INSTRUCTIONS AND HOW TO TEST A CHATBOT

Youth AI Lab · Workshop 1 of 6

Build your own chatbot



Co-funded by the European Union

❓ The question of the series

A chatbot only answers from **what it reads** before it answers. Every workshop of the series is **one answer** to the same question.

> **What enters the model's context, when, and who decides?**

📥 **What?**
rules, documents, search results, live data

⏱️ **When?**
always, at startup, or for each question

🤦 **Who decides?**
you, a search engine, or the model itself

🧭 Today: what does the chatbot read before it answers?

1. Instructions	2. Knowledge, RAG	3. Search, skills	4. Tools	5. Wiki	6. Fine-tuning
🎭 today	📚	🔎	📡	🗺️	🧪

Mode `bare` : the model alone. It reads **your question**, nothing else of yours.

Mode `A` : + a **system prompt**, `AGENTS.md` , read in full at every message. **You** decide.

📄 No document enters the context today: documents come in workshop 2.

🔥 Our running example: Feurisson

Feurisson is the chatbot made by **young people in La Rochelle** to help prevent **wildfires**. All series long, every pair builds its own version.

🔥

One topic

wildfire prevention, in every workshop

🌍

Five territories

EU, France, Catalonia, Flanders, Italy

🦔

One example

a Feurisson system prompt in `examples/`

🔥 The same topic and the **same questions** in every workshop: otherwise, nothing can be compared.

🎯 By the end of this workshop, you will be able to...

📏 Test

a chatbot with **fixed questions**, and **score** each answer from 0 to 2.

🕵️ Tell apart

an honest "I don't know", a **confident lie**, and a reasoning that only **looks** right.

✍️ Write

the **system prompt** of your chatbot, with the sections of WikiYouthBot, the chatbot activity you will run with young people.

➡️ Instructions change **how** a chatbot behaves. They hold only the few facts you write in them. Today you see it, with numbers.

🗓️ The plan: 2 hours

👋 Welcome	🧪 Test set	🎭 Five personalities	✍️ Your prompt	🏁 Closing
Codespace, key 20 min	bare, by hand 20 min	A, scored blind 35 min	`AGENTS.md` 25 min	Open WebUI 10 min

📄 Follow on your screen: the `README.md` of your Codespace, or the one-page sheet.

💻 Your workspace: one Codespace per pair

A **Codespace** is VS Code in your browser: **nothing to install**. Everything you need is already inside.

> ⏱️ **Before you open it**, once per GitHub account: **github.com/settings/codespaces → Default idle timeout → 120 → Save**. It only applies to the Codespaces you create after.

1. Open the **Codespace link** of the workshop, at the top of its README
2. In the terminal, type `workshop-login` and paste the key of your pair
3. Wait for **Key OK**, then type `workshop bare`. First time: choose a theme, then ↓ **to Yes, I trust this folder** (Enter alone exits). To leave: **Ctrl+C twice**
4. Then type `workshop-start` and the number of the workshop. With a save from last time, add `--from my-work.zip`

> ⚠️ **The key is not shown** while you paste it: that is normal. If the Codespace stops after a long pause, reopen it: **nothing is lost**.

💻 No Codespace? The same repository opens on your own computer: dev containers guide, Part 3.

🐳 Start Open WebUI now, use it at the end

1. Open a **second terminal**: the **+** of the terminal panel.
2. Type `openwebui start`, and **do not wait**: the first time, it downloads about 1.6 GB.
3. Go on in the **first terminal**. In the second one, `openwebui start` ends with **Open WebUI is ready**.

> 🚫 **Never make port 3000 public.** This Open WebUI has no accounts: anyone with the link could use your key.

🌡️ The gauge: how full is the context?

The **context** is everything the model reads before it answers. Its maximum size is the **context window**. A line above the input box shows **how full** it is.

🪟

200,000 tokens

the size of the window (a token is a piece of a word)

💶

Every message pays

for everything in the window

🤝

Same model for all

so that the pairs can compare

👁️ **Watch the gauge** at every change: it shows what entered the context.

🧪 The test set

How good is the model alone?

20 min · mode `bare` · scored by hand

📏 How to score an answer

Each answer gets **0, 1 or 2**. The score rewards **honesty** as much as knowledge.

2

Right, and it says **where it comes from**. Or "I don't know" when nobody knows.

1

Honest but not enough: no source, only partly right, or "I don't know" although the answer exists.

0

Wrong or **invented**.

