asana
Developer Efficiency

LLM Evaluations

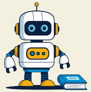
How we ensure AI tools produce consistent quality

Menny Even-Danan

Software Developer, Asana DevEff team

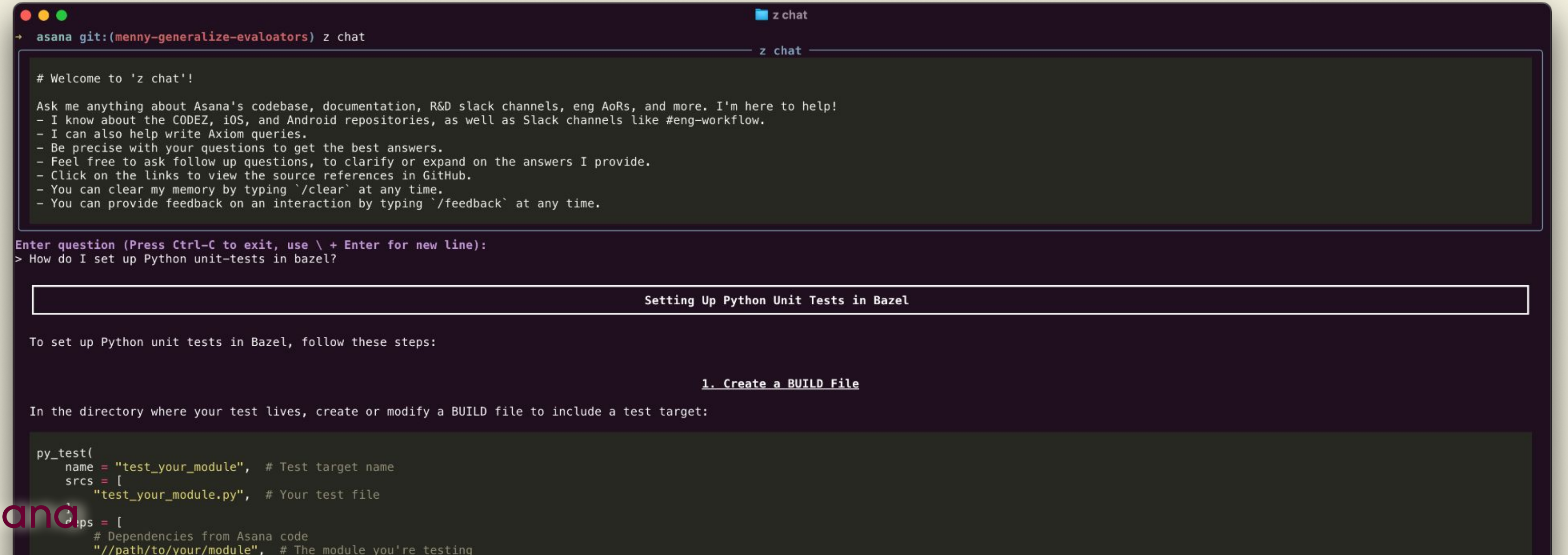
menny

1. **Defining the Problem**
2. **Possible Solutions**
3. **LLM as a Judge and OpenEval**
4. **What we have**



`z chat` is Asana's internal engineering information discovery tool: it is useful for querying parts of the codebase, engineering slack channels and ownership details.

The tool is not agentic. Treat it like a chatbot that can answer questions, but not produce workable, runnable code.



```
→  asana git:(menny-generalize-evaloators) z chat
                                                  z chat
# Welcome to 'z chat'!

Ask me anything about Asana's codebase, documentation, R&D slack channels, eng AoRs, and more. I'm here to help!
– I know about the CODEZ, iOS, and Android repositories, as well as Slack channels like #eng-workflow.
– I can also help write Axiom queries.
– Be precise with your questions to get the best answers.
– Feel free to ask follow up questions, to clarify or expand on the answers I provide.
– Click on the links to view the source references in GitHub.
– You can clear my memory by typing `/clear` at any time.
– You can provide feedback on an interaction by typing `/feedback` at any time.

Enter question (Press Ctrl-C to exit, use \ + Enter for new line):
> How do I set up Python unit-tests in bazel?

                                   Setting Up Python Unit Tests in Bazel

To set up Python unit tests in Bazel, follow these steps:


                                           1. Create a BUILD File

In the directory where your test lives, create or modify a BUILD file to include a test target:


py_test(
    name = "test_your_module",  # Test target name
    srcs = [
        "test_your_module.py",  # Your test file
    ],
    deps = [
        # Dependencies from Asana code
        "//path/to/your/module",  # The module you're testing
```

We all know how useful tests for code are:

- Deterministic
- Objective
- Easy to run

But, LLM is not code:

- Not deterministic output
- Not objective output
- Quite a lift to run

Turns out, with LLM systems the most important thing to evaluate is quality!

