Bring Your Problem: Can AI Help?

TPL Innovator in Residence, Discovery Lab, North York Central Library · 2026-10-05

CONTENTS

Two rules for today

1. The problem you keep doing by hand

What do you keep doing by hand?
Specific, recurring, annoying

2. Seven questions

What is the real problem?
What have you already tried?
What would a useful output look like?
What information is safe to share?
Is this low-stakes or high-stakes?
When the stakes are high: choosing a service
What must a person or an official source check?
A smooth "I am sure" is not evidence
Reusable workflow, or a coded tool?
Open or local tools?
If not: solve it, skip it, or refuse it.

3. Two problems, worked out loud

Case one: a stack of letters
Case two: everything checked out
Different costs and values

4. Your turn

Where does it belong?

5. Where this goes

How to get started
OpenCode
Getting OpenCode
Thank you

Welcome. This is the first workshop after the opening talk, which means it is the practical one. Almost everyone here does something by hand, over and over, that feels like it should be easier. We are going to take one of those things and decide together whether AI should touch it at all.

Nothing gets built today. We are learning about the decision to use AI: what it might help with and if it can do so safely. That is the whole session.

And the order matters, because as with most things in life you can get into trouble if you jump in without looking. We are going to start with the problem, and dig into the way to think problem solving with AI. If you want to build things with AI, that's great, and there are other sessions and the hackathon where we do that, but knowing when and what to build is the most important part. Just because you can doesn't mean you should.

Two rules for today

Two rules for today:

- No devices. We decide before we demo.
- Keep your personal problems out of the AI (for now).

In a while you will each write one problem on a card, and we'll write the problem without the private details, and not write your name on the card. There is a box on it that says I may photograph it. I will keep the photos of ticked cards to study and to bring to the hackathon in December as possible problems for people to build tools for. The card is yours to take home either way.

1. The problem you keep doing by hand

What do you keep doing by hand?

- forms and paperwork that keep coming back
- appointments and renewals to keep track of
- the one phone number or office you can never find
- comparing prices before you buy something
- a to-do list that never gets shorter

If you are stuck, look at that list and tell me the closest thing, even if it is not exact. And if you do not have a problem, stay with me: helping a neighbour with theirs is the same exercise, and we have cards at the end for anyone who wants one.

Specific, recurring, annoying

Three words for what makes a problem worth bringing. Specific, so we can actually work on it. Recurring, so the fix is worth the trouble. Annoying, which is a good sign, not a bad one.

Something small that happens every week is worth more of our time today than something enormous that happened once. If you brought something big, that is fine too, but we will probably work on the small version of it, because that is the part we can see clearly.

2. Seven questions

1. What is the real problem?
2. What would a useful output look like?
3. What information is safe to share?
4. Low stakes, or high stakes?
5. What must a person or an official source check?
6. Reusable workflow, or a coded tool?
7. Open or local tools? If not: solve it, skip it, or refuse it.

Seven questions, in this order, one at a time. Nothing here requires a computer, and most of it is just the kind of thinking you already do when a job goes wrong.

We will walk through each one, then do them out loud twice, then you will do them with your own problem.

What is the real problem?

What have you already tried?

Sometimes that hardest thing to do is identifying the real problem. The thing that annoys you is rarely the problem. "My desk is a mess" is a symptom, not a problem. The problem is that paper arrives every day and nothing decides what to keep.

Before anyone, much less an AI can help you, you should do some exploration. What have you already tried? The thing you tried tells you what the real problem is. If you tried three apps and stopped using all three, the problem is not finding an app. It is that setting it up costs more than the thing it saves.

> *Our league changes game times by email and I only find out when I am standing in an empty field. I tried phone reminders, but the changes never reach my calendar.*

real problem

> *I am bad at keeping up with my email.*

symptom

The first has a place where it goes wrong and something already tried, so there is something to work on.

The second is a feeling, and nothing in it tells you where to start.

What would a useful output look like?

Before you ask anything or anyone for help, write down what a good answer looks like. A draft. A list. One page. One question answered.

That sounds obvious, and it is where most disappointing AI experiences start: somebody asked for something, got something, and had no way to judge it, because they never said what they wanted.

