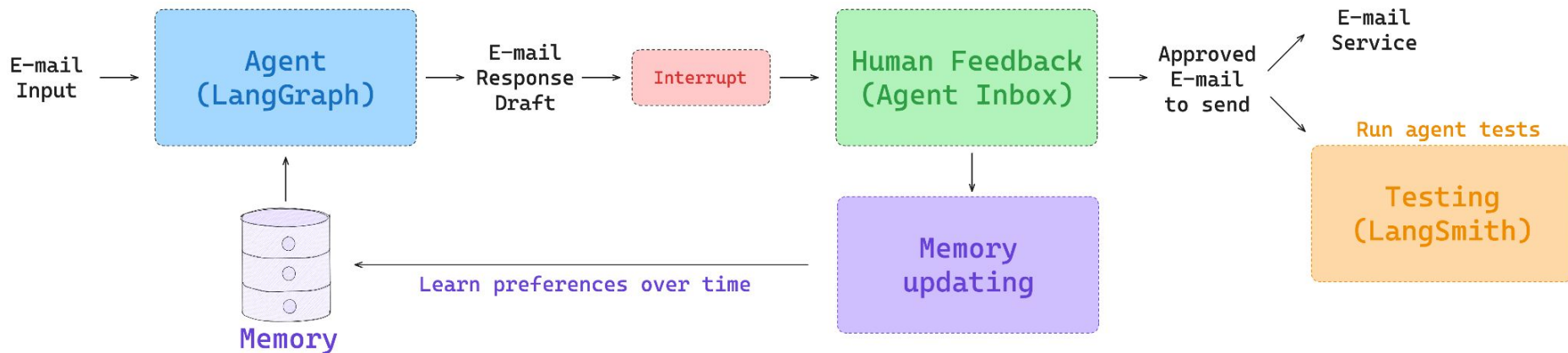


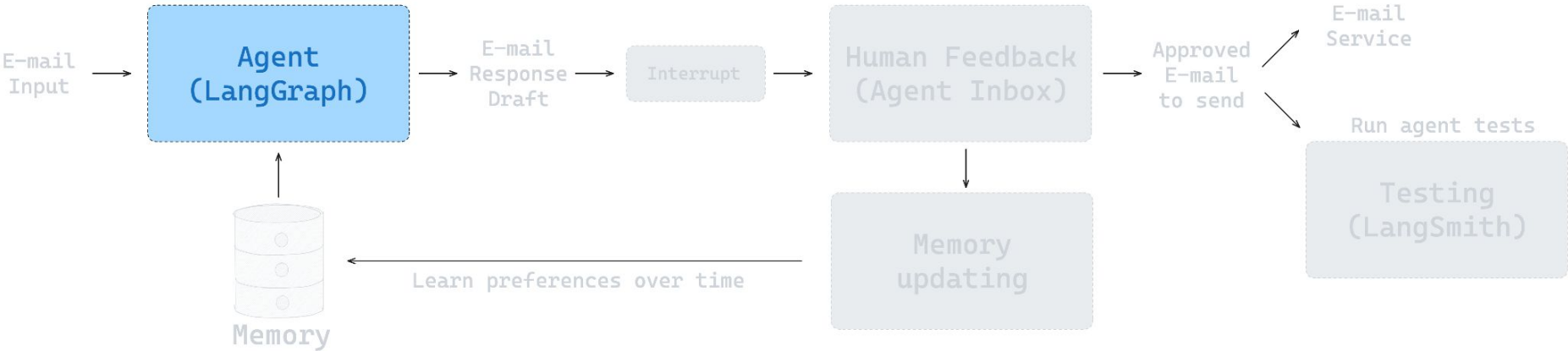
# Building ambient agents with LangGraph

Building + testing long-running agents with human-in-the-loop and memory

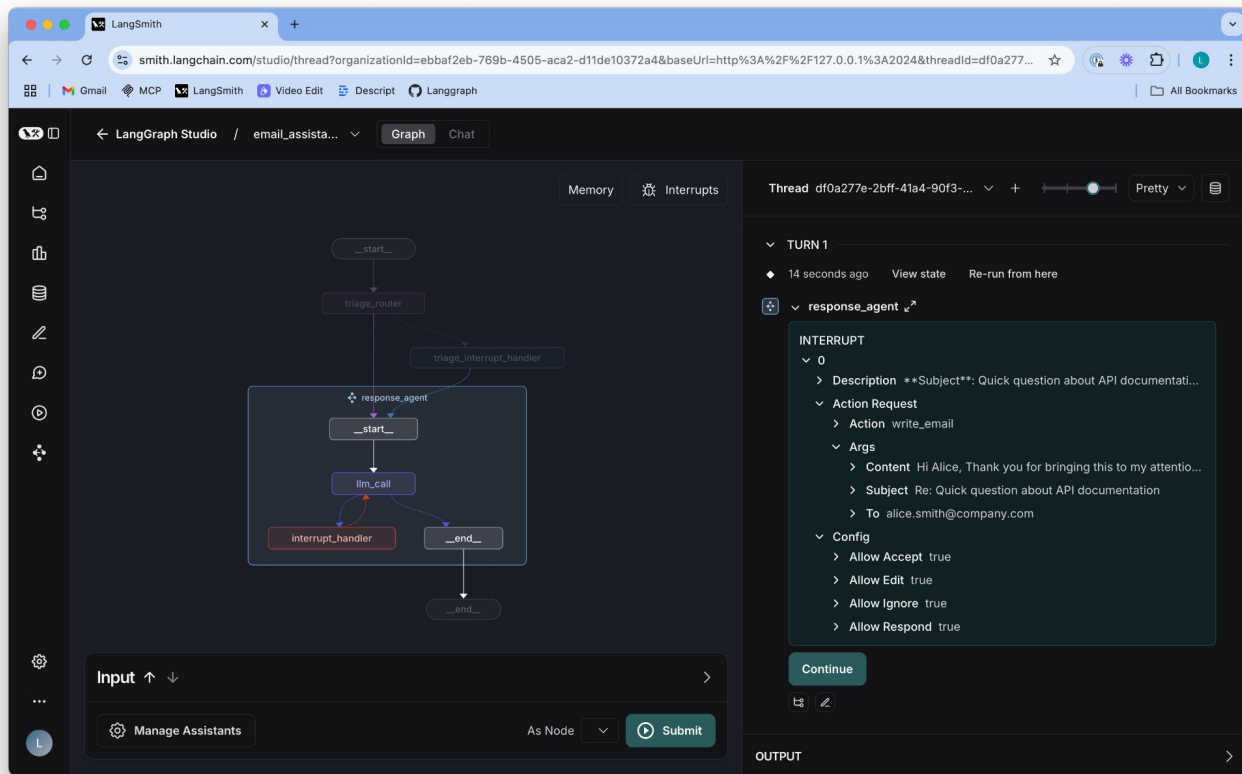
# We'll build an ambient agent that can run your email



