

LangGraph Crash Course #10 - Reflexion Agent - Introduction



Harish Neel | AI
7.72K subscribers

Subscribe

90



Share

Clip



4,335 views Feb 18, 2025 #python #artificialintelligence #langgraph

In this beginner-friendly playlist, I'll show you how to use LangGraph to build powerful AI-agents from scratch

Reflexion Agent System

Reflexion Agents in LangGraph

Recap of what we saw previously:

Reflection Agent System consists of a generator and a reflector component

Although, iteratively making a post better is significantly better than just prompting ChatGPT, the content generated is still not grounded in live data

It could be hallucination or outdated content and we have no way of knowing

Reflexion Agent System address this exact drawback



Reflexion Agents in LangGraph

What is Reflexion Agent System:

The reflexion agent, similar to reflection agent, not only critiques it's own responses but also fact checks it with external data by making API calls (Internet Search)

In the Reflection agent pattern, we had to rely on the training data of LLMs but in this case, we're not limited to that.

Reflexion Agents in LangGraph

What is Reflexion Agent System:

The main component of Reflexion Agent System is the "actor"

The "actor" is the main agent that drives everything - it reflects on it's responses and re-executes.

It can do this with or without tools to improve based on self-critique that is grounded in external data

It's main sub-components include:

1. Tools/tool execution
2. Initial responder: generate an initial response & self-reflection
3. Revisor: re-respond & reflect based on previous reflections



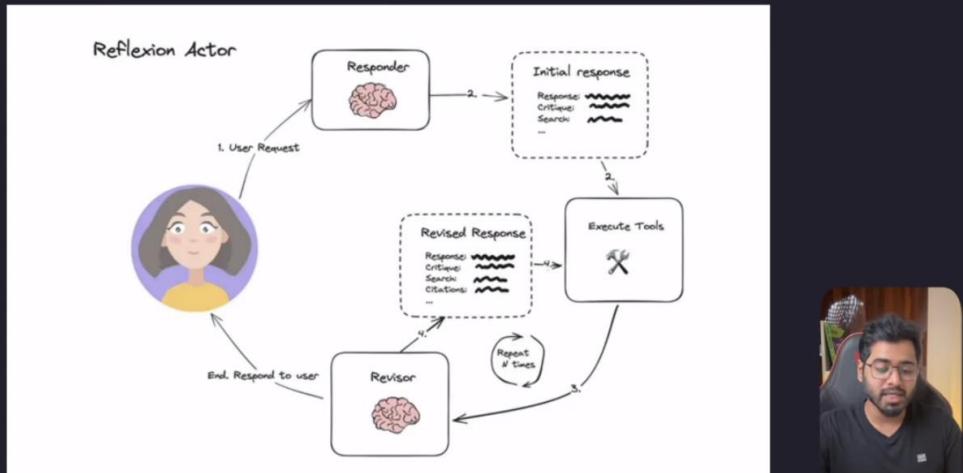
Reflexion Agents in LangGraph

Episodic memory

In the context of Reflexion agents, episodic memory refers to an agent's ability to recall specific past interactions, events, or experiences, rather than just generalized knowledge.

This is crucial for making agents feel more context-aware, personalized, and human-like over time.

Reflexion Agent System



LangGraph Crash Course #11 - Reflexion Agent - Building Responder Chain



Harish Neel | AI
7.72K subscribers

Subscribe

Like 92 Share Clip Save ...

4,652 views Feb 19, 2025 #python #artificialintelligence #langgraph

In this beginner-friendly playlist, I'll show you how to use LangGraph to build powerful AI-agents from scratch

EXPLORER

- LANGGRAPH
 - 1_Introduction
 - react_agent_basic.py
 - 2_basic_reflection_system
 - basic.py
 - chains.py
 - 3_reflexion_agent_system
 - chains.py
- venv
- .env
- .gitignore

chains.py U X

```
3_reflexion_agent_system > chains.py > ...
1 from langchain.prompts import ChatPromptTemplate, MessagesPlaceholder
2 import datetime
3
4 # Actor Agent Prompt
5 actor_prompt_template = ChatPromptTemplate.from_messages(
6 [
7     {
8         "system",
9         """You are expert AI researcher.
10 Current time: {time}
11
12 1. {first_instruction}
13 2. Reflect and critique your answer. Be severe to maximize improvement.
14 3. After the reflection, **list 1-3 search queries separately** for
researching improvements. Do not include them inside the reflection.
15 """
16     ],
17     MessagesPlaceholder(variable_name="messages"),
18     ("system", "Answer the user's question above using the required
format."),
19 ]
20 )
```

This is how we build the **Responder** agent and it's expected output format in the Reflexion Actor above.

LLM Response Parser System

The system converts unstructured LLM outputs into well-defined Python objects through a series of structured parsing steps, ensuring data validation and consistent formatting.

What are the key components?

1. Chat Prompt Template
2. Function Calling with Pydantic Schema
3. Pydantic Parser



We can then ground and convert the initial response into a structured format/schema as below.

LLM Response Parser System

2. Function Calling with Pydantic Schema

Function calling:

Similar to how we make tools available to the LLM, we can also send a schema to the LLM and force it to structure its JSON output according to the schema

Pydantic:

A Python library that defines data structures using classes

Provides automatic validation of JSON data against these class definitions



We will define a schema and then add it as a tool for the LLM to call and use to structure the response as we want

LLM Response Parser System

3. Pydantic Parser

Takes the JSON output from the LLM's function call

Validates it against the defined Pydantic schema (class definition)

Creates instances of Pydantic classes with the validated data

If the LLM's output does not match with the defined schema, it will throw an error



This screenshot shows a code editor interface with two tabs open: `chains.py` and `schema.py`. The `chains.py` tab contains Python code defining classes for reflection and answer questions. The `schema.py` tab is currently active.

```

3_reflexion_agent_system > schema.py > ...
1 from pydantic import BaseModel, Field
2 from typing import List
3
4
5 class Reflection(BaseModel):
6     missing: str = Field(description="Critique of what is missing.")
7     superfluous: str = Field(description="Critique of what is
8     superfluous")
9
10 class AnswerQuestion(BaseModel):
11     """Answer the question."""
12
13     answer: str = Field(
14         description="~250 word detailed answer to the question.")
15     search_queries: List[str] = Field(
16         description="1-3 search queries for researching improvements to
17         address the critique of your current answer."
18     )
19     reflection: Reflection = Field(
20         description="Your reflection on the initial answer.")

```

This is how we further ground the response of the Responder LLM's response. We are going to ask for what is missing and what is unneeded or superfluous.



This screenshot shows a code editor interface with two tabs open: `chains.py` and `schema.py`. The `chains.py` tab contains Python code importing ChatPromptTemplate, MessagesPlaceholder, ChatOpenAI, and AnswerQuestion from langchain and schema respectively. It defines an actor prompt template with a system message and a user message.

```

3_reflexion_agent_system > chains.py > ...
1 from langchain.prompts import ChatPromptTemplate, MessagesPlaceholder
2 import datetime
3 from langchain_openai import ChatOpenAI
4 from schema import AnswerQuestion
5
6 # Actor Agent Prompt
7 actor_prompt_template = ChatPromptTemplate.from_messages(
8     [
9         (
10             "system",
11             """You are expert AI researcher.
12             Current time: {time}"""
13         )
14     ]
15 )

```

Import ChatOpenAI and the AnswerQuestion as above, then use them as below



This screenshot shows a code editor interface with two tabs open: `chains.py` and `schema.py`. The `chains.py` tab contains Python code defining a first responder prompt template that partials the actor prompt template with a current time and a first instruction, and then binds it to a ChatOpenAI language model with an AnswerQuestion tool.

```

3_reflexion_agent_system > chains.py > ...
17     ),
18     MessagesPlaceholder(variable_name="messages"),
19     ("system", "Answer the user's question above using the required
20     format."),
21 ]
22 ).partial(
23     time=lambda: datetime.datetime.now().isoformat(),
24 )
25
26 first_responder_prompt_template = actor_prompt_template.partial(
27     first_instruction="Provide a detailed ~250 word answer"
28 )
29
30 llm = ChatOpenAI(model="gpt-4o")
31
32 first_responder_chain = first_responder_prompt_template | llm.bind_tools
33     [tools=[AnswerQuestion], tool_choice='AnswerQuestion']

```

The use of `tool_choice` above is going to force the LLM to call and use the AnswerQuestion tool to structure its response.

```
from langchain.prompts import ChatPromptTemplate, MessagesPlaceholder
import datetime
from langchain_openai import ChatOpenAI
from schema import AnswerQuestion
from langchain_core.output_parsers.openai_tools import PydanticToolsParser
from langchain_core.messages import HumanMessage
pydantic_parser = PydanticToolsParser(tools=[AnswerQuestion])
# Actor Agent Prompt
actor_prompt_template = ChatPromptTemplate.from_messages(
```

We also need to validate the response and use OpenAI's tool parser that will know how to extract the information

```
time=lambda: datetime.datetime.now().isoformat(),
)
first_responder_prompt_template = actor_prompt_template.partial(
    first_instruction="Provide a detailed ~250 word answer"
)
llm = ChatOpenAI(model="gpt-4o")
first_responder_chain = first_responder_prompt_template | llm.bind_tools
(tools=[AnswerQuestion], tool_choice='AnswerQuestion') | pydantic_parser
response = first_responder_chain.invoke([
    "messages": [HumanMessage(content="Write me a blog post on how small
business can leverage AI to grow")]
])
print(response)
```