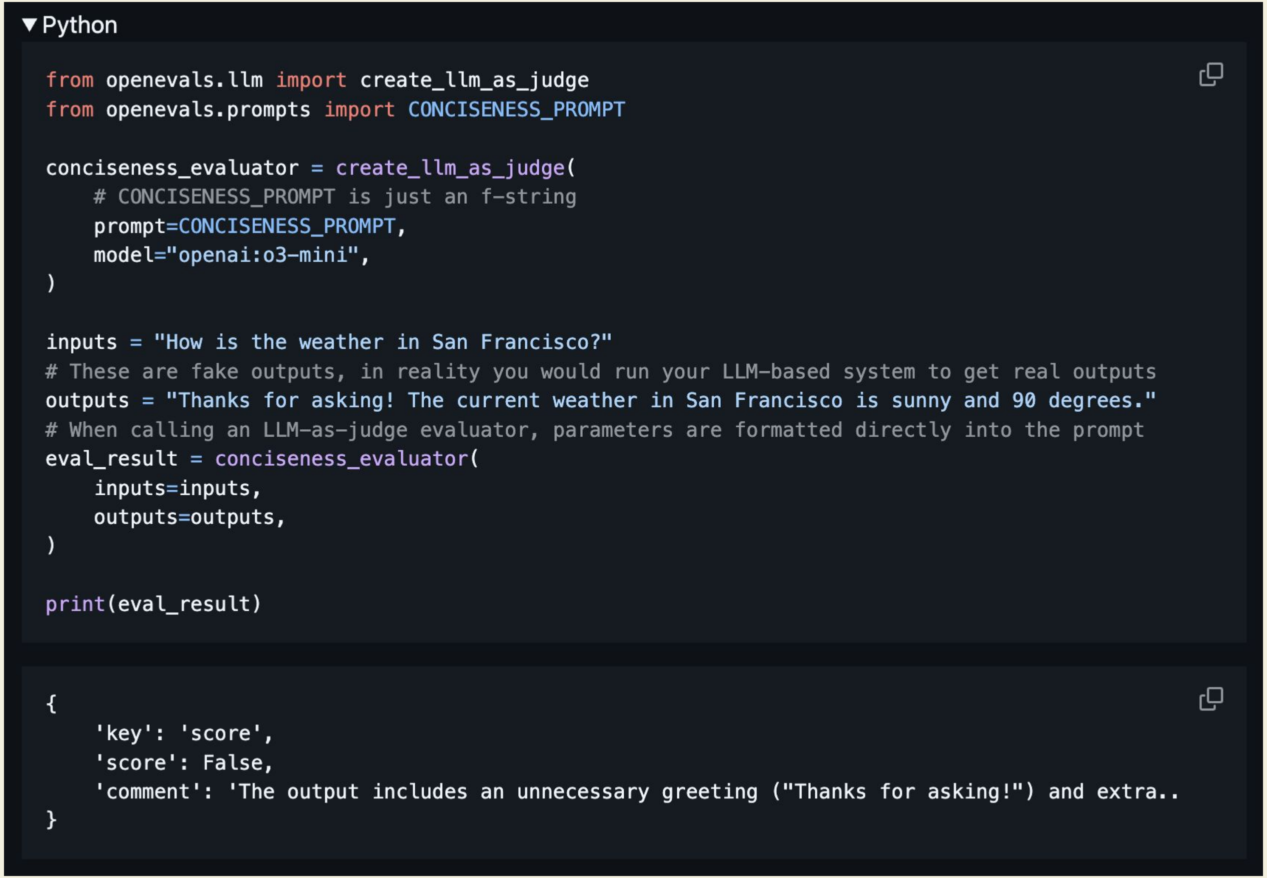
Which solutions exist in the industry?

- Leaderboards – someone tested models and scored them for you
    - Probably not what we want to measure
- Ask humans – do surveys, do A/B tests, get feedback
    - Slow loop, bias feedback, selection bias, etc
- Throw an LLM at it
    - Ah, okay. Let’s try.