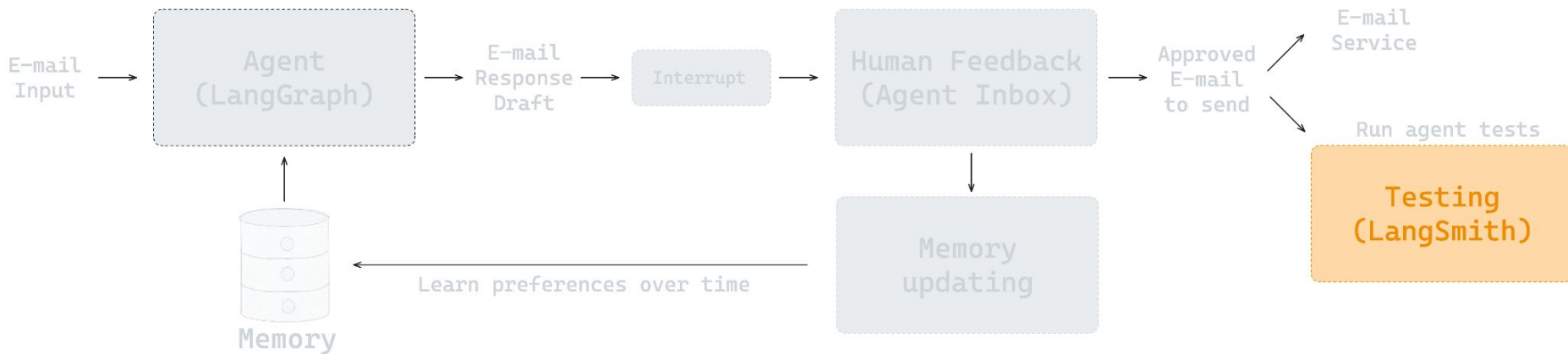
# Building Agents



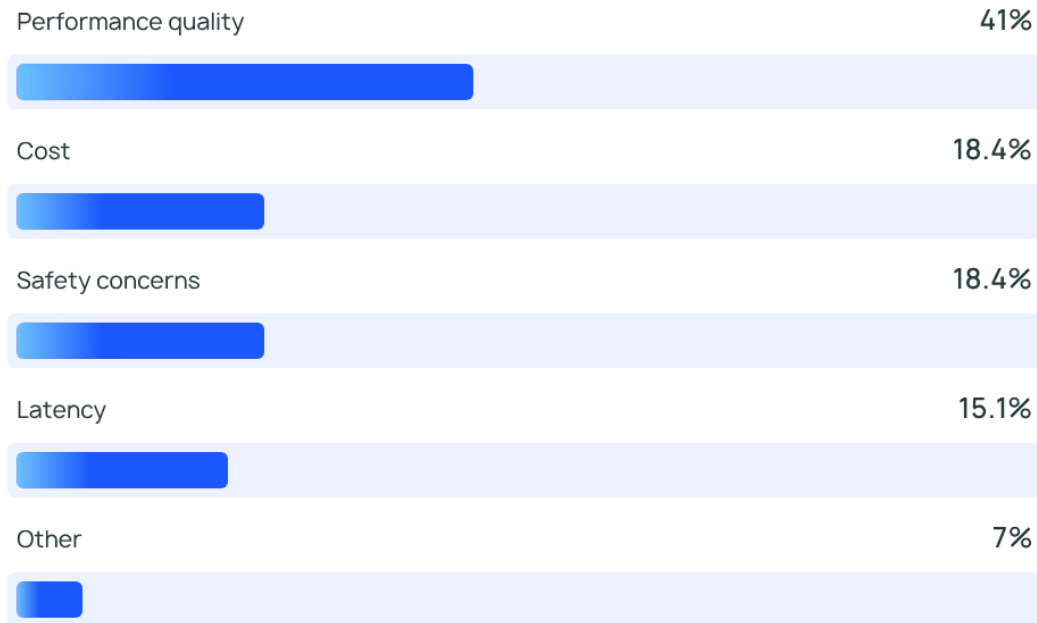
# We'll build this from scratch, and show our custom agent IDE



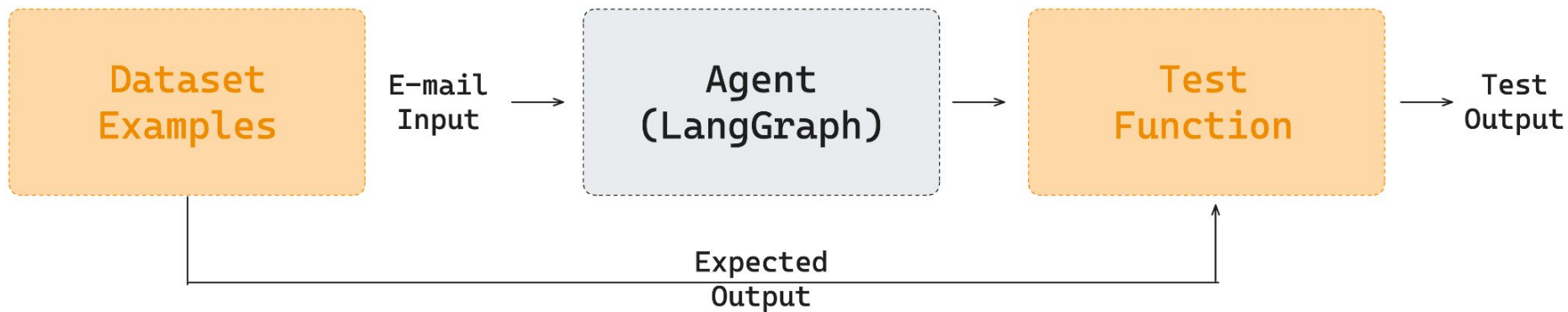
## But, would you just turn this on?