If you cannot describe the output, you cannot tell whether the tool succeeded, and you cannot tell whether it failed either.

> *Make my newsletter better.*

no way to judge it

Better than what, and by whose measure? You could read ten drafts and never know whether any of them succeeded.

> *A half-page newsletter for our building: three items, plain words, under 150 words.*

a useful output

This one you can hold a draft up against.

What information is safe to share?

Share the problem, not the private details. This is the rule that lets everyone speak freely for the rest of the session.

Almost every real problem has a version that carries none of your private information. "I keep losing track of appointments" does not need your actual appointments. "My father's medication schedule is confusing" does not need his name, his doctor, or his condition.

And if the problem only works with the private details, that is worth knowing right now, before you spend an afternoon on it. That is the kind of problem where the cost arrives later, as a bill or as a leak.

> *Help me write a complaint about noise from the apartment upstairs. The names, the address and the dates stay with me.*

safe to share

The problem can be asked with nothing private in it.

> *Here are my last year of pay stubs, with my social insurance number on them. Am I being underpaid?*

not safe to share

This one only works with the private details, and that is worth knowing before you type any of them.

Is this low-stakes or high-stakes?

Same tool, completely different answer, depending on this one.

Sorting your photos is low stakes. Worst case, the albums are wrong and you move a few pictures around, and if things go really badly it might delete some, which is why you keep a backup, or tell it to make one first and to ask permission before it changes anything. Making an action reversible is how you lower the stakes.

Some stakes can't be lowered that way, because some actions can't be undone. Deciding what a medication does, or whether you qualify for something, is high stakes, and the worst case is that you act on something wrong.

Stakes decide how much checking an answer needs, and they decide who has to do the checking. Low stakes, you can be the checker. High stakes, you need a person or an official source, and no amount of confidence from a machine replaces that.

> *Should I send in my work permit renewal this month or next?*

high stakes

If the permit answer is wrong, you may not find out until it is too late to fix.

> *Give me ten names and a colour scheme for our book club flyer.*

low stakes

If the flyer is bad, you print another one.

When the stakes are high: choosing a service

Question four sends a lot of people to the fine print, so let's read some of it. You will see the phrase "zero data retention", and it is real, but you cannot have it: it is a contract that a company applies for and the vendor approves, it covers the developer side of the product rather than the chat window, and nobody gets it by ticking a box in a personal account.[1] Even for the companies that do have it, it is a promise with terms. In June 2026 Anthropic started keeping prompts and outputs for thirty days on its newest models, including for most business customers who had zero retention, and they announced it and gave their safety reasons.[2] Ordinary Claude chat was not part of that change, which is the point: the promise belongs to the vendor, and it is worth reading the date on it.

For anything that matters, I would look at four things, and none of them is on the home page. First, whether your chats train the model and whether you can switch that off, which is a setting in the account that you want to find before you type anything. Second, how long chats are kept and who can read them, because deleting is not erasing: at Google, a chat that a human reviewer has looked at is kept for up to three years even after you delete your activity,[3] and for several months in 2025 a US court made OpenAI keep chats that users had deleted.[4] Free developer tools are no different: Google says that on the free tier of AI Studio, people may read what you type, and it tells you not to put in

anything personal.[5] Third, which company you are really talking to, since an app built on top of a chatbot has its own policy and then the chatbot's policy sits underneath it. And fourth, whether it is legally private, and a chat with an AI has none of the protection of a talk with your doctor or your lawyer, and a court can demand it.

Two things are about the world rather than the settings. In May 2026 four Canadian privacy regulators found that OpenAI broke privacy law in how it trained ChatGPT, including on what people typed into it, and OpenAI said it cannot take one person's information back out of a trained model.[6] So the law exists, but it works after the fact, and it did not stop your chat going in. And as far as I can tell, health privacy law mostly binds your clinic and your hospital and not the chatbot, which means that once you type your own results into an app you have stepped outside the system that was protecting them.

