**Using trace() with OpenAI**

The OpenAI Agents SDK provides a built-in trace() function for monitoring and debugging your AI agent workflows.

**What tracing does**

Tracing captures a detailed record of events occurring within your agent's execution, including LLM generations, tool calls, handoffs, and guardrails. This information can be visualized in the OpenAI Traces dashboard to help you identify performance bottlenecks, diagnose issues like inaccurate tool calls, or understand the agent's decision-making process.

**How to use trace()**

The trace() function can be used in two main ways:

1. Context Manager (Recommended): Wrap the agent's execution within a with trace(...) block. This automatically handles starting and ending the trace.

python

from agents import Agent, Runner, trace

async def main():

agent = Agent(name="Joke generator", instructions="Tell funny jokes.")

with trace("Joke workflow"):

first\_result = await Runner.run(agent, "Tell me a joke")

second\_result = await Runner.run(agent, f"Rate this joke: {first\_result.final\_output}")

print(f"Joke: {first\_result.final\_output}")

print(f"Rating: {second\_result.final\_output}")

1. Manual Start/Stop: You can manually call trace.start() and trace.finish() to define the trace boundaries. Note that if you manually manage the trace, you'll need to pass mark\_as\_current and reset\_current to start() and finish() respectively to update the current trace.