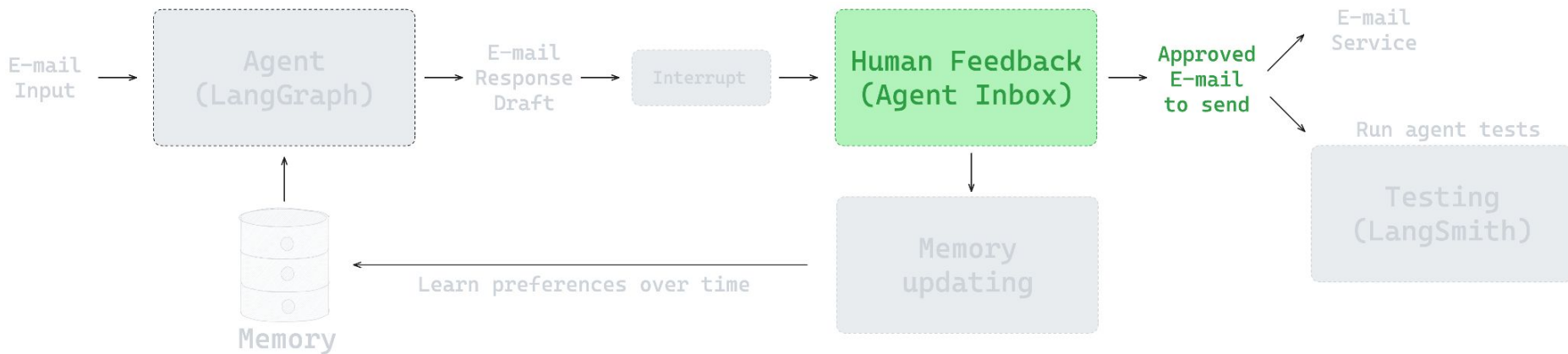
# Building agents is hard



## Use testing to benchmark the email assistant

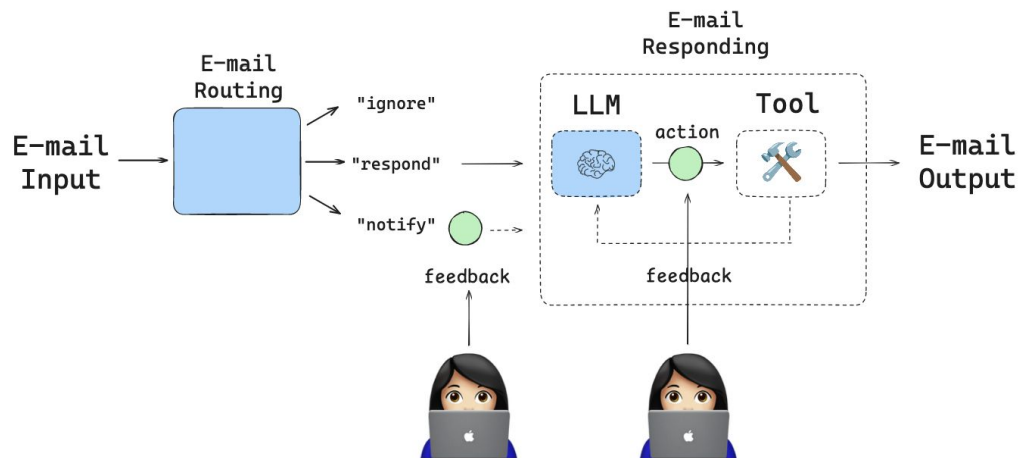


# The cost of error is high, so we want some human oversight!

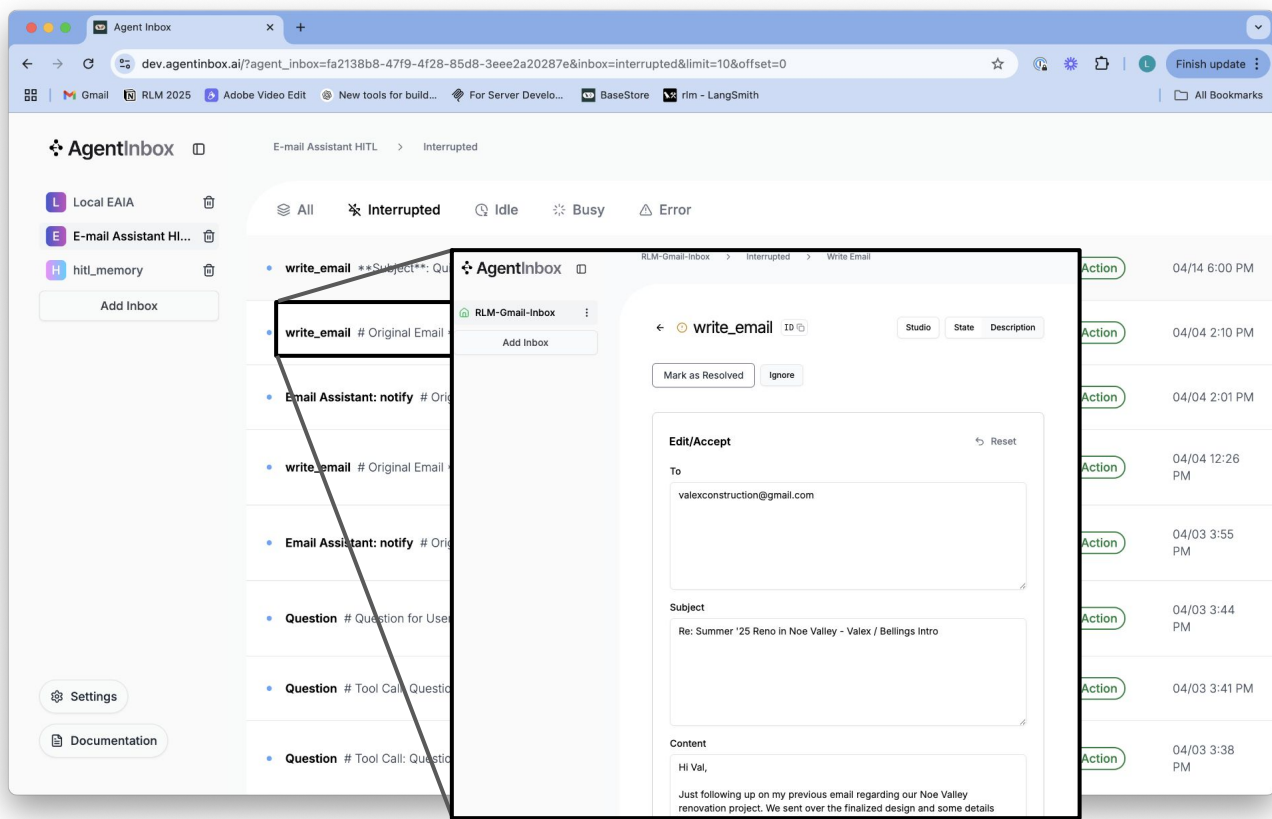




