clinging to what worked six months ago. The capability floor is rising fast.

Bonus: The Teacher Salary Demo

For an extended example of bounded agency in practice, see the teacher salary analysis materials included with this course:

- **`agentic_workflow_reflection.md`** — A reflection on what worked, what broke, and why human validation mattered
- **`teacher_salary_analysis_guide.md`** — The optimized prompt and validation checklist
- **`teacher_salary_conversation.md`** — The full conversation showing 3 iterations from naive → flawed → correct

The key moment: The AI confidently reported California teachers earning $16k adjusted salary. The human said "That's fishy." That one sentence triggered a complete methodology overhaul. Without it, the analysis would have been published nonsense.

The lesson: Execution speed without validation is just faster failure. The human's job isn't to do the work — it's to know when something doesn't smell right.

Course: AI Agents 101 — Bounded Agency Over Autonomous Agents