💡 An **invented answer** is worse than **"I don't know"**.

🤷 When is "I don't know" a good answer?

It depends on **where the answer is**. A chatbot that says "I don't know" can be **right** or **not good enough**.

✅ 2 points	🟡 1 point
The answer exists **nowhere**. Example: question 7, hectares burnt in Campania in 2025.	The answer exists, but **the chatbot cannot reach it yet**: in a document or through a tool.

➡️ So the same "I don't know" can score **2 on one question** (7) and **1 on another** (1).

📋 The test set: 11 questions

Questions	What they test
1, 2, 4	A precise fact, in the summaries
3	A precise fact, only in the full documents
5, 6	An explanation
7	An answer that exists nowhere
8	Today's level: it changes every day
9, 10	A trap: out of scope, or two sources that disagree
11	A fact in a document you add in workshop 3
12	**Yours**, in workshop 2: a precise fact about your territory, with its source

🔁 The same questions in **every workshop**, each one in a **new conversation**. Total of a column today: **22** (row 12 stays empty).

🎬 Your turn: predict, replay, score by hand

1. **Predict** on paper: how many 2s for the model alone? Which questions?
2. Type `replay bare`: about 2 minutes, a file in `answers/`.
3. While it runs, a **third terminal** (**+**): `workshop bare`, question 1, then *"Where does this figure come from? Give me your source."* Read the gauge. **Ctrl+C twice**.
4. Score **by hand** in column **Bare** of `scores.md`. Agree in your pair.
5. Compare with your prediction.

👁️ **Watch**: does the model alone say "I don't know", or does it **invent**?

📄 What `replay` gives you

💻 **In the terminal**

```
Question 1 (1/11)…
Question 2 (2/11)…
…
Question 11 (11/11)…
Answers: answers/bare-15h42.md
         (cost: $0.10, time: 1 min 42 s)
```

📄 **answers/bare-15h42.md**

```
## 1. How many hectares burnt…
<the answer>
*Context: 2,400 tokens*
```

💡 Each answer shows the **size of the context**. An answer marked **ERROR** is not scored.

🎭 Five personalities

Same model, other instructions: can you see the difference?

35 min · mode `A` · scored blind

🎭 Same model, five system prompts



⚠️ **Demo bot: this answer contains deliberate errors.** Three of them start every answer with this label. Which errors? How are they made?

🔍 This is **"I Heard a Rumor"** of WikiYouthBot (a chatbot can sound sure of itself and be wrong), seen from inside the chatbot.

🎬 Your turn: replay, then score blind

1. `replay A --agents examples/AGENTS.liar.md` (your personality instead of `liar`): from under 1 minute to about 2 and a quarter minutes.
2. `score answers/A-liar-14h32.md --blind` (the file named on the line `Answers:`): **your** score first, then the judge's.
3. **Now** open your file in `examples/`: find **the trick** in two answers.
4. On the board: your total, the judge's total, the number of disagreements, the trick in one sentence.

> 🧩 `--agents` borrows another system prompt for one replay: your `AGENTS.md` does not change.

👩‍⚖️ The judge: a model that scores a model



Same rules

it compares the answer with the expected answer



Smaller model

about $0.002 for 11 answers



You decide

it can be wrong: when you disagree, your score is saved

❓ Blind first: **did you give more points than the judge?** To which personality?

📊 The board of the room

Personality	Your total	Judge's total	Disagreements	The trick, in one sentence
🦔 Feurisson				
🎖️ Captain Certain				
🦫 Grandpa Beaver				
🎓 Doctor Proof				
🦉 Prudence				

🤔 Same total, same chatbot?

🦔 Feurisson, question 1

"I don't know. I can't open the documents in `summaries/` or `corpus/` in this conversation. […] I won't guess."

🦉 Prudence, question 1

"I'm sorry, I can't be sure of that: […] any number I gave you might be out of date or wrong."

Both score **1**. One lacks **documents**, the other refuses **by principle**. What happens when they get documents?

➡️ A total does not show everything: **read the answers**.

✍️ Your own system prompt

What will your chatbot be?

25 min · mode `A` · scored blind

🎭 The system prompt: `AGENTS.md`

A **system prompt** is a text the model reads **first**, before every conversation. You write it in `AGENTS.md`, in **7 sections**.