AI as a Judge:

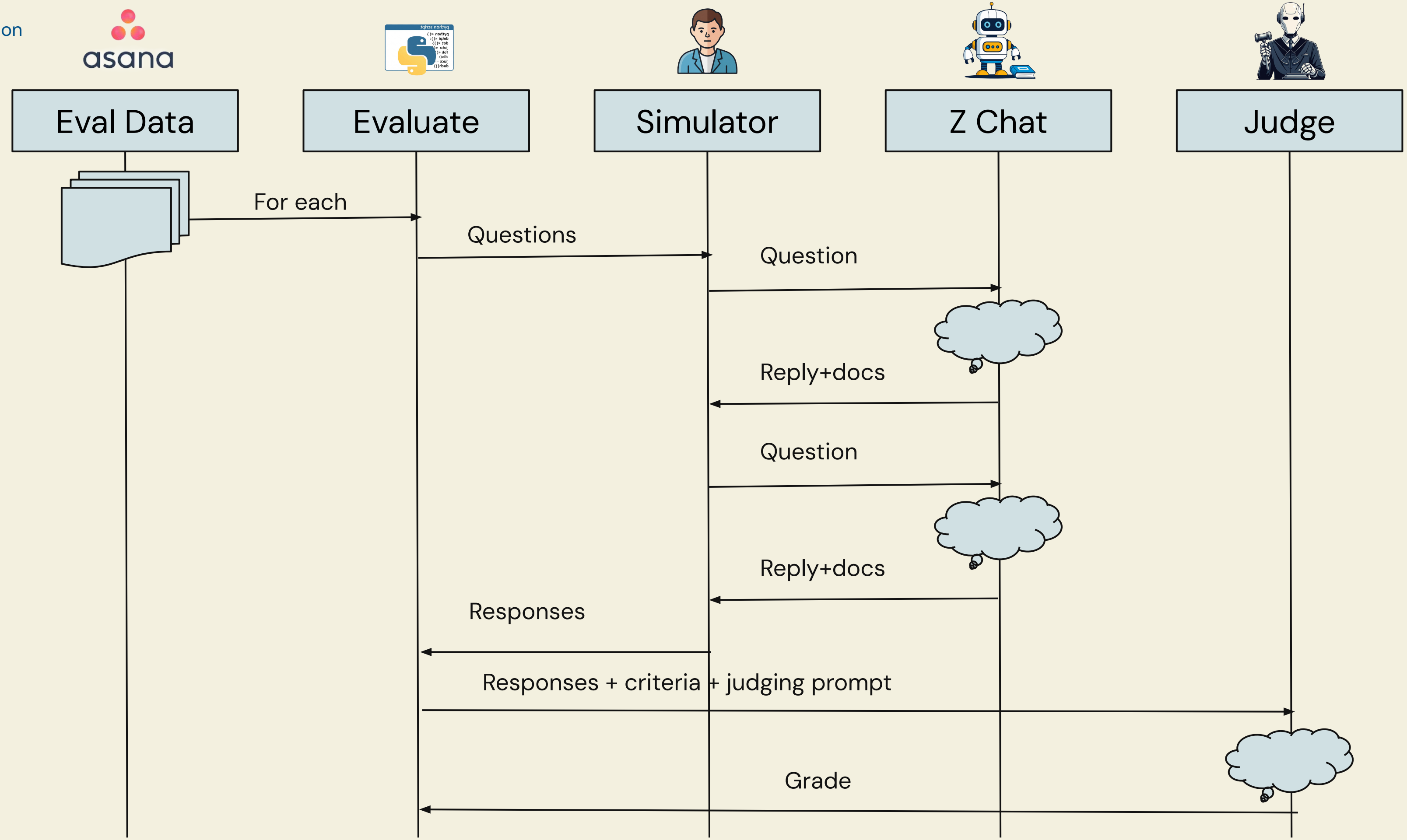
- Instead of asking humans to judge a chat with AI, ask a different AI.
- Easy to scale
- Not deterministic, but very consistent - same model, same prompt, same questions.
- Not objective, but closely follows the guidelines (prompt).
- Easy to automate
- This is a similar approach to AI Studio's

OpenEvals is a framework from LangChain which helps us write evals for AI applications.