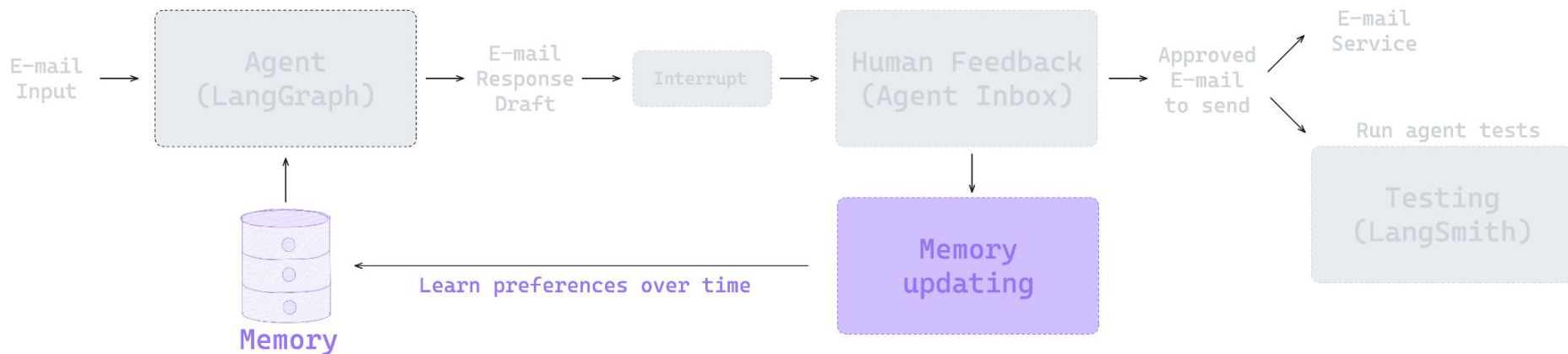
## Use human in the loop to gate sensitive actions



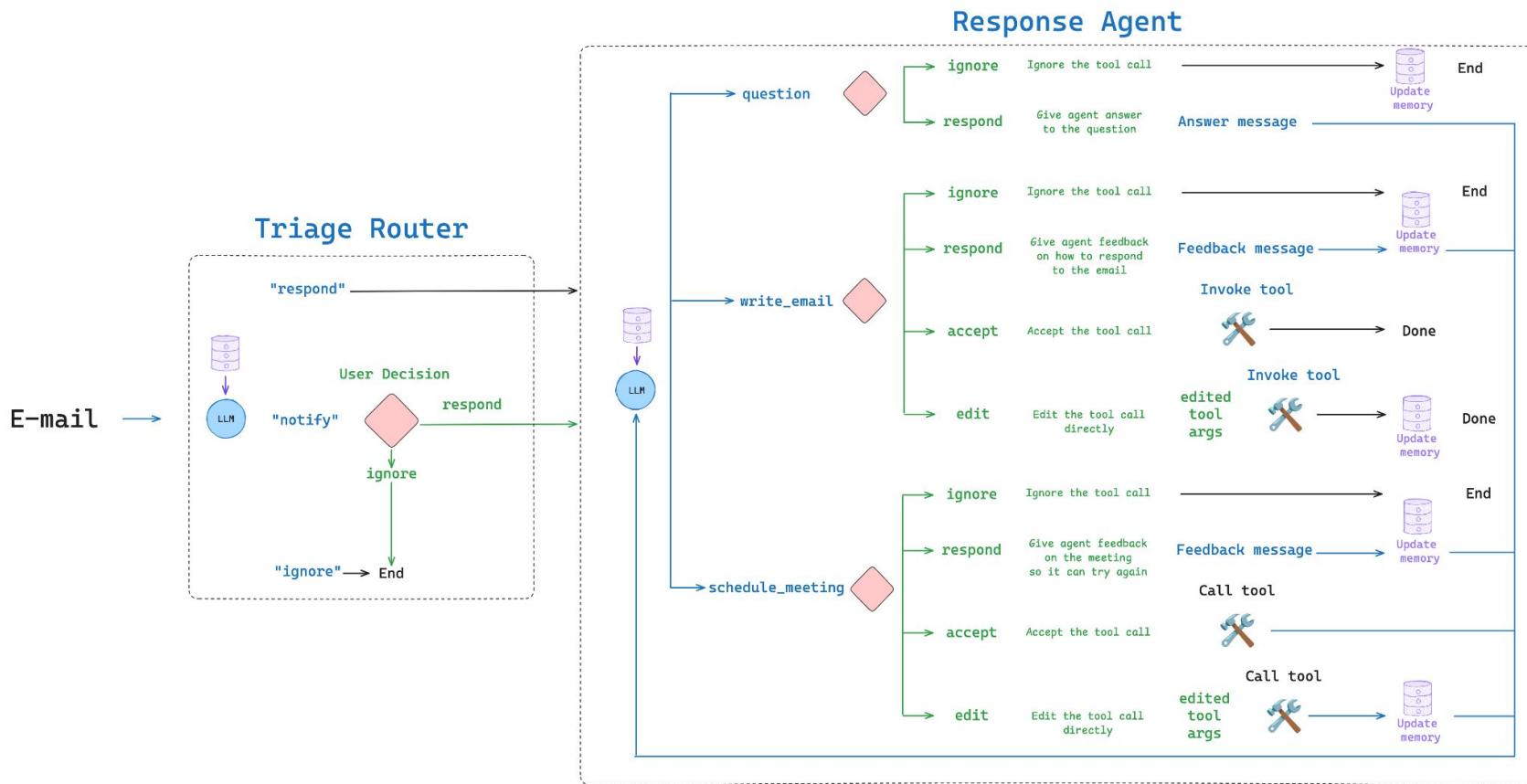
# We'll show Agent Inbox as a lightweight UX for our assistant