After parsing, we can now invoke this chain with a HumanMessage as the prompt

```
/Users/harishb/Desktop/LangGraph/venv/bin/python /Users/harishb/Desktop/LangGraph/3_reflexion_agent_system/chains.py
(venv) (base) harishb@Harishb-MacBook-Air LangGraph % /Users/harishb/Desktop/LangGraph/venv/bin/python /Users/harishb/Desktop/LangGraph/3_reflexion_agent_system/chains.py
[AnswerQuestion(answer="In today's fast-paced digital landscape, small businesses are increasingly turning to artificial intelligence (AI) to streamline operations, understand customers better, and chart a path for growth. While AI might seem daunting due to perceptions of high costs and complexity, advancements in technology have made it accessible to even the smallest companies.\n\nOne of the key areas where AI can be a boon for small businesses is in customer service. AI chatbots can handle queries 24/7, providing immediate responses and significantly improving customer satisfaction. These tools not only reduce the burden on human staff but also allow businesses to gather valuable data on customer preferences and behavior.\n\nMarketing is another arena where AI shines. By utilizing AI-driven analytics, businesses can gain insights into market trends and consumer habits, tailoring their marketing strategies accordingly. For instance, AI algorithms can predict customer needs and automate content ideas, ensuring that marketing efforts resonate with the target audience.\n\nFurthermore, AI helps in optimizing inventory management. Predictive analytics can forecast inventory needs, helping businesses maintain optimal stock levels, reduce waste, and ensure that popular products are always available.\n\nAI also plays a crucial role in financial forecasting. AI-driven tools can analyze historical data to predict cash flow trends, helping businesses make informed decisions about budgeting and expansion.\n\nIn summary, although initially intimidating, AI offers an array of tools that can help small businesses improve efficiency, engage with their customer base more effectively, and ultimately grow in the competitive marketplace.", search_queries=['how small businesses use AI', 'AI tools for small business growth', 'AI benefits for small business'], reflection=Reflection(missing='The current answer lacks specific examples of AI tools or services that small businesses can use,')
```



The screenshot shows a code editor interface with two tabs open: 'chains.py' and 'schema.py'. The 'schema.py' tab is active, displaying Python code for a Pydantic model named 'AnswerQuestion'. The code defines three fields: 'answer', 'search_queries', and 'reflection'. The 'answer' field is a string with a description of being a ~250 word detailed answer to the question. The 'search_queries' field is a list of strings for researching improvements. The 'reflection' field is a reflection on the initial answer. Below the code editor, there is a large text block containing a summary of AI's role in business forecasting and marketing, followed by a section about AI's role in customer service. At the bottom right of the screen, there is a video call window showing a person speaking.

```
class AnswerQuestion(BaseModel):
    answer: str = Field(
        description="~250 word detailed answer to the question.")
    search_queries: List[str] = Field(
        description="1-3 search queries for researching improvements to address the critique of your current answer."
    )
    reflection: Reflection = Field(
        description="Your reflection on the initial answer.")
```



This screenshot is identical to the one above, showing the same code editor with 'schema.py' active and the same explanatory text below. The video call window at the bottom right is also present, showing the same person speaking.

```
class AnswerQuestion(BaseModel):
    answer: str = Field(
        description="~250 word detailed answer to the question.")
    search_queries: List[str] = Field(
        description="1-3 search queries for researching improvements to address the critique of your current answer."
    )
    reflection: Reflection = Field(
        description="Your reflection on the initial answer.")
```

We can see that we now have the response in the Pydantic model format we wanted with the `AnswerQuestion`, `search_queries`, `reflection` and `superfluous` sections correctly stated.

← → ⌛ ⌂ smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects?paginationState=%7B"pageIndex":%3A0%2... ☆ ⚡ 🔍 🗃 | ↴ ↵ ⌂ New Chrome available : |

All Bookmarks

1. New key created:
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

2. Install dependencies

1 pip install -U langchain langchain-openai Copy

3. Configure environment to connect to LangSmith.

Project Name
pr-minty-councilperson-79

```
1 LANGSMITH_TRACING=true
2 LANGSMITH_ENDPOINT="https://api.smith.langchain.com"
3 LANGSMITH_API_KEY="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"
4 LANGSMITH_PROJECT="pr-minty-councilperson-79"
5 OPENAI_API_KEY=<your-openai-api-key>
```

4. Run any LLM, Chat model, or Chain. Its trace will be sent to this project.

```
1 from langchain_openai import ChatOpenAI
2
3 llm = ChatOpenAI()
4 llm.invoke("Hello, world!")
```

EXPLORER

- LANGGRAPH
 - 1_Introduction
 - react_agent_basic.py
 - 2_basic_reflection_system
 - > __pycache__
 - basic.py
 - chains.py
 - 3_reflexion_agent_system
 - > __pycache__
 - chains.py
 - schema.py
 - venv
- .env
- .gitignore

chans.py U **schema.py U** **.env**

```
1 GOOGLE_API_KEY=AIzaSyAG1YLD2BXS-BS4vfNhNk5nb0oSm_nbwXw
2 TAVILY_API_KEY=tavy-dev-CP1lbnSRxALr4URiCygivzegXB5b42B4
3 OPENAI_API_KEY="sk-proj-8UnYCpCKWnHeJxoY84PuPn92vlyeXLtt6HwdFZeheALEwbDh-rr
G-rk2lqzxIo3cZInNvy18t3BlbkFJB1vKez0RfpTQJuBggg9aQKe64j4tkB9LrovUx-2aLEM
OnId9Pr0b0EMBC8CXCnbfBsohNdowA"
4 LANGSMITH_TRACING=true
5 LANGSMITH_ENDPOINT="https://api.smith.langchain.com"
6 LANGSMITH_API_KEY="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"
7 LANGSMITH_PROJECT="pr-minty-councilperson-79"
8 |
```

PROBLEMS **OUTPUT** **DEBUG CONSOLE** **TERMINAL** **PORTS**

businesses improve efficiency, engage with their customer base more effectively, and ultimately grow in the competitive marketplace.", search_queries='how small businesses use AI', 'AI tools for small business growth', 'AI benefits for small business', reflection=Reflection(missing='The current answer lacks specific examples of AI tools or services that small businesses can use, such as particular chatbot platforms or marketing analytics tools. It also does not address potential challenges or considerations small businesses might face when implementing AI, like cost and expertise.', superfluous='The section on AI's role in customer service might be overly detailed in regards to chatbots, while the areas of marketing and financial forecasting are under-represented, leading to an imbalance in parts of the explanation.'))

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The left sidebar displays a file tree under the 'EXPLORER' tab, showing a folder named 'LANGGRAPH' containing files like '1_Introduction.py', '2_basic_reflection_system.py', and '3_reflexion_agent_system.py'. The main area shows code snippets for 'chains.py' and 'schema.py'. The bottom right corner features a video overlay of a man with glasses and a beard, wearing a black t-shirt, sitting in front of a microphone. The terminal at the bottom shows a large block of text about AI's impact on small businesses.

```
3_reflexion_agent_system > chains.py > ...
25     ]
26     ).partial(
27         time=datetime.datetime.now().isoformat(),
28     )
29
30     first_responder_prompt_template = actor_prompt_template.partial(
31         first_instruction="Provide a detailed ~250 word answer"

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python + □ ☰ ⌂ ⌃ ⌄ ⌅