A leak does not need a villain, either. Chat apps have shipped share links that search engines then indexed, and databases have been left open on the internet,[7] and anything that is stored can be taken, by someone outside the company or by someone inside it, and a company that lets AI agents work inside its own systems has added a new kind of insider. I think the only protection that does not depend on someone else's promise is storing less, which is question three again. And when the details are the whole problem, that is question seven's answer: a tool on your own machine, which will be weaker, or a person, or waiting until the local tools catch up.

Sources: 1, 2, 3, 4, 5, 6, 7

What must a person or an official source check?

This question decides whether the whole thing is safe to use. It is also the question most people answer with a feeling: it sounded right, it was detailed, it was confident.

The feeling is not usable, because confidence in the words is the machine's least reliable channel. These models are trained to be helpful, and an answer that sounds sure feels more helpful than one full of maybes, so the training tends to reward sounding sure. It is hard to build an assistant that is both useful and clear about what it does not know.

The more useful signal about confidence is hidden inside the machine, in the math, and you cannot see it from the outside. There is ongoing research on making a model's confidence in its answers more visible and more correct, but it is a hard problem, and so far an unsolved one.

So the rule is short: trust the verification, not the vibe.

> *An AI summary of the new rules for the community garden plots, which I plan to read out to everyone.*

a check is needed

Somebody has to confirm which version of the rules is the current one, and the machine cannot see the filing cabinet.

> *A packing list for a weekend camping trip.*

no outside check needed

You are the checker here, because you will notice what is missing, and a forgotten sock is cheap.

A smooth "I am sure" is not evidence

Researchers at Google DeepMind tested this directly: do these models carry an inner reading of their own sureness, and does it change what they do?[8]

Free to pass on a thousand hard questions, one model stayed quiet more than half the time, and the answers it did give got more accurate: about 64 percent right before, about 69 percent right after.[8] Its hidden sureness reading predicted staying quiet about ten times better than how hard the question looked.[8]

You cannot see that reading. But you can use two levers, in any chatbot, starting today. One: "if you are not sure, say I do not know." Two: "rate your confidence from 1 to 10." Both change what it does.

The self-grade is a flag, not a verdict. A low score means check this one. A high score never means skip the check.

And they differ from each other like people do. Across the four models tested, how often one stayed quiet ran from about 27 percent to about 82 percent.[8] An assistant that never says "I do not know", even when you invite it to, is telling you something about itself.

Sources: 8

Reusable workflow, or a coded tool?

Most people jump straight to building something, and almost nobody needs to.

A reusable workflow is instructions you or somebody else can follow again: the steps, the order, the checks, and one example of a good answer. You can keep it in a document, or on a card. That covers most of what we will map today. Any online chatbot with a recent AI can handle these sorts of problems and you can save the workflow into a text file or document and just paste it into a chat when you need it. It's simple and effective for problems that don't require the AI to interact with your computer or do things on your behalf.

A coded tool is for when you do the same thing often enough that the setup starts paying for itself, and when the workflow has stopped changing every week. Start with the workflow. It is cheaper, and it tells you whether the tool is worth building. There is a growing number of tools that allow for "agentic AI" or "AI agents" or "AI assistants", these can interact with your computer, create software for you, and run on a schedule. These are far more powerful and we'll talk about this approach in some of the other talks like "Peek Inside an AI Assistant: How It Thinks and Works" and "Run Your One-Person Business with AI".

> *Every Friday a supplier sends me the same spreadsheet and I rebuild it, by hand, into the same four-column list. It has not changed in a year.*

a coded tool

This is the same job every week, which is what a small tool is for.

> *Every few weeks I cook dinner for friends who eat differently. I paste the same prompt into a chatbot: it asks who is coming and what they can't eat, suggests a menu, then writes the shopping list and a cooking schedule.*

a reusable workflow

The details are different every time, so the prompt asks for them first, and nobody had to write any code.

Open or local tools?

If not: solve it, skip it, or refuse it.

First, where does the tool live? Most AI tools run on a company's computers, far away from you, and you borrow them. That is a cloud tool. A local tool runs on your own computer, so nothing leaves it. An open tool is one that anyone can copy and use, so it does not belong to just one company.