# Use memory to learn preferences over time

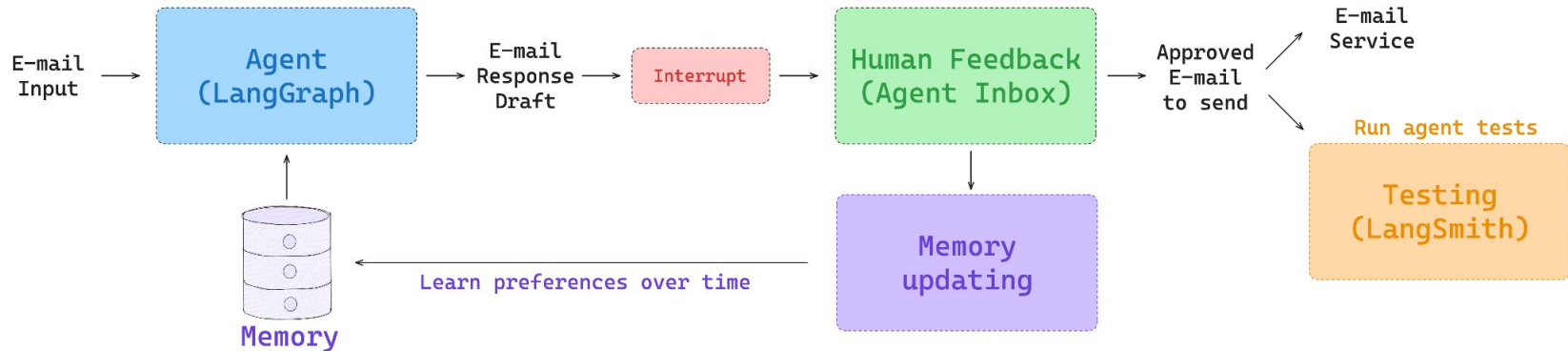


# This will all build towards an email assistant, which we will deploy



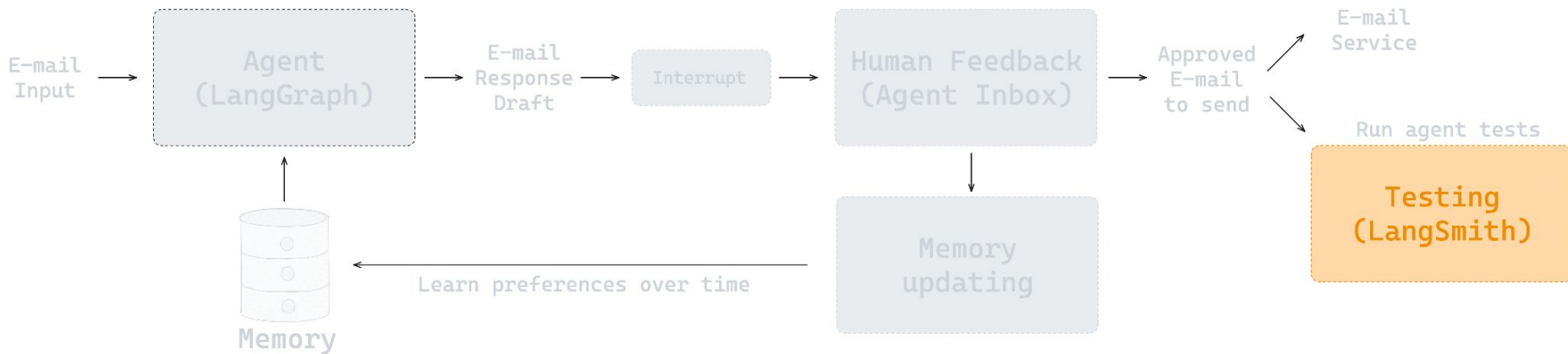
## ... and will showcase some tools for building agents

What we will show	What we will use	Lesson
How to build our email assistant	LangGraph	Notebook 1
How to test our email assistant	LangSmith	Notebook 2
Use threads + human-in-the-loop to pause agent	LangGraph	Notebook 3
Use Agent Inbox as an interface	Agent Inbox	
Use memory to learn preferences over time	LangGraph	Notebook 4
Deploy our agent	LangGraph Platform	