```
l businesses looking to enhance their operations and scale effectively. AI provides numerous opportunities to automate repetitive tasks, gain deeper customer insights, and improve decision-making processes, all of which can lead to significant growth.\nOne of the fundamental ways small businesses can harness AI is through customer service automation. Tools like chatbots and virtual assistants can handle basic customer inquiries 24/7, freeing up human resources to tackle more complex tasks. This not only improves efficiency but also enhances customer satisfaction by providing immediate responses.\nMarketing is another area where AI can have a huge impact. AI-driven analytics can help small businesses understand customer behavior better, enabling them to tailor their marketing strategies with precision. Through AI, businesses can harness predictive analytics to forecast trends and adapt their strategies accordingly, ensuring they stay ahead of their competitors.\nOperational efficiency is crucial for growth, and AI can be instrumental here as well. Small businesses can automate inventory management, supply chain logistics, and even manage their booking systems more effectively, reducing errors and cutting down on costs.\nIncorporating AI into decision-making processes allows small businesses to analyze large volumes of data quickly, providing them with smarter, data-driven decisions. These decisions can open up new opportunities and identify areas for financial optimization.\nOverall, embracing AI technology can give small businesses a competitive edge, positioning them for sustainable growth in an increasingly digital economy.', search_queries=['AI for small business growth', 'small business AI solutions'], reflection=Reflection(mis could have included more specific examples or case studies of small businesses successfully implemented AI for growth. This would make the benefits more tangible or readers. Additionally, mentioning potential challenges or considerations small businesses m
```

Tracing projects								
Search by name...		Columns		Actions				
Name	Feedback (7D)	Run Count (7D)	Error Rate (7D)	P50 Latency (7D)	P99 Latency (7D)	% Streaming (7D)	Total Tokens (7D)	Total C
pr-mint...	1	0%	7.13s	7.13s	7.13s	0%	673	\$0.005
pr-eld...	43	65%	2.56s	32.65s	32.65s	0%	17,629	\$0.100
pr-dow...	2	0%	3.83s	3.92s	3.92s	0%	770	\$0.004
pr-bum...	5	60%	1.24s	9.02s	9.02s	0%	1,105	\$0.005

Personal > Tracing projects > pr-minty-councilperson-79

pr-minty-councilperson-79

Runs Threads Monitor Setup

1 filter Last 7 days Root Runs LLM Calls All Runs

Name	Input	Output	Error	Start Time	End Time
RunnableSequence	human: Write me a blog post on how small business can benefit from AI	[{"answer": "Leve..."}]		18/02/2025, 13:43:45	18/02/2025, 13:43:45

Stats

Last 7 days

RUN COUNT
1

TOTAL TOKENS
673 / \$0.01

MEDIAN TOKENS
673

ERROR RATE
0%

% STREAM
0%

LATENCY
P50: 7.13s

Filter

Input Key

smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects/p/9ed8e2e3-4ed6-42f6-9f34-dbe45900bb... | New Chrome available | All Bookmarks

pr-minty-councilperson-79

Runs Threads Monitor Setup

1 filter Last 7 days Root Runs LLM Calls

Name Input

RunnableSequence human: Write me a b

TRACE

RunnableSequence 7.13s

gpt-4o 7.02s

PydanticToolsParser 0.00s

RunnableSequence

Run Feedback Metadata

Input

HUMAN

Write me a blog post on how small business can leverage AI to grow

Output

Output 0

- Answer Leveraging artificial intelligence (AI) c...
- Reflection
 - Missing The answer could have included m...
 - Superfluous The mention of AI-driven cust...
- Search Queries
 - 0 AI for small business growth
 - 1 AI tools for small businesses
 - 2 small business AI solutions

Run ID Trace ID

START TIME 02/18/2025, 01:43:45 PM

END TIME 02/18/2025, 01:43:52 PM

TIME TO FIRST TOKEN N/A

STATUS Success

TOTAL TOKENS 67

LATENCY 7.13s

TYPE Sequence

We can also see the traces stated as expected

smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects/p/9ed8e2e3-4ed6-42f6-9f34-dbe45900bb... | New Chrome available | All Bookmarks

pr-minty-councilperson-79

Runs Threads Monitor Setup

1 filter Last 7 days Root Runs LLM Calls

Name Input

RunnableSequence human: Write me a b

TRACE

RunnableSequence 7.13s

gpt-4o 7.02s

PydanticToolsParser 0.00s

RunnableSequence

Run Feedback Metadata

Output

0

Answer

Leveraging artificial intelligence (AI) can be a game-changer for small businesses looking to enhance their operations and scale effectively. AI provides numerous opportunities to automate repetitive tasks, gain deeper customer insights, and improve decision-making processes, all of which can lead to significant growth.

One of the fundamental ways small businesses can harness AI is through customer service automation. Tools like chatbots and virtual assistants can handle basic customer inquiries 24/7, freeing up human resources to tackle more complex tasks. This not only improves efficiency but also enhances customer satisfaction by providing immediate responses.

Marketing is another area where AI can have a huge impact. AI-driven analytics can help small businesses understand customer behavior better, enabling them to tailor their marketing strategies with precision. Through AI, businesses can harness predictive analytics to forecast trends and adapt their strategies accordingly, ensuring they stay

Run ID Trace ID

STATUS Success

TOTAL TOKENS 673 tokens / \$0.00508

LATENCY 7.13s

TYPE Sequence

smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects/p/9ed8e2e3-4ed6-42f6-9f34-dbe45900bb... | New Chrome available : |

Personal > Tracing projects > pr-minty-councilperson-79

Runs Threads Monitor Setup

1 filter Last 7 days Root Runs LLM Calls

Name Input

RunnableSequence human: Write me a t

TRACE

RunnableSequence 7.13s

gpt-4o 7.02s

PydanticToolsParser 0.00s

RunnableSequence

Run Feedback Metadata

Overall, embracing AI technologies can give small businesses a competitive edge, positioning them for sustainable growth and success in an increasingly digital economy.

Reflection

Missing

The answer could have included more specific examples or case studies of small businesses that have successfully implemented AI for growth. This would make the benefits more tangible and relatable for readers. Additionally, mentioning potential challenges or considerations small businesses might face when integrating AI, such as costs or technical expertise required, could provide a more rounded perspective.

Superfluous

The mention of AI-driven customer service and operational efficiency, while important, could be considered somewhat generic as these are commonly discussed benefits. More emphasis on unique or lesser-known AI applications, or emerging AI trends that small businesses might capitalize on, would add value to the response.

Search Queries

0 AI for small business growth

1 AI tools for small businesses

2

Run ID Trace ID



smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects/p/9ed8e2e3-4ed6-42f6-9f34-dbe45900bb... | New Chrome available : |

Personal > Tracing projects > pr-minty-councilperson-79

Runs Threads Monitor Setup

1 filter Last 7 days Root Runs LLM Calls

Name Input

RunnableSequence human: Write me a t

TRACE

RunnableSequence 7.13s

gpt-4o 7.02s

PydanticToolsParser 0.00s

RunnableSequence

Run Feedback Metadata

Missing

The answer could have included more specific examples or case studies of small businesses that have successfully implemented AI for growth. This would make the benefits more tangible and relatable for readers. Additionally, mentioning potential challenges or considerations small businesses might face when integrating AI, such as costs or technical expertise required, could provide a more rounded perspective.

Superfluous

The mention of AI-driven customer service and operational efficiency, while important, could be considered somewhat generic as these are commonly discussed benefits. More emphasis on unique or lesser-known AI applications, or emerging AI trends that small businesses might capitalize on, would add value to the response.

Search Queries

0 AI for small business growth

1 AI tools for small businesses

2

Run ID Trace ID



smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects/p/9ed8e2e3-4ed6-42f6-9f34-dbe45900bb... New Chrome available

Personal > Tracing projects > pr-minty-councilperson-79

Runs Threads Monitor Setup

1 filter Last 7 days Root Runs LLM Calls

Name Input

RunnableSequence human: Write me a blog post on how small business can leverage AI to grow

TRACE

ChatOpenAI

Run Feedback Metadata

AnswerQuestion CALLED

END TIME 02/18/2025, 01:43:52 PM

TIME TO FIRST TOKEN N/A

STATUS Success

TOTAL TOKENS 673 tokens / \$0.00508

LATENCY

Input

SYSTEM

You are expert AI researcher.
Current time: 2025-02-18T13:43:45.663268

- Provide a detailed ~250 word answer
- Reflect and critique your answer. Be severe to maximize improvement.
- After the reflection, **list 1-3 search queries separately** for researching improvements. Do not include them inside the reflection.

HUMAN

Write me a blog post on how small business can leverage AI to grow

SYSTEM

Answer the user's question above using the required format.

smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects/p/9ed8e2e3-4ed6-42f6-9f34-dbe45900bb... New Chrome available

Personal > Tracing projects > pr-minty-councilperson-79

Runs Threads Monitor Setup

1 filter Last 7 days Root Runs LLM Calls

Name Input

RunnableSequence human: Write me a blog post on how small business can leverage AI to grow

TRACE

ChatOpenAI

Run Feedback Metadata

SYSTEM

Answer the user's question above using the required format.

Output

AI

AnswerQuestion

```
call_Duncbzm0iJGkNUS4Wtfc9c4t
{
  "answer": "Leveraging artificial intelligence (AI) can be a game-changer for small businesses looking to enhance their operations and scale effectively. AI provides numerous opportunities to automate repetitive tasks, gain deeper customer insights, and improve decision-making processes, all of which can lead to significant growth.\n\nOne of the fundamental ways small businesses can harness AI is through customer service automation. This involves using AI-powered chatbots and virtual assistants to handle routine customer inquiries, process orders, and provide personalized recommendations. By automating these interactions, businesses can not only save on labor costs but also improve response times and satisfaction levels.\n\nAnother key area where AI can benefit small businesses is in supply chain management. AI-powered tools can help optimize inventory levels, predict demand more accurately, and identify potential supply chain disruptions before they occur. This can lead to reduced waste, lower costs, and improved delivery times.\n\nIn addition to operational efficiency, AI can also help small businesses in marketing and sales. AI-powered tools like chatbots and recommendation engines can analyze customer behavior and preferences to suggest products or services that are most likely to appeal to them. This can lead to higher conversion rates and increased revenue.\n\nOverall, AI has the potential to transform the way small businesses operate, providing them with the tools and resources they need to compete in today's fast-paced market. By embracing AI and integrating it into their operations, small businesses can stay ahead of the curve and achieve greater success in the years to come."
}
```

We are getting the JSON output that is also grounded

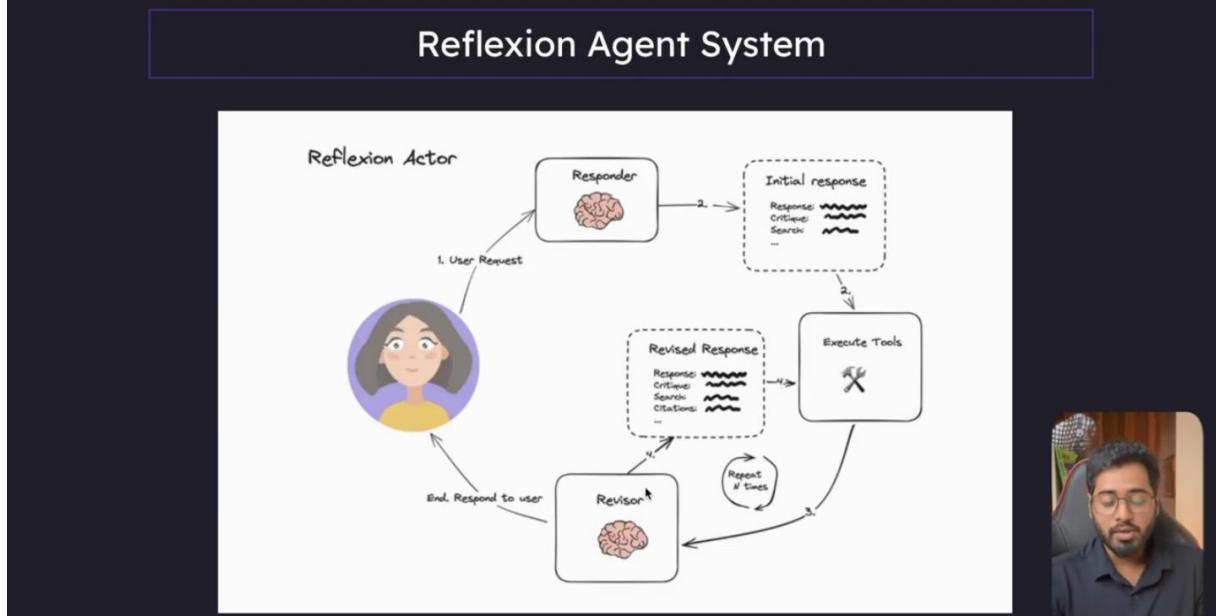
The screenshot shows the LangChain interface for monitoring AI agent interactions. On the left, a sidebar provides navigation and filtering options. The main area displays a trace for the 'PydanticToolsParser' component. The trace details pane shows the input 'human: Write me a business plan' and the output, which is a JSON object containing an AI-generated response. The response includes a code block and several search queries. A video thumbnail of Harish Neel is visible on the right.

Next, let us build the Revisor agent chain in a structured way

LangGraph Crash Course #12 - Reflexion Agent - Building Revisor Chain



2,648 views Feb 23, 2025 #python #artificialintelligence #langgraph
In this beginner-friendly playlist, I'll show you how to use LangGraph to build powerful AI-agents from scratch



EXPLORER

- LANGGRAPH
 - 1_Introduction
 - react_agent_basic.py
 - 2_basic_reflection_system
 - __pycache__
 - basic.py
 - chains.py
 - 3_reflexion_agent_system
 - __pycache__
 - chains.py M
 - schema.py

chains.py M X