Here is the test I use all fall: would this still work if the API shut off tomorrow? An API is a drive through service window for software. A program hands in a request, and the AI company's computers hand back the answer, the way a drive-through window works. Many apps are built on a big company's AI this way, and the company runs the window. It can close it, change how the AI works or raise the price, whenever it likes.

Here is where I land. For most of the things people want to do, I think local tools are not good enough yet. The cloud tools do a better job, and they cost less. Most everyday jobs are low stakes, too, so a cloud tool, with your private details left out, is a fine answer.

Local tools matter in a few cases. The details cannot be left out. The stakes are high. Or you use the tool so much that a price jump would hurt. Then a local tool can be worth the extra work, even though it is weaker. A person is often better still.

And if the only way you would feel safe is a local tool that is not good enough yet, I think you should wait. Leave it for now, and come back when local tools are better and cheaper.

Then the last part, because not every problem gets solved. There are three answers, and two of them are good. Solve it. Skip it, which means it costs too much right now, or it is not safe enough yet, and that is a choice, not a failure. Or refuse it, which means you can see a way to make it work, and you do not want the world it would make.

Refusing is an answer too. Please do not treat it as a failure today.

> *Summarizing my own journal once a month. A model on my own computer does it, and nothing leaves the machine.*

a local tool works

A tool on your own machine does this well enough, and keeps it private.

> *After reading my journal once a month, I want it to suggest changes to my life that would improve my wellbeing.*

skip it for now

This needs good advice, and I suspect that is more than a local model can give you today, while a cloud model that could give it would be reading the most private thing you own. So you skip it for now, and a person is probably still the better reader of a journal.

> *A weekly message to my mother, written by AI, so I never have to think what to say.*

refuse it

This would work, and I do not want the family it makes.

3. Two problems, worked out loud

Case one: a stack of letters

> *"I have three letters from the government and I do not know what they say."*

1. What is the real problem?
2. What would a useful output look like?
3. What information is safe to share?
4. Low stakes, or high stakes?
5. What must a person or an official source check?
6. Reusable workflow, or a coded tool?
7. Open or local tools? If not: solve it, skip it, or refuse it.

Watch what happens when we run the seven on this.

The real problem is not the letters, it is that she cannot tell which one has a deadline in it, and she has already tried asking a neighbour, once, and felt bad about it. A useful output is one page: what each letter is, what it wants, and by when. That is specific and it is small.

Is it safe to share? The version above has no names, no file numbers, no address. Safe. And it is high stakes in a quiet way, because missing a deadline has real consequences. So question five lands hard: a person or the official source has to confirm the deadline. The machine can sort the pile and explain the language. It cannot be the last word on a date.

Workflow or tool? Workflow. A card with four steps beats an app she has to learn.

Case two: everything checked out

> *The clinic is on the third floor. The elevator has been out since spring, and no website says so.*

1. What is the real problem?
2. What would a useful output look like?
3. What information is safe to share?
4. Low stakes, or high stakes?
5. What must a person or an official source check?
6. Reusable workflow, or a coded tool?
7. Open or local tools? If not: solve it, skip it, or refuse it.

The second case is the one I want you to remember.

An AI assistant was helping a clinic route people to their appointments. It was accurate. It read the official directory. It read the accessibility page. And it sent a man who uses a walker to a clinic on the third floor.

The elevator has been out since spring. The accessibility page was written before that, and nobody updated it.

Everything up to question four was fine: the output was useful, the information was handled carefully, and the stakes were understood. Question five was skipped, so nobody checked the one thing the machine could not see, which is the building, on the day.

I think this is what AI failure usually looks like in real work: something correct, in the wrong place.

And the check would have cost one phone call to somebody who works there, with one question. Is the elevator working?

Different costs and values

One more thing before you go work on your own.

Sometimes the problems where AI helps most are the problems where it is most risky or costly to use it.

Take health questions. If you are generally healthy, there is not much value in having a machine look at your medical details, and there is real risk in putting them out there. So you keep them private, and you are not missing much.

But imagine somebody with a serious condition. Suddenly there is genuine value: the machine can help them understand, prepare questions, catch something. And the same person is far more exposed, because their information is worth more and is more useful against them. The person with the serious condition may also be more desperate for answers, real or not.