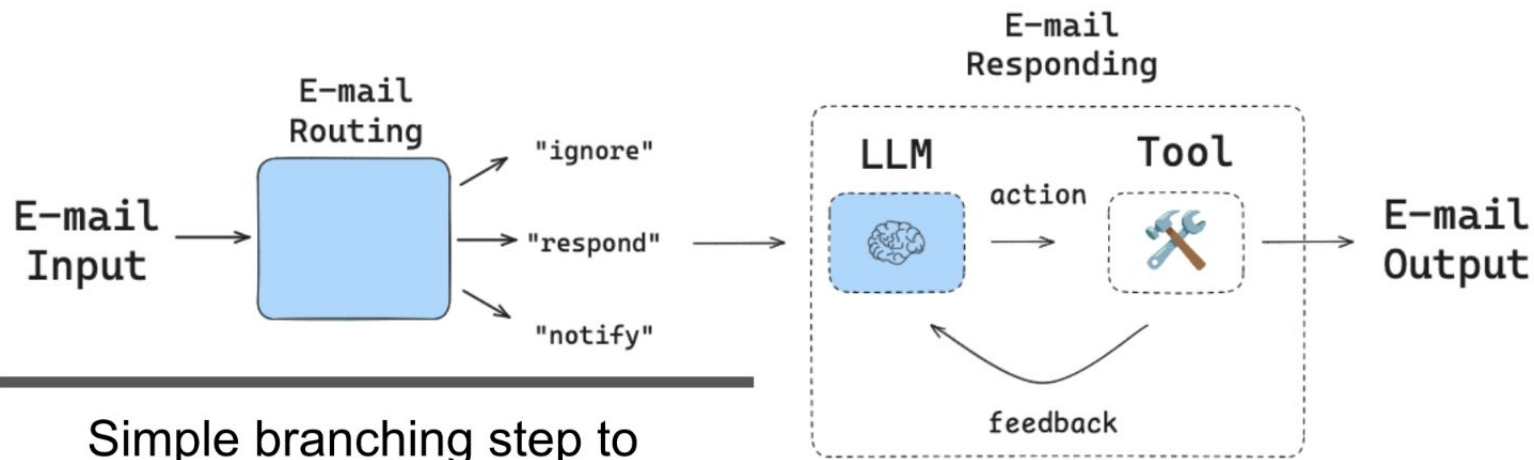


Evals

# Evaluating our Email Assistant



# Building Agents



---

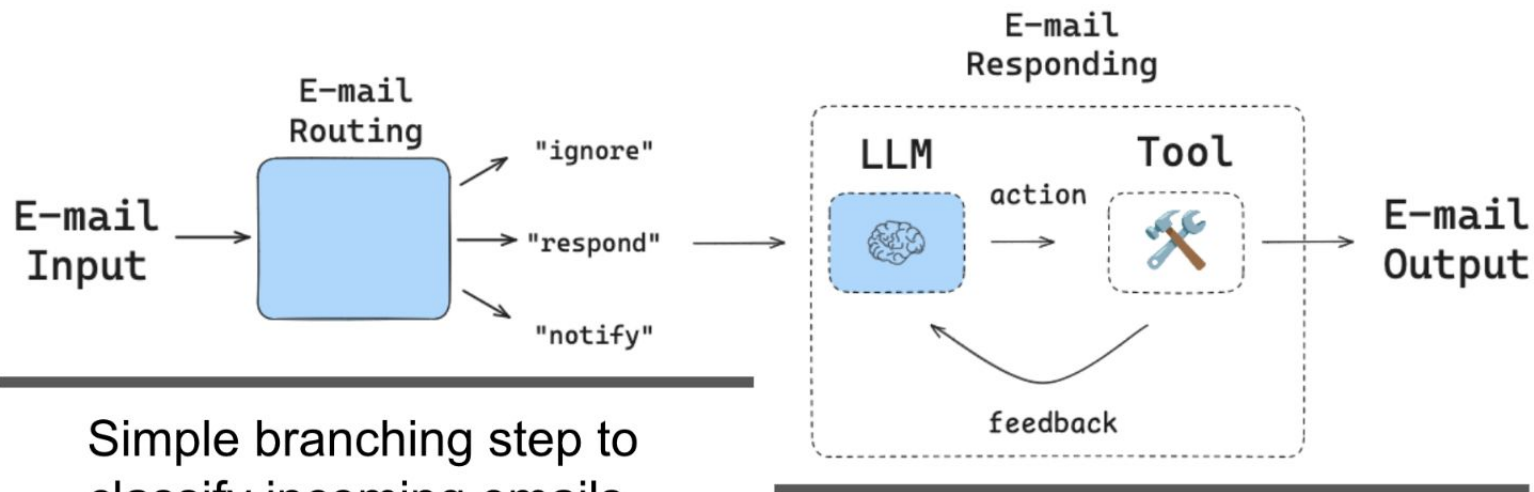
Simple branching step to classify incoming emails, so use a router