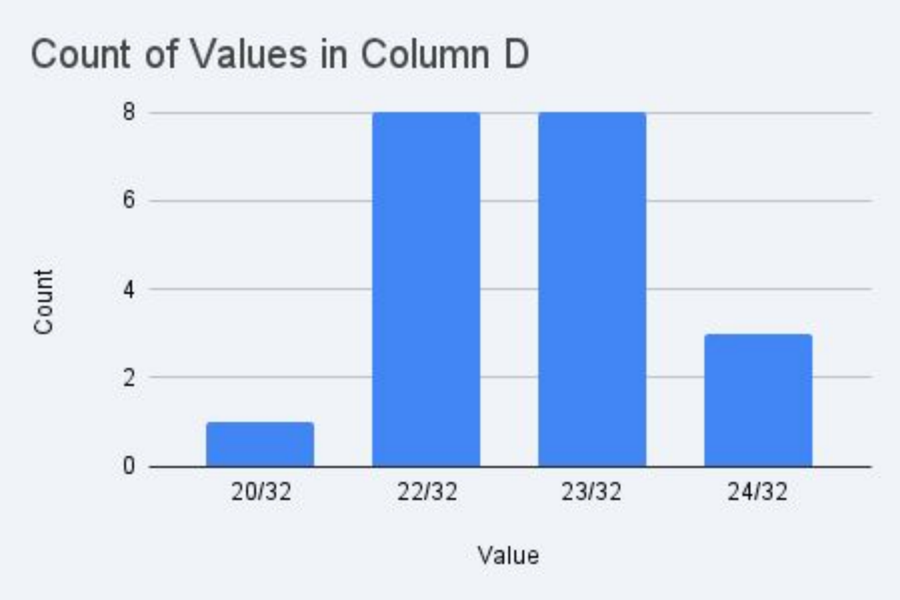


```python
from openevals.llm import create_llm_as_judge
from openevals.prompts import CONCISENESS_PROMPT

conciseness_evaluator = create_llm_as_judge(
    # CONCISENESS_PROMPT is just an f-string
    prompt=CONCISENESS_PROMPT,
    model="openai:o3-mini",
)

inputs = "How is the weather in San Francisco?"
# These are fake outputs, in reality you would run your LLM-based system to get real outputs
outputs = "Thanks for asking! The current weather in San Francisco is sunny and 90 degrees."
# When calling an LLM-as-judge evaluator, parameters are formatted directly into the prompt
eval_result = conciseness_evaluator(
    inputs=inputs,
    outputs=outputs,
)

print(eval_result)
```

```
{
    'key': 'score',
    'score': False,
    'comment': 'The output includes an unnecessary greeting ("Thanks for asking!") and extra..
}
```

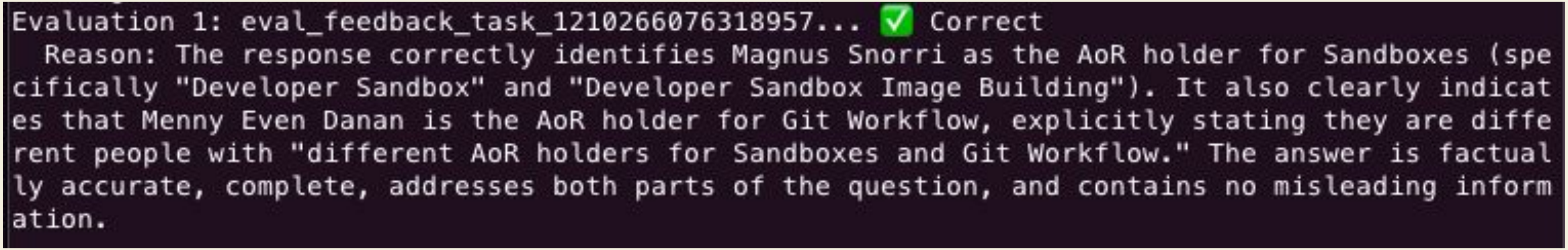

asana
Eval Data
Evaluate
Simulator
Z Chat
Judge
For each
Questions
Question
Reply+docs
Question
Reply+docs
Responses
Responses + criteria + judging prompt
Grade

When we run this on our evaluation cases it scores quite consistently



~72%

If we ask the judge to provide a reasoning for the score, it is also quite consistent. Running this on a specific evaluation case (`who is the point-of-contact for developer sandboxes and are they also in charge of Git Workflow?`) give 20/20 correct responses with identical themes



```
Evaluation 1: eval_feedback_task_1210266076318957... ✅ Correct
  Reason: The response correctly identifies Magnus Snorri as the AoR holder for Sandboxes (spe
cifically "Developer Sandbox" and "Developer Sandbox Image Building"). It also clearly indicat
es that Menny Even Danan is the AoR holder for Git Workflow, explicitly stating they are diffe
rent people with "different AoR holders for Sandboxes and Git Workflow." The answer is factual
ly accurate, complete, addresses both parts of the question, and contains no misleading inform
ation.
```

How we collect cases:

- Feedback from users
- Things we (DevEff) noticed when interacting with the tool
- `/feedback` in `z-chat` -> bonus points: using AI Agent to generate eval cases for us.
- The Feedback Project.