```

3_reflexion_agent_system > chains.py > ...
(tools=[AnswerQuestion], tool_choice='AnswerQuestion') | pydantic_parser
37
38 # Revisor section
39
40 revise_instructions = """Revise your previous answer using the new
information.
- You should use the previous critique to add important information to
your answer.
- You MUST include numerical citations in your revised answer to
ensure it can be verified.
- Add a "References" section to the bottom of your answer (which
does not count towards the word limit). In form of:
    - [1] https://example.com
    - [2] https://example.com
- You should use the previous critique to remove superfluous
information from your answer and make SURE it is not more than 250
words.

.....
47
48
49 revisor_chain = actor_prompt_template.partial(
    first_instruction="""

```

EXPLORER

- LANGGRAPH
 - 1_Introduction
 - react_agent_basic.py
 - 2_basic_reflection_system
 - __pycache__
 - basic.py
 - chains.py M
 - schema.py
 - 3_reflexion_agent_system
 - __pycache__
 - chains.py M
 - schema.py

chains.py M

```

words.

.....
47
48
49 revisor_chain = actor_prompt_template.partial(
    first_instruction=""
)
50
51
52
53 response = first_responder_chain.invoke({
54     "messages": [HumanMessage(content="Write me a blog post on how
business can leverage AI to grow")]
55 })
56

```

Next, we need to ground the response of the LLM in the revise structure to include same things plus the citations

EXPLORER

- LANGGRAPH
 - 1_Introduction
 - react_agent_basic.py
 - 2_basic_reflection_system
 - __pycache__
 - basic.py
 - chains.py
 - 3_reflexion_agent_system
 - __pycache__
 - chains.py M
 - schema.py M

chains.py M ● schema.py M X

```

3_reflexion_agent_system > schema.py > ReviseAnswer
9 class AnswerQuestion(BaseModel):
10     """Answer the question."""
11
12     answer: str = Field(
13         description="~250 word detailed answer to the question.")
14     search_queries: List[str] = Field(
15         description="1-3 search queries for researching improvements to
address the critique of your current answer.")
16     reflection: Reflection = Field(
17         description="Your reflection on the initial answer.")
18
19 class ReviseAnswer(AnswerQuestion):
20     """Revise your original answer to your question."""
21
22     references: List[str] = Field(
23         description="Citations motivating your updated answer.")
24
25

```

EXPLORER

- LANGGRAPH
 - 1_Introduction
 - react_agent_basic.py
 - 2_basic_reflection_system
 - __pycache__
 - basic.py
 - chains.py

chains.py M X schema.py M

```

3_reflexion_agent_system > chains.py > ...
1 from langchain.prompts import ChatPromptTemplate, MessagesPlaceholder
2 import datetime
3 from langchain_openai import ChatOpenAI
4 from schema import AnswerQuestion, ReviseAnswer
5 from langchain_core.output_parsers.openai_tools import PydanticToolsParser
6 from langchain_core.messages import HumanMessage
7

```

```

EXPLORER          chains.py M X schema.py M
LANGGRAPH
  1_Introduction
  react_agent_basic.py
  2_basic_reflection_system
    > __pycache__
    basic.py
    chains.py
  3_reflexion_agent_system
    > __pycache__
    chains.py M
    schema.py M
  venv
  .env
  .gitignore

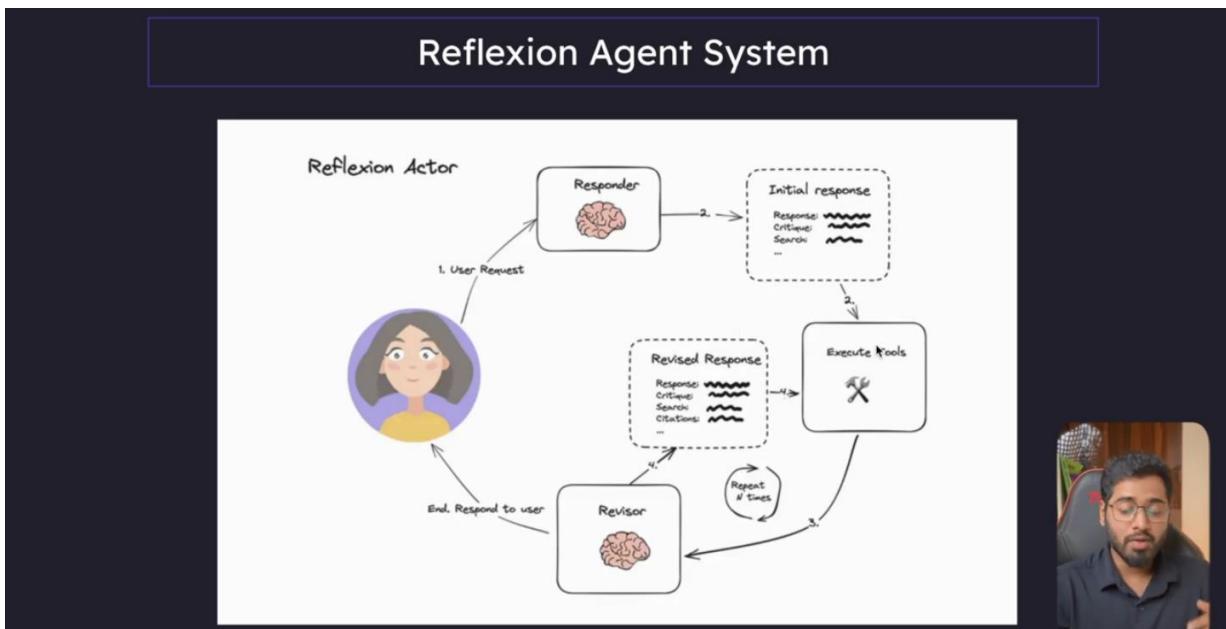
3_reflexion_agent_system > chains.py > ...
ensure it can be verified.
- Add a "References" section to the bottom of your answer (which does not count towards the word limit). In form of:
  - [1] https://example.com
  - [2] https://example.com
- You should use the previous critique to remove superfluous information from your answer and make SURE it is not more than 250 words.

"""
revisor_chain = actor_prompt_template.partial(
    first_instruction=revise_instructions
) | llm.bind_tools[tools=[ReviseAnswer], tool_choice="ReviseAnswer"]

response = first_responder_chain.invoke({
    "messages": [HumanMessage(content="Write me a blog post on how small business can leverage AI to grow")]
})