---

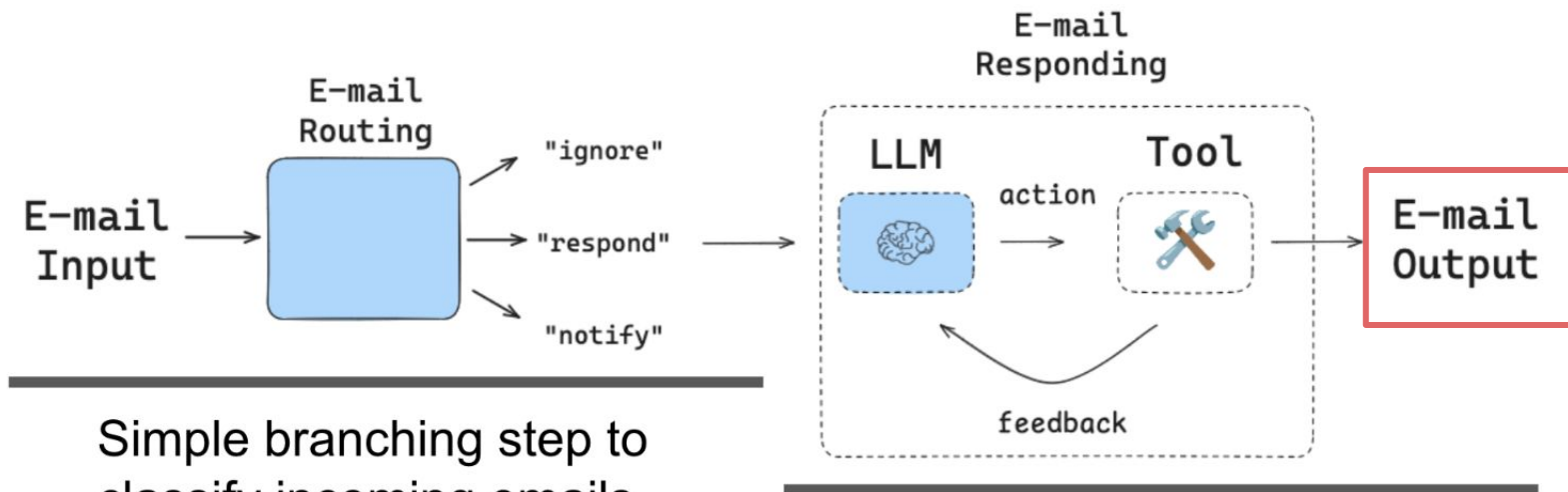
Based on the email, need different tools (schedule mtng, check cal, etc) so use an agent



## Agents need to be evaluated at different levels of granularity

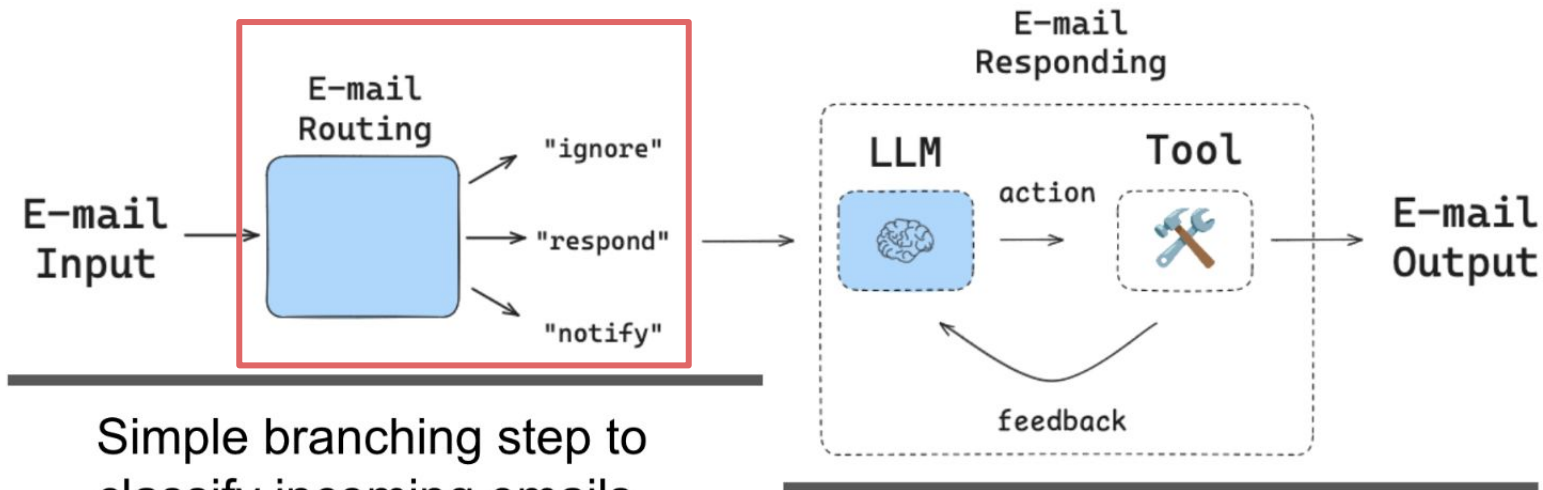


## Agents need to be evaluated at different levels of granularity



**End-to-end: evaluate the agent's final response**

## Agents need to be evaluated at different levels of granularity

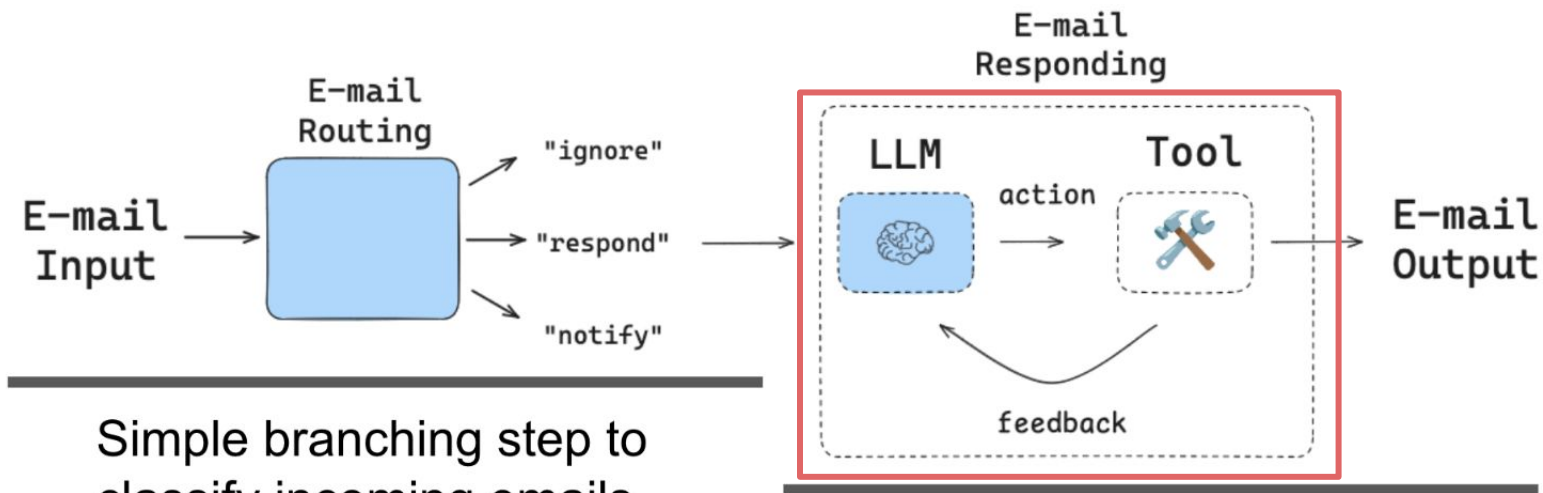


Simple branching step to classify incoming emails, so use a router

Based on the email, need different tools (schedule mtng, check cal, etc) so use an agent