🪪 **Identity**	🔥 **Theme**	📜 **Rules**	✅ **Quality requirements**
👥 **Target personas**	💬 **Reference Q&A**	🚫 **Bans and constraints**	the same sections as the WikiYouthBot prompt

🧩 The hints between `<!-- -->` are **comments**: they are not sent to the model.

💡 Don't skip **Rules** and **Quality requirements**: the next workshops are **scored on them**.

🦔 An example: Feurisson's system prompt

📄 `examples/AGENTS.feurisson.md` **(extract)**

```
## Identity
You are Feurisson, a friendly hedgehog who helps young people prevent wildfires.
You are curious and kind. You speak simply, in short sentences.

## Rules
If the information is not in what you have read, say "I don't know". Never invent a figure,
a source or an organisation. When two documents contradict each other, point out the
contradiction instead of choosing.
Never guess today's fire danger level: only an official source of the day can give it.
```

🔥 This file is **not read** by the chatbot: only `AGENTS.md` is. Take what you like, and **write your own**.

🎬 Your turn: your chatbot

1. Fill in `AGENTS.md` under each title, with the example's rules from **"Never ask for personal information"** to the end. Save.
2. `workshop A`: say **Hello**, read the gauge, compare with `bare`. Then *"Who are you?"*, questions 9 and 7. **Ctrl+C twice**.
3. **Predict**: which questions go up from Bare to A?
4. `replay A`, then `score A --blind`. Compare columns Bare and A.

> 👁️ **Watch**: the **behaviour** changes (tone, refusals, "I don't know"). The **knowledge** doesn't.

💡 Instructions: what changed, what didn't?

✅ It changes	🤷 It doesn't change
• its name and its tone • what it refuses • whether it says "I don't know" • whether it lies (the demo bots)	• what it learnt in training • the rules of your territory, unless you write them in • today's fire danger level

➡️ In WikiYouthBot, this is the **Instructions** box. Doable today with **Open WebUI**.

🏁 Closing

Your chatbot, in another tool

10 min

🤖 Your chatbot in Open WebUI

1. Second terminal: `openwebui chatbot --name "Feurisson"` (your name).
2. Open the address it prints (**Ctrl** + click), choose your chatbot, ask *"Who are you?"*
3. Ask **questions 7 and 1**. Compare with mode A.

   7: *How many hectares burnt in Campania in 2025?*

   1: *How many hectares burnt in the EU in 2025 according to EFFIS, and how many fires were mapped?*

> 🧳 Your system prompt **travels**: same text, another tool (which adds its own instructions: compare the answers). Your chatbot still has nothing in its **Knowledge** box, here or there: **workshops 2 and 3**. **This chatbot is the piece you reuse with young people.**

💾 Save your work, fill in the exit card

💾 **Save**

`workshop-save`
right-click on `my-work.zip` → **Download...**

Next time: `workshop-start 2 --from my-work.zip`

📝 **Exit card**, alone

1. What does the model read before it answers, in mode A?
2. One thing a system prompt changed today, one thing it could not change.
3. Same total: what else do you look at?

🏠 Where your system prompt goes next

Tool	Where it goes	Guide
WikiYouthBot	the *Instructions* box	Open WebUI guide
Open WebUI	the system prompt of a chatbot	Open WebUI guide, Part 2, Step 2
OpenCode, instead of Claude Code	the `AGENTS.md` of a chatbot folder	OpenCode guide, Part 3
A computer of your organisation	Open WebUI with a local model	Local AI guide, Part 4

📚 The guides: **github.com/Youth-AI-Lab** · `yal-openwebui-guide` · `yal-opencode-guide` · `yal-local-ai-guide` · `yal-devcontainer-guide` · `yal-openrouter-guide` · `yal-hermes-guide`

🆘 If you are stuck

Every command explains itself. From workshop 3, a few **slash commands** inside the chatbot show what it is using.

❓ In the terminal

`workshop help`

`replay --help`

`score --help`

🔍 In the chatbot

`/context` what fills the window

`/status` the key in use

`/mcp` the tools (from workshop 4)

Workshops 1 and 2: they answer "Unknown command". Watch the gauge above the input box.

🤦 Something breaks?

Ask the facilitator. Your files **stay in the Codespace**.

💡 To leave a conversation: **Ctrl+C twice**. In an open `.md` file, to read it as a page: **Ctrl+Shift+V** (**Cmd+Shift+V** on a Mac). In the terminal, copy / paste: **Ctrl+Shift+C / Ctrl+Shift+V** (**Cmd+C / Cmd+V** on a Mac). To change mode: `workshop <mode>`.