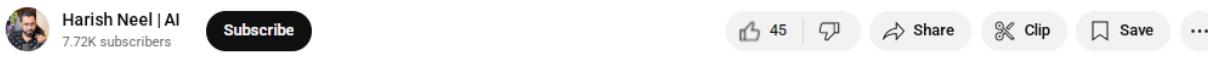
print(response)

```



Next, we will start building out the 3rd component **Execute Tools** before connecting it all together using LangGraph

LangGraph Crash Course #13 - Reflexion Agent - Tool Execution Component



2,247 views Apr 2, 2025 #python #langgraph #artificialintelligence
In this beginner-friendly playlist, I'll show you how to use LangGraph to build powerful AI-agents from scratch

Execute Tools method needs to inspect the last message which is the AIMessage and look at the search terms inside it, loop through each search term and formulate one unified tool message with all the information.



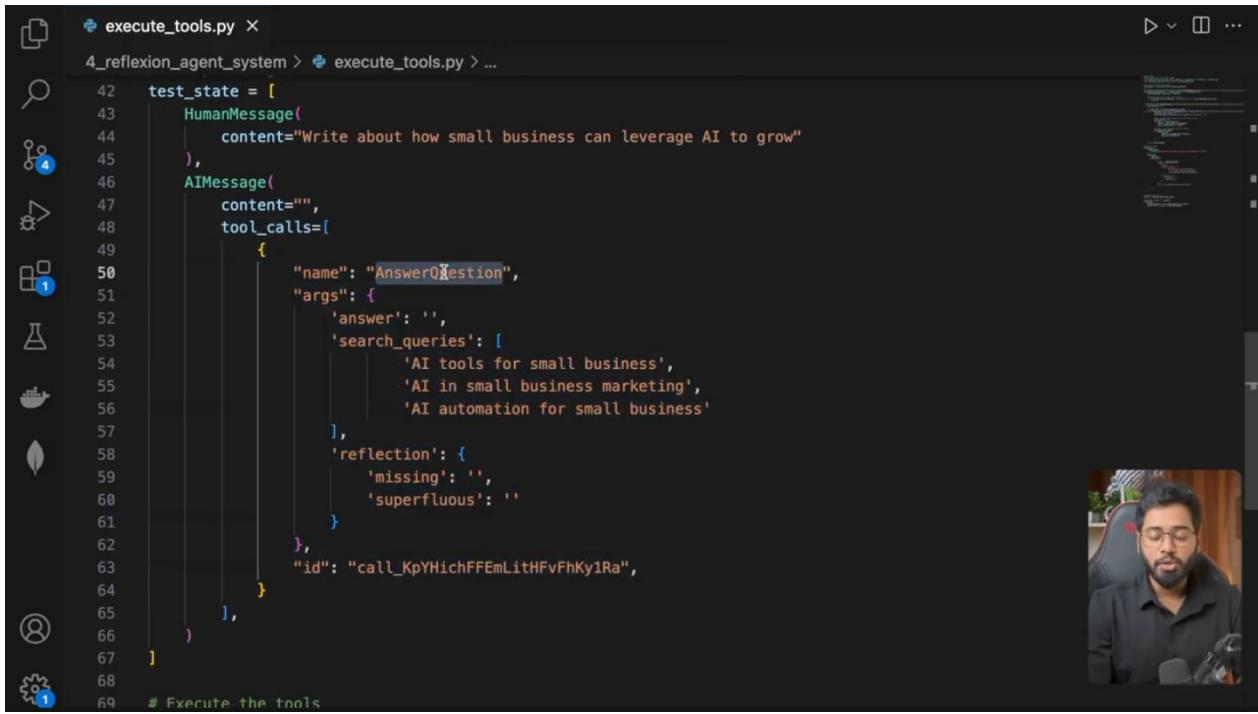
```
execute_tools.py X
4_reflexion_agent_system > execute_tools.py > ...
1 import json
2 from typing import List, Dict, Any
3 from langchain_core.messages import AIMessage, BaseMessage, ToolMessage, HumanMessage
4 from langchain_community.tools import TavilySearchResults
5
6 # Create the Tavily search tool
7 tavily_tool = TavilySearchResults(max_results=5)
8
9 # Function to execute search queries from AnswerQuestion tool calls
10 def execute_tools(state: List[BaseMessage]) -> List[BaseMessage]:
11     last_ai_message = state[-1]
12
13     # Extract tool calls from the AI message
14     if not hasattr(last_ai_message, "tool_calls") or not last_ai_message.tool_calls:
15         return []
16
17     # Process the AnswerQuestion or ReviseAnswer tool calls to extract search queries
18     tool_messages = []
19
20     for tool_call in last_ai_message.tool_calls:
21         if tool_call["name"] in ["AnswerQuestion", "ReviseAnswer"]:
22             call_id = tool_call["id"]
23             search_queries = tool_call["args"].get("search_queries", [])
24
25             # Execute each search query using the tavily tool
26             query_results = {}
27             for query in search_queries:
28                 result = tavily_tool.invoke(query)
29
30                 # Create a tool message with the results
31                 tool_messages.append(
32                     ToolMessage(
33                         content=json.dumps(query_results),
34                         tool_call_id=call_id
35                     )
36                 )
37
38
39     return tool_messages
```



```
execute_tools.py X
4_reflexion_agent_system > execute_tools.py > execute_tools
10 def execute_tools(state: List[BaseMessage]) -> List[BaseMessage]:
11     # Extract tool calls from the AI message
12     if not hasattr(last_ai_message, "tool_calls") or not last_ai_message.tool_calls:
13         return []
14
15     # Process the AnswerQuestion or ReviseAnswer tool calls to extract search queries
16     tool_messages = []
17
18     for tool_call in last_ai_message.tool_calls:
19         if tool_call["name"] in ["AnswerQuestion", "ReviseAnswer"]:
20             call_id = tool_call["id"]
21             search_queries = tool_call["args"].get("search_queries", [])
22
23             # Execute each search query using the tavily tool
24             query_results = {}
25             for query in search_queries:
26                 result = tavily_tool.invoke(query)
27                 query_results[query] = result
28
29             # Create a tool message with the results
30             tool_messages.append(
31                 ToolMessage(
32                     content=json.dumps(query_results),
33                     tool_call_id=call_id
34                 )
35             )
36
37
38     return tool_messages
```

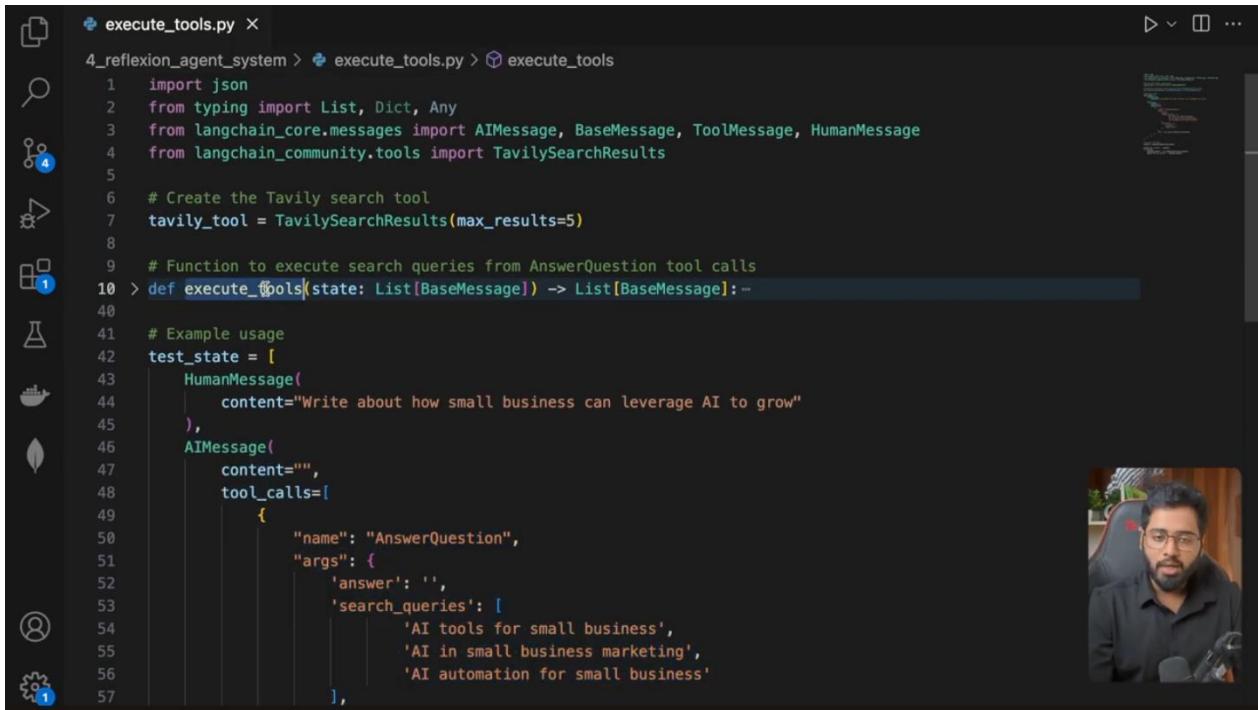


```
execute_tools.py X
4_reflexion_agent_system > execute_tools.py > execute_tools
33             ToolMessage(
34                 content=json.dumps(query_results),
35                 tool_call_id=call_id
36             )
37
38
39     return tool_messages
40
41 # Example usage
42 test_state = [
43     HumanMessage(
44         content="Write about how small business can leverage AI to grow"
45     ),
46     AIMessage(
47         content="",
48         tool_calls=[
49             {
50                 "name": "AnswerQuestion",
51                 "args": {
52                     "search_queries": [
53                         "How can small businesses leverage AI to grow their operations?"
54                     ]
55                 }
56             }
57         ]
58     )
59 ]
```



```
execute_tools.py x
4_reflexion_agent_system > execute_tools.py > ...
42     test_state = [
43         HumanMessage(
44             content="Write about how small business can leverage AI to grow"
45         ),
46         AIMessage(
47             content="",
48             tool_calls=[
49                 {
50                     "name": "AnswerQuestion",
51                     "args": {
52                         "answer": '',
53                         "search_queries": [
54                             'AI tools for small business',
55                             'AI in small business marketing',
56                             'AI automation for small business'
57                         ],
58                         "reflection": {
59                             "missing": '',
60                             "superfluous": ''
61                         },
62                     },
63                     "id": "call_KpYHichFFEmLitHFvFhKy1Ra",
64                 }
65             ],
66         )
67     ]
68
69 # Execute the tools
```

Let us test as below



```
execute_tools.py x
4_reflexion_agent_system > execute_tools.py > execute_tools
1 import json
2 from typing import List, Dict, Any
3 from langchain_core.messages import AIMessage, BaseMessage, ToolMessage, HumanMessage
4 from langchain_community.tools import TavilySearchResults
5
6 # Create the Tavily search tool
7 tavily_tool = TavilySearchResults(max_results=5)
8
9 # Function to execute search queries from AnswerQuestion tool calls
10 def execute_tools(state: List[BaseMessage]) -> List[BaseMessage]:-
11
12 # Example usage
13 test_state = [
14     HumanMessage(
15         content="Write about how small business can leverage AI to grow"
16     ),
17     AIMessage(
18         content="",
19         tool_calls=[
20             {
21                 "name": "AnswerQuestion",
22                 "args": {
23                     "answer": '',
24                     "search_queries": [
25                         'AI tools for small business',
26                         'AI in small business marketing',
27                         'AI automation for small business'
28                     ],
29                 }
30             }
31         ],
32     )
33 ]
```



```
execute_tools.py
4_reflexion_agent_system > execute_tools.py > ...
45     ),
46     AIMessage(
47         content="",
48         tool_calls=[
49             {
50                 "name": "AnswerQuestion",
51                 "args": {
52                     "answer": '',
53                     "search_queries": [
54                         'AI tools for small business',
55                         'AI in small business marketing',
56                         'AI automation for small business'
57                     ],
58                     'reflection': [
59                         'missing': '',
60                         'superfluous': ...
61                     ]
62                 },
63                 "id": "call_KpYHichFFEEmLitHFvFhKy1Ra",
64             },
65         ],
66     ],
67 )
68 # Execute the tools
69 results = execute_tools(test_state)
70
71
```