**Triage Unit Test: evaluate a triage decision**

## Agents need to be evaluated at different levels of granularity

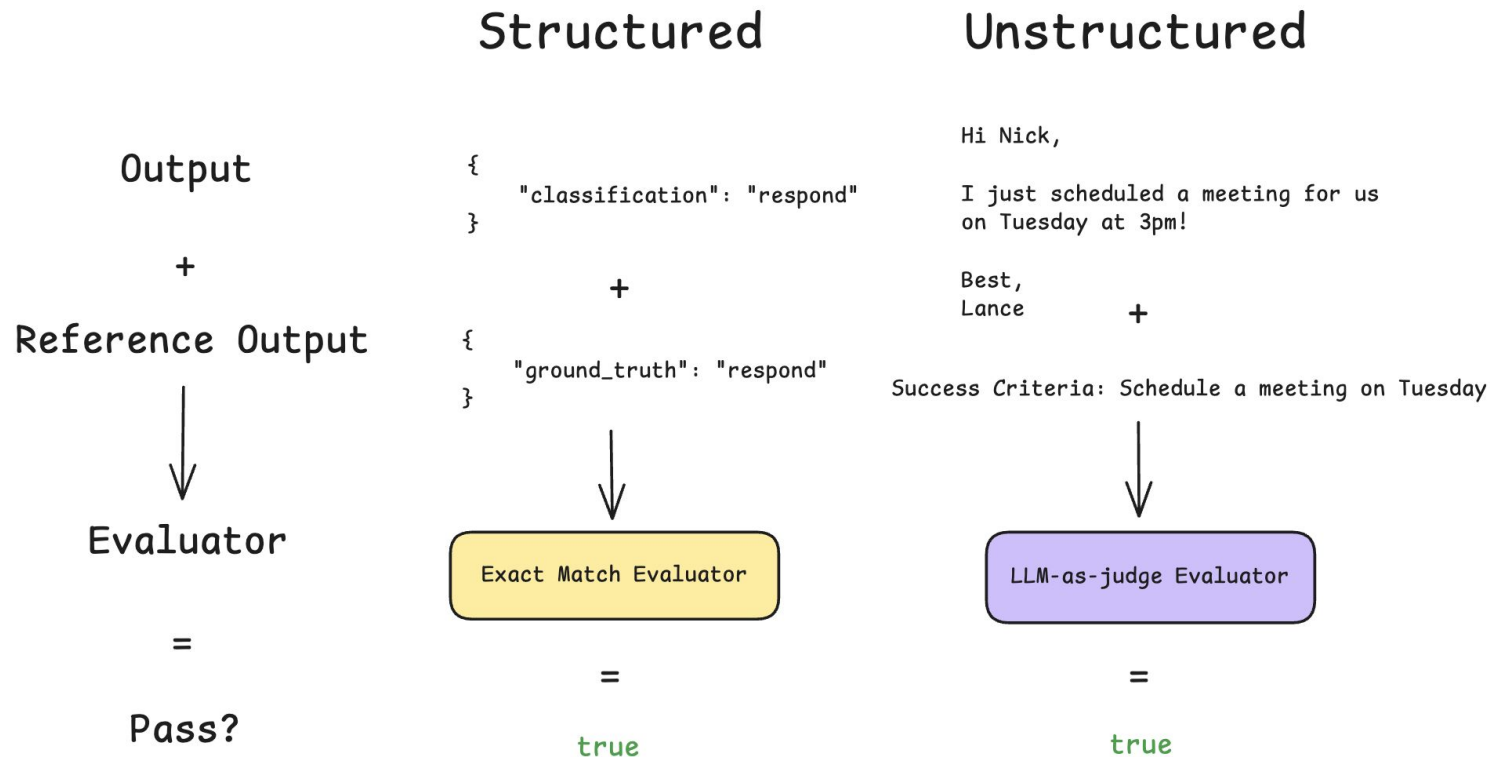


Simple branching step to classify incoming emails, so use a router

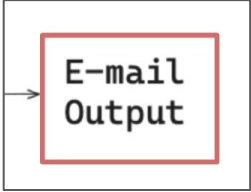
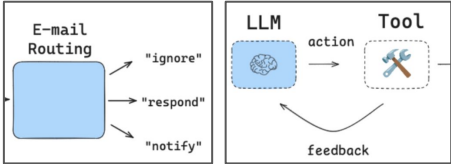
Based on the email, need different tools (schedule mtng, check cal, etc) so use an agent

**Trajectory Unit Test: evaluate the agent's tool call trajectory**

# Evaluating structured vs unstructured output



# Type of evaluations in our Email Assistant

	Structured	Unstructured
End to End		
Unit Test		

Deployment

	Local	Hosted
Run	langgraph dev	Connect GitHub
Threads	Saved locally	Postgres
Store	Picked dict	Postgres
Good for	Prototyping	Production / cron

<https://langchain-ai.github.io/langgraph/tutorials/deployment/>  
[https://langchain-ai.github.io/langgraph/concepts/langgraph\\_platform/](https://langchain-ai.github.io/langgraph/concepts/langgraph_platform/)  
[https://langchain-ai.github.io/langgraph/concepts/application\\_structure/#key-concepts](https://langchain-ai.github.io/langgraph/concepts/application_structure/#key-concepts)