That is the shape of it. The people who need the help most are the ones most at risk from asking for it. So the answer is never "never use it". The answer is question three, what is safe to share, and question five, what must be checked, and deciding on purpose.

4. Your turn

- write your problem on a card, one line, no name
- say to your group, combine any that match
- take the cards one at a time through all seven

The rule for group work is question three: share the problem, not the private details.

If you get stuck, that is normal, and it is useful. Where you got stuck is a finding too.

Where does it belong?

- AI might help here
- A person or an official source has to check something here
- Better done without AI

Pick one of your cards and tell us where it belongs. Then we will argue a little, which is the useful part.

5. Where this goes

How to get started

- arena.ai
- Free AI on the web:
  - Google Gemini, OpenAI ChatGPT, Anthropic Claude, Mistral
- Free "agentic" AI:
  - it does the work for you, not just the talking

You can start today with nothing to install. Open Gemini, ChatGPT, Claude or Mistral in your web browser. Each has a free plan. Type your problem the way you would say it to a person, and then ask follow-up questions. Before you paste anything private, ask yourself question four: is it safe to share? The free plans are the ones most likely to learn from what you type.

You can use the arena.ai website that we used today that will allow you to use the top AI for free, and compare multiple AI's answers. Everything you type in will be made available to researchers and others studying the AI though, so absolutely nothing sensitive.

The second kind of AI you can use is designed for creating software, sometimes called an agent. A chat tool answers you. An agent goes and does the work: it reads your files, makes changes, and checks them. Many of them have free AI built in, so you can try one without paying. Here is a very quick example.

OpenCode

- Open-source, originally for building software
- A desktop app, a terminal program, or inside your code editor
- Works with any AI, and comes with free ones

OpenCode is an open-source agent made for building software, like Claude Code, except that you choose the AI. It comes as a desktop app for Windows, Mac and Linux, as a program you run in the terminal, or inside a code editor like VS Code.

Getting OpenCode

```
opencode.ai/download
```

- Download the desktop app

- Type `/models` to pick an AI
- Free ones mostly have **Free** in their name
- Or pay: OpenCode Go starts at $10 USD/month

Download the desktop app from opencode.ai/download. To choose the AI, type slash models. Most of the free ones have the word Free in their name, and they work without an account. The cheapest paid plan, OpenCode Go, starts at ten US dollars a month, which is more in Canadian, and the free models are enough to start. To use an AI you already pay for, type slash connect.

Thank you

Thank you for coming. You're ready to get started with AI in your life. The problems you let me photograph do not get filed and forgotten. They are where the challenges we build together at the hackathon in December come from, which is what this program does with what the community brings it. If you ticked the box, you may see your problem again, and either way your card is yours: pick it up on the way out, and I will photograph the ticked ones where they lie.

Sources

1. OpenAI, "Data controls in the OpenAI platform", developers.openai.com/api/docs/guides/your-data.
2. Anthropic, "How long do you store my organization's data?" and "Data retention practices for covered models", privacy.claude.com (updated 2026-09-05).
3. Google, "Gemini Apps Activity", support.google.com/gemini/answer/13594961.
4. Order of Magistrate Judge Ona T. Wang, 13 May 2025, The New York Times v. OpenAI, requiring OpenAI to preserve output logs. Secondary reports say the preservation obligation for new logs ended in September 2025.
5. Google, Gemini API Additional Terms of Service, ai.google.dev/gemini-api/terms (last modified 2026-04-28).
6. Office of the Privacy Commissioner of Canada with the Quebec, BC and Alberta commissioners, PIPEDA Findings #2026-002, Joint Investigation of OpenAI OpCo, LLC (2026-05-06).
7. Security-press reports of exposed chat data: Grok share links indexed by search engines (Aug 2025), Vyro AI apps (Sep 2025), Chat & Ask AI (Feb 2026) and DeepSeek's open database found by Wiz (Jan 2025).
8. Kumaran, Daw, Osindero et al. (Google DeepMind), "Causal evidence that language models use confidence to drive behaviour", Nature Machine Intelligence (2026), doi:10.1038/s42256-026-01293-x.