```
execute_tools.py
4_reflexion_agent_system > execute_tools.py > ...
4 from langchain_community.tools import TavilySearchResults
5
6 # Create the Tavily search tool
7 tavily_tool = TavilySearchResults(max_results=5)
8
9 # Function to execute search queries from AnswerQuestion tool calls
10 def execute_tools(state: List[BaseMessage]) -> List[BaseMessage]: ...
11
12 # Example usage
13 test_state = [
14     HumanMessage(
15         content="Write about how small business can leverage AI to grow"
16     ),
17     AIMessage(-
18 )
19 ]
20
21 # Execute the tools
22 results = execute_tools(test_state)
23
24 print("Raw results:", results)
25 if results:
26     parsed_content = json.loads(results[0].content)
27     print("Parsed content:", parsed_content)
```

We have now built all the components of the Reflexion system, we now need to pull them together and run the graph

LangGraph Crash Course #14 - Reflexion Agent - Building Graph



Harish Neel | AI
7.72K subscribers

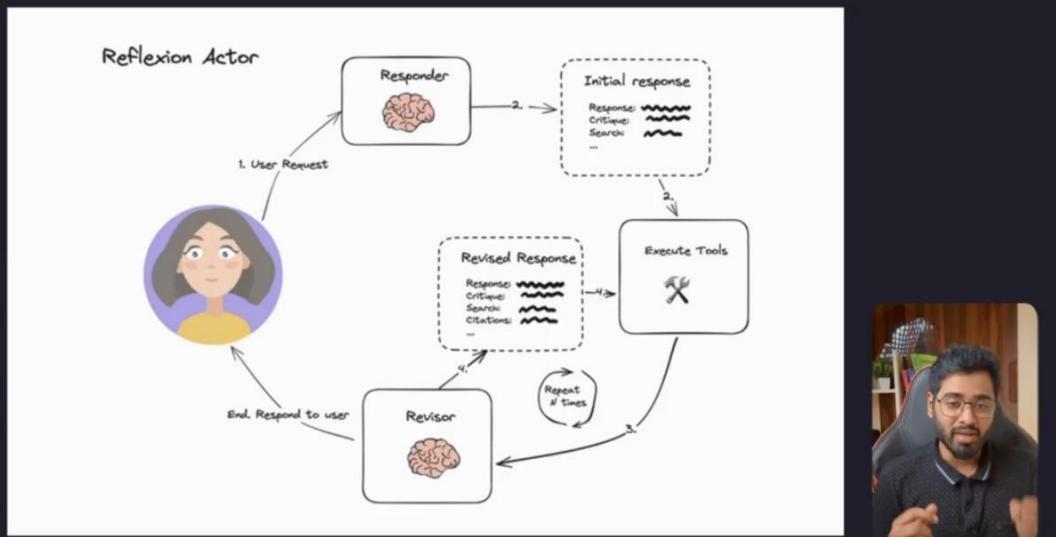
Subscribe



2,673 views Feb 24, 2025 #python #artificialintelligence #langgraph

In this beginner-friendly playlist, I'll show you how to use LangGraph to build powerful AI-agents from scratch

Reflexion Agent System



EXPLORER

- LANGGRAPH
 - 1_Introduction
 - react_agent_basic.py
 - 2_basic_reflection_system
 - > __pycache__
 - basic.py
 - chains.py
 - 3_reflexion_agent_system
 - > __pycache__
 - chains.py
 - execute_tools.py
 - reflexion_graph.py **U**
 - sample_final_output.py
 - schema.py

reflexion_graph.py U X

```

3_reflexion_agent_system > reflexion_graph.py > ...
1  from typing import List
2
3  from langchain_core.messages import BaseMessage, ToolMessage
4  from langgraph.graph import END, MessageGraph
5
6  from chains import revisor_chain, first_responder_chain
7  from execute_tools import execute_tools
8
9  graph = MessageGraph()
10
11 graph.add_node("draft", first_responder_chain)
12 graph.add_node("execute_tools", execute_tools)
13 graph.add_node("revisor", revisor_chain)
14
15
16
    
```

We can now connect the nodes together using edges as below

EXPLORER

- LANGGRAPH
 - 1_Introduction
 - react_agent_basic.py
 - 2_basic_reflection_system
 - > __pycache__
 - basic.py
 - chains.py
 - 3_reflexion_agent_system
 - > __pycache__
 - chains.py
 - execute_tools.py
 - reflexion_graph.py **U**
 - sample_final_output.py
 - schema.py

reflexion_graph.py U X

```

3_reflexion_agent_system > reflexion_graph.py > ...
3
4  from langchain_core.messages import BaseMessage, ToolMessage
5  from langgraph.graph import END, MessageGraph
6
7  from chains import revisor_chain, first_responder_chain
8  from execute_tools import execute_tools
9
10 graph = MessageGraph()
11 MAX_ITERATIONS = 2
12
13
    
```

EXPLORER

- LANGGRAPH
 - 1_Introduction
 - react_agent_basic.py
 - 2_basic_reflection_system
 - > __pycache__
 - basic.py
 - chains.py
 - 3_reflexion_agent_system
 - > __pycache__
 - chains.py
 - execute_tools.py **M**
 - reflexion_graph.py **U**
 - sample_final_output.py **U**
 - schema.py

reflexion_graph.py U X

```

3_reflexion_agent_system > reflexion_graph.py > event_loop
15
16
17 graph.add_edge("draft", "execute_tools")
18 graph.add_edge("execute_tools", "revisor")
19
20 def event_loop(state: List[BaseMessage]) -> str:
21     count_tool_visits = sum(isinstance(item, ToolMessage) for item in
22     state)
23     num_iterations = count_tool_visits
24     if num_iterations > MAX_ITERATIONS:
25         return END
26     return "execute_tools"
27
28
29 graph.add_conditional_edges("revisor", event_loop)
    
```

```
def event_loop(state: List[BaseMessage]) -> str:
    return "execute_tools"
graph.add_conditional_edges("revisor", event_loop)
graph.set_entry_point("draft")
app = graph.compile()
print(app.get_graph().draw_mermaid())
response = app.invoke(
    "Write about how small business can leverage AI to grow"
)
print(response[-1].tool_calls[0]["args"]["answer"])
print(response, "response")
```

```
from typing import List
from langchain_core.messages import BaseMessage, ToolMessage
from langgraph.graph import END, MessageGraph
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
(venv) (base) harishb@Harishs-MacBook-Air LangGraph % /Users/harishb/Desktop/LangGraph/venv/bin/python /Users/harishb/Desktop/LangGraph/3_reflexion_agent_system/reflexion_graph.py
%{init: {'flowchart': {'curve': 'linear'}}}%
graph TD;
    _start_([<p>_start_</p>]):::first
    draft(draft)
    execute_tools(execute_tools)
    revisor(revisor)
    _end_([<p>_end_</p>]):::last
    _start_ --> draft;
    draft --> execute_tools;
    execute_tools --> revisor;
    revisor --> draft;
    revisor --> execute_tools;
    revisor --> _end_;
    classDef default fill:#f2f0ff,line-height:1.2
    classDef first fill-opacity:0
    classDef last fill:#fbfbfc
```

```
.partial(
    time=lambda: datetime.datetime.now().isoformat(),
)
first_responder_prompt_template = actor_prompt_template.partial(
    first_instruction="Provide a detailed ~250 word answer"
)
llm = ChatOpenAI(model="gpt-4o")
first_responder_chain = first_responder_prompt_template |
llm.bind_tools(tools=[AnswerQuestion], tool_choice='AnswerQuestion')
validator = PydanticToolsParser(tools=[AnswerQuestion])
# Revisor section
```

Take away the parser from line 38 and run again



PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
(venv) (base) harishb@Harishs-MacBook-Air LangGraph % /Users/harishb/Desktop/LangGraph/venv/bin/python /Users/harishb/Desktop/LangGraph/3_reflexion_agent_system/reflexion_graph.py
%%init: {'flowchart': {'curve': 'linear'}}%%
graph TD;
    _start_((<p>_start_</p>)):::first
    draft(draft)
    execute_tools(execute_tools)
    revisor(revisor)
    _end_((<p>_end_</p>)):::last
    _start_ --> draft;
    draft --> execute_tools;
    execute_tools --> revisor;
    revisor --> draft;
    revisor --> execute_tools;
    revisor --> _end_;
    classDef default fill:#f2f0ff,line-height:1.2
    classDef first fill-opacity:0
    classDef last fill:#bfbbfc
```



EXPLORER

LANGGRAPH

- 1_Introduction
 - react_agent_basic.py
- 2_basic_reflection_system
 - _pycache_
 - basic.py
 - chains.py
- 3_reflexion_agent_system
 - _pycache_
 - chains.py
 - execute_tools.py
 - reflexion_graph.py
 - sample_final_output.py
 - schema.py