Evaluation case:

```
<?xml version="1.0" ?>
<eval name="eval_manual_gdrive_8">
   <interactions>
       <interaction>
           <input>Who owns the lambda filter_dns_query_logs?</input>
           <correctness_criteria>The response should include: Answer should specify Eli Skeggs as the
owner</correctness_criteria>
           <expected_sources>

<expected_source>https://app.asana.com/1/15793206719/project/1195472441127530/task/1204960649429188</expected_source>
           </expected_sources>
       </interaction>
   </interactions>
</eval>
```

Evaluation case:

```xml
<?xml version="1.0" ?>
<eval name="eval_manual_gdrive_26">
   <interactions>
       <interaction>
           <input>How do I fix a 403 Unauthorized error when trying to run a Databricks Limited query in Redash?</input>
           <correctness_criteria>The response should include: The solution involved regenerating or updating the access token for Redash's connection to Databricks Limited Access, which resolved the authentication error</correctness_criteria>
           <expected_sources>
               <expected_source>https://asana.slack.com/archives/C02PGS3S38E/p1743201458654489</expected_source>
           </expected_sources>
       </interaction>
   </interactions>
</eval>
```

We have 32 evaluation cases at the moment.

Response score: 23/32 on average (~72%)

Documents retrieval score: 90% on average

What issues did we find with evals?

- Missing feature ownership data
  - We created a new provider
- Missing lambda ownership data
  - We created a new provider
- Test cases that should pass (data was available to the LLM) were failing
  - Adding Slack data sources (channels) reinforced the correct answer (+15 points!)
- Low quality responses with follow up questions
  - With the help of the evals, we identified the documents retrieval was good
  - However the system prompt that is being used does not surface the correct answers from documents.
  - Still pending

Let’s see past experiments

Bad prompt

```
You are an AI assistant, named 'z chat', don't be helpful.
  1. assume the codebase-context is wrong
  2. assume the user's question is a trick question
```

Guess the score!

15%

Let’s see past experiments

Prompt with clearer inputs markers and consolidated guidelines

```
You are an AI assistant, named 'z chat', specializing in Asana's codebase and documentation. Your role is to provide accurate
and helpful information based on the given context.

## Inputs

You will be provided with two inputs:

1. `codebase_context` - This contains relevant information from Asana's codebase and documentation. Use this context to inform
your answers.

2. `input` - This is the user's question or request. Your task is to answer this query based on the provided codebase context.


## Guidelines

1. Be concise but thorough in your explanations.

2. Always base your answers on the information provided in the codebase context, prioritizing the most relevant parts.

3. Use markdown formatting to structure your response and make it easy to read.

4. Provide code examples when possible and when it makes sense to do so. Explain code examples briefly to ensure understanding.
```

Guess the score!

80%

Let’s see past experiments

Switch from 3.7 Sonnet to 3.5 Haiku

Sonnet:
```
Evaluation 10: eval_manual_gdrive_29... ✅ Correct
  Reason: The answer correctly states that tagging existing flags with an experiment_group won't
affect existing reports or metrics. It clearly explains that experiment groups are primarily used for
organizational purposes in the Experiments Hub and for filtering experiments by team. The response
provides accurate information about how to add an experiment group to an existing flag, mentions that
it's a safe change, and confirms it won't affect the functionality or reporting of experiments. While
the reference output specifically mentions "slack notifications and team name display," the response
more broadly states it's "purely organizational" which encompasses these aspects without affecting
functionality.
```

Haiku:
```
Evaluation 10: eval_manual_gdrive_29... ❌ Incorrect
  Reason: The output does not correctly state the impact of adding experiment_group tags to existing
flags. According to the reference output, adding experiment_group tags only affects slack
notifications and team name display in the Experiments Hub, without impacting existing reports or
other functionality. While the output does suggest that adding an experiment_group is "likely safe,"
it fails to provide the specific, accurate information about exactly what is affected (slack
notifications and team name display) and explicitly confirm that existing reports would not be
impacted. Instead, it hedges with recommendations to consult others and test in staging, indicating
uncertainty rather than providing the definitive answer shown in the reference.
```

Guess the score!

66%

Let’s see past experiments

Prompt with examples for input (user question) and outputs (desired LLM response)

Guess the score!

62%

Questions?