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
classDef default fill:#f2f0ff,line-height:1.2
classDef first fill-opacity:0
classDef last fill:#bfbbfc
```

Small businesses can leverage AI to grow by enhancing customer service, optimizing operations, and fostering innovation. AI-powered chatbots provide around-the-clock customer support, enhancing engagement and satisfaction without hefty investments in customer service teams [1]. Additionally, AI-driven analytics process large datasets for insights into consumer behavior and market trends, enabling informed decisions on product offerings, pricing, and target audiences [2].

Automation of repetitive tasks through AI drives operational efficiency. For example, AI can streamline inventory management by forecasting stock levels and optimizing supply chains, thereby reducing waste and ensuring product availability [3]. AI tools also personalize marketing efforts and optimize ad campaigns, amplifying reach and conversion rates by analyzing real-time feedback and social media trends, allowing businesses to quickly adapt their strategies [4].

In terms of innovation, AI simplifies product design and testing. Simulation tools powered by AI help businesses experiment and refine products before launching, minimizing risks and enhancing success potential [5]. To achieve these benefits, small businesses should prioritize integrating AI solutions that are both cost-effective and scalable in alignment with their growth strategies [6].



EXPLORER

LANGGRAPH

- 1_Introduction
 - react_agent_basic.py
- 2_basic_reflection_system
 - _pycache_
 - basic.py
 - chains.py
- 3_reflexion_agent_system
 - _pycache_
 - chains.py
 - execute_tools.py
 - reflexion_graph.py
 - sample_final_output.py
 - schema.py

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

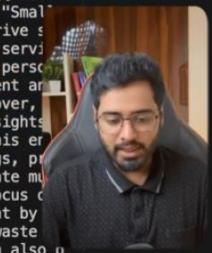
```
nching, minimizing risks and enhancing success potential [5]. To achieve these benefits, small businesses should prioritize integrating AI solutions that are both cost-effective and scalable in alignment with their growth strategies [6].
```

Incorporating case studies of successful AI implementation in small businesses could provide valuable insights and guidance for new adopters, showcasing effective applications and achievable outcomes.

References:

- [1] <https://quantive.com/resources/articles/ai-in-business-strategy>
- [2] <https://aitgglobalinc.com/ai-for-small-business/>
- [3] <https://rtslabs.com/top-ai-tools-for-small-businesses>
- [4] <https://www.dialpad.com/blog/ai-tools-for-small-business/>
- [5] <https://online.hbs.edu/blog/post/ai-business-strategy>
- [6] <https://www.accenture.com/us-en/insights/strategy/ai-enabled-growth>

[HumanMessage(content='Write about how small business can leverage AI to grow', additional_kwargs={}, response_metadata={}, id='4d82e5e3-ae39-40d5-8c99-ca225961a0'), AIMessage(content='', additional_kwargs={'tool_calls': [{"id": "call_cyKmztxM6xWzopTkp03HjC", "function": {"arguments": {"answer": "Small businesses can harness the power of Artificial Intelligence (AI) to drive significant growth and competitiveness. Firstly, AI can enhance customer service through chatbots, which are capable of providing 24/7 responsive and personalized customer interactions. This helps in improving customer engagement and satisfaction without the need for a large customer service team. Moreover, AI can optimize operations by streamlining inventory management, supply chain processes, and marketing efforts, leading to better efficiency and cost savings."}}}], tool_call_ids=['call_cyKmztxM6xWzopTkp03HjC'])]



This screenshot shows a developer's workspace in a code editor. The left sidebar displays a file tree under the 'EXPLORER' tab, showing a directory structure for 'LANGGRAPH' with sub-folders '1_Introduction', '2_basic_reflection_system', and '3_reflexion_agent_system'. Inside '3_reflexion_agent_system', there are files like 'react_agent_basic.py', 'basic.py', 'chains.py', 'execute_tools.py', and 'reflexion_graph.py'. The main editor area shows the content of 'reflexion_graph.py' with the following code:

```
26
27 graph.add_conditional_edges("revisor", event_loop)
28 graph.set_entry_point("draft")
29
30 app = graph.compile()
```

The 'TERMINAL' tab is active, showing a terminal session with two tabs: 'zsh' and 'Python'. The Python tab is selected, displaying the command 'cd /Users/.../Desktop/execute_tools'. The right sidebar contains a 'PROBLEMS' section with several references to AI strategy articles, and a 'PORTS' section.



This screenshot shows the same developer's workspace as the previous one, but with a different video call overlay. The man with glasses and a beard is now looking directly at the camera. The code editor and file tree are identical to the first screenshot, showing the 'reflexion_graph.py' file with the same code. The terminal session and sidebar sections are also the same.

EXPLORER

LANGGRAPH

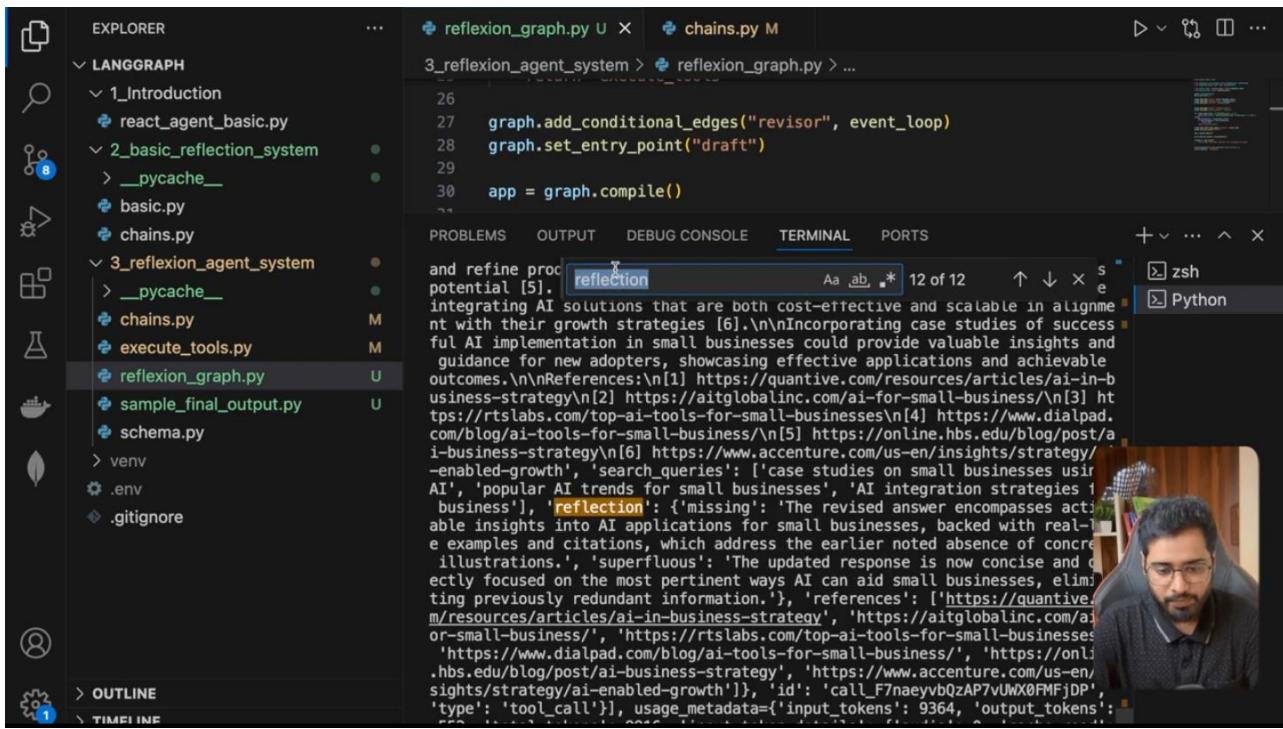
- 1_Introduction
 - react_agent_basic.py
- 2_basic_reflection_system
 - basic.py
 - chains.py
- 3_reflexion_agent_system
 - chains.py
 - execute_tools.py
 - reflexion_graph.py
 - sample_final_output.py
 - schema.py

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

and refine proc potential [5].
integrating AI solutions that are both cost-effective and scalable in alignment with their growth strategies [6]. Incorporating case studies of successful AI implementation in small businesses could provide valuable insights and guidance for new adopters, showcasing effective applications and achievable outcomes.

References:
[1] <https://quantive.com/resources/articles/ai-in-business-strategy>
[2] <https://aitglobalinc.com/ai-for-small-business/>
[3] <https://rtslabs.com/top-ai-tools-for-small-businesses>
[4] <https://www.dialpad.com/blog/ai-tools-for-small-business/>
[5] <https://online.hbs.edu/blog/post/ai-business-strategy>
[6] <https://www.accenture.com/us-en/insights/strategy/ai-enabled-growth>

reflection: {'missing': 'The revised answer encompasses actionable insights into AI applications for small businesses, backed with real-life examples and citations, which address the earlier noted absence of concrete illustrations.', 'superfluous': 'The updated response is now concise and directly focused on the most pertinent ways AI can aid small businesses, eliminating previously redundant information.'}, 'references': ['<https://quantive.com/resources/articles/ai-in-business-strategy>', '<https://aitglobalinc.com/ai-for-small-business/>', '<https://rtslabs.com/top-ai-tools-for-small-businesses>', '<https://www.dialpad.com/blog/ai-tools-for-small-business/>', '<https://online.hbs.edu/blog/post/ai-business-strategy>', '<https://www.accenture.com/us-en/insights/strategy/ai-enabled-growth>'], 'id': 'call_F7naeyvbzAP7vWx0FMFjDP', 'type': 'tool_call'}, usage_metadata={'input_tokens': 9364, 'output_tokens': 120}



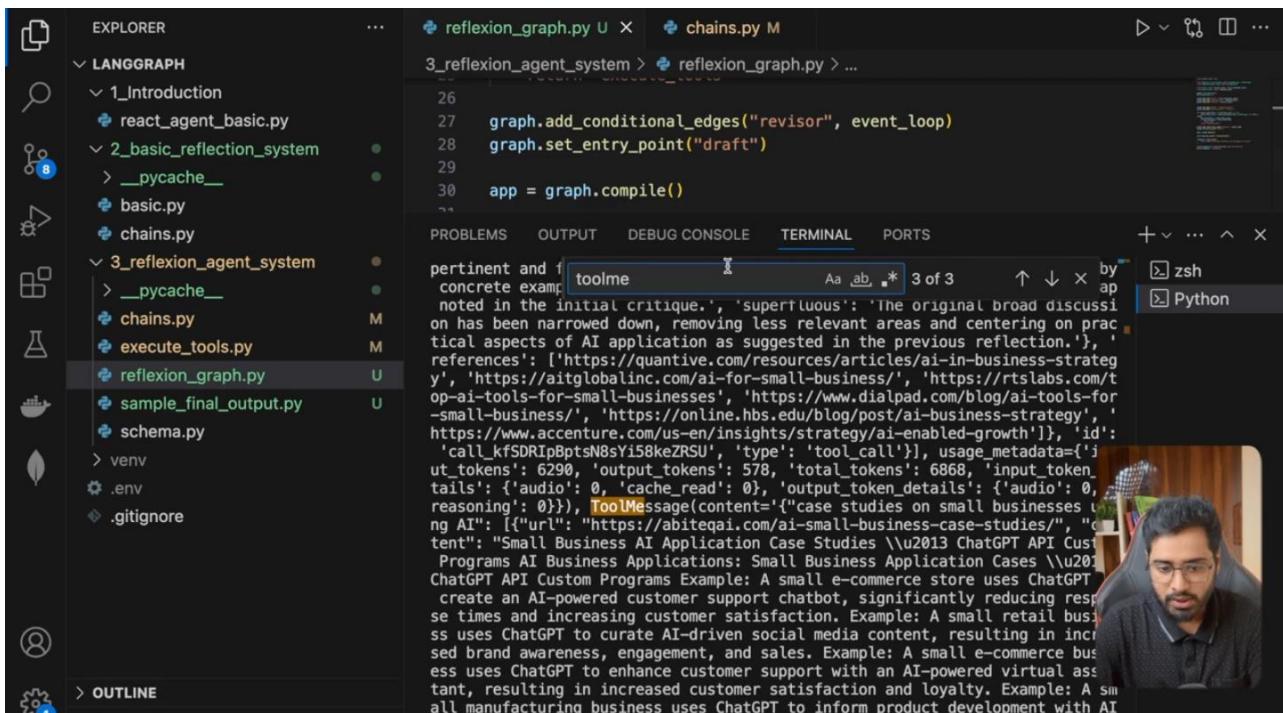
EXPLORER

LANGGRAPH

- 1_Introduction
 - react_agent_basic.py
- 2_basic_reflection_system
 - basic.py
 - chains.py
- 3_reflexion_agent_system
 - chains.py
 - execute_tools.py
 - reflexion_graph.py
 - sample_final_output.py
 - schema.py

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

pertinent and concrete example toolme noted in the initial critique., 'superfluous': 'The original broad discussion has been narrowed down, removing less relevant areas and centering on practical aspects of AI application as suggested in the previous reflection.'}, 'references': ['<https://quantive.com/resources/articles/ai-in-business-strategy>', '<https://aitglobalinc.com/ai-for-small-business/>', '<https://rtslabs.com/top-ai-tools-for-small-businesses>', '<https://www.dialpad.com/blog/ai-tools-for-small-business/>', '<https://www.accenture.com/us-en/insights/strategy/ai-enabled-growth>'], 'id': 'call_kfSDR1pBptsN8sYi58keZRSU', 'type': 'tool_call'}, usage_metadata={'input_tokens': 6290, 'output_tokens': 578, 'total_tokens': 6868, 'input_token_details': {'audio': 0, 'cache_read': 0}, 'output_token_details': {'audio': 0, 'reasoning': 0}}, ToolMessage(content='("case studies on small businesses using AI": [{"url": "https://abiteqai.com/ai-small-business-case-studies"}], "content": "Small Business AI Application Case Studies \u2013 ChatGPT API Custom Programs AI Business Applications: Small Business Application Cases \u2013 ChatGPT API Custom Programs Example: A small e-commerce store uses ChatGPT to create an AI-powered customer support chatbot, significantly reducing response times and increasing customer satisfaction. Example: A small retail business uses ChatGPT to curate AI-driven social media content, resulting in increased brand awareness, engagement, and sales. Example: A small e-commerce business uses ChatGPT to enhance customer support with an AI-powered virtual assistant, resulting in increased customer satisfaction and loyalty. Example: A small manufacturing business uses ChatGPT to inform product development with AI")'})



smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects/p/9ed8e2e3-4ed6-42f6-9f34-dbe45900bbc4?timeModel=%7B"duration":

pr-minty-councilperson-79

Runs Threads Monitor Setup

1 filter Last 7 days Root Runs LLM Calls All Runs Columns

Name	Input	Output	Error	Start Time	Latency	Dataset
LangGraph	Write about how small business can leverage AI to grow	ai: {"tool_calls":{}}		24/02/2025, 11:48:43	79.11s	
LangGraph	Write about how small business can leverage AI to grow	NotImplementedError: No implementation found for tool call		24/02/2025, 11:47:00	9.69s	
LangGraph	Write about AI-Power...	ai: {"tool_calls":{}}		23/02/2025, 18:37:42	43.30s	
LangGraph	Write about AI-Power...	NotImplementedError: No implementation found for tool call		23/02/2025, 18:34:20	11.63s	
LangGraph	Write about AI-Power...	1 validation error ...		23/02/2025, 18:31:46	12.09s	
tavily_search_results_json	AI automation for small business	None: AI For Small Business		23/02/2025, 18:31:43	2.07s	
tavily_search_results_json	AI in small business ...	None: AI For Sma...		23/02/2025, 18:31:43	2.07s	
tavily_search_results_json	AI tools for small busi...	None: AI For Smal...		23/02/2025, 18:31:43	3.09s	
JsonOutputToolsParser	ai: ""	[{"args": {"answer": "Artificial Intelligence (AI) is revolutionizing the way small businesses operate. By leveraging AI, companies can automate repetitive tasks, analyze large amounts of data to gain insights, and improve customer service through chatbots and personalization. This allows them to stay competitive and efficient in today's fast-paced market."}}]		23/02/2025, 18:31:43	0.00s	
RunnableSequence	human: Write me a bl...	[{"answer": "Artificial Intelligence (AI) is revolutionizing the way small businesses operate. By leveraging AI, companies can automate repetitive tasks, analyze large amounts of data to gain insights, and improve customer service through chatbots and personalization. This allows them to stay competitive and efficient in today's fast-paced market."}]		23/02/2025, 18:31:30	12.99s	

Stats
Last 7 days
RUN COUNT 33
TOTAL TOKENS 52,035 / \$0.20
MEDIAN TOKENS 665
ERRORS 18%
% STR 0%
LATEN...
P50: 12.09s
F...
Input Key Input Tool Calls Answer Score

smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects/p/9ed8e2e3-4ed6-42f6-9f34-dbe45900bbc4?timeModel=%7B"duration":

pr-minty-councilperson-79

Runs Threads Monitor Setup

1 filter Last 7 days Root Runs LLM Calls All Runs

Name	Input	Output	Error	Start Time	Latency	Dataset
LangGraph	Write about how small business can leverage AI to grow	ai: {"tool_calls":{}}		24/02/2025, 11:48:43	79.11s	
LangGraph	Write about how small business can leverage AI to grow	NotImplementedError: No implementation found for tool call		24/02/2025, 11:47:00	9.69s	
LangGraph	Write about AI-Power...	ai: {"tool_calls":{}}		23/02/2025, 18:37:42	43.30s	
LangGraph	Write about AI-Power...	NotImplementedError: No implementation found for tool call		23/02/2025, 18:34:20	11.63s	
LangGraph	Write about AI-Power...	1 validation error ...		23/02/2025, 18:31:46	12.09s	
tavily_search_results_json	AI automation for small business	None: AI For Small Business		23/02/2025, 18:31:43	2.07s	
tavily_search_results_json	AI in small business ...	None: AI For Sma...		23/02/2025, 18:31:43	2.07s	
tavily_search_results_json	AI tools for small busi...	None: AI For Smal...		23/02/2025, 18:31:43	3.09s	
JsonOutputToolsParser	ai: ""	[{"args": {"answer": "Artificial Intelligence (AI) is revolutionizing the way small businesses operate. By leveraging AI, companies can automate repetitive tasks, analyze large amounts of data to gain insights, and improve customer service through chatbots and personalization. This allows them to stay competitive and efficient in today's fast-paced market."}}]		23/02/2025, 18:31:43	0.00s	
RunnableSequence	human: Write me a bl...	[{"answer": "Artificial Intelligence (AI) is revolutionizing the way small businesses operate. By leveraging AI, companies can automate repetitive tasks, analyze large amounts of data to gain insights, and improve customer service through chatbots and personalization. This allows them to stay competitive and efficient in today's fast-paced market."}]		23/02/2025, 18:31:30	12.99s	

LangGraph

Run Feedback Metadata Run ID Trace ID

Input

Input
Write about how small business can leverage AI to grow

Output

HUMAN
Write about how small business can leverage AI to grow

AI
AnswerQuestion
call_cyKMztxM6xWzMopTkP03Hjc
answer: |
Small businesses can harness the power of Artificial Intelligence (AI) to drive significant growth and competitiveness. Firstly, AI can enhance customer service through chatbots, which are capable of

START TIME 02/24/2025, 11:48:43 AM
END TIME 02/24/2025, 11:50:02 AM
TIME TO FIRST TOKEN N/A
STATUS Success

smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects/p/9ed8e2e3-4ed6-42f6-9f34-dbe45900bbc4?timeModel=%7B"duration": 1000, "start": 0, "end": 1000, "step": 1000, "unit": "ms"}, "language": "JavaScript", "version": "0.1.0", "name": "LangGraph", "description": "A LangChain component that performs search and retrieval using a vector database.", "category": "Tool", "status": "Success", "tokens": 21367, "latency": 79.11, "type": "Chain", "image": "https://www.langchain.com/images/langgraph.png"}

smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects/p/9ed8e2e3-4ed6-42f6-9f34-dbe45900bbc4?timeModel=%7B"duration": 1000, "start": 0, "end": 1000, "step": 1000, "unit": "ms"}, "language": "JavaScript", "version": "0.1.0", "name": "LangGraph", "description": "A LangChain component that performs search and retrieval using a vector database.", "category": "Tool", "status": "Success", "tokens": 21367, "latency": 79.11, "type": "Chain", "image": "https://www.langchain.com/images/langgraph.png"}

smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects/p/9ed8e2e3-4ed6-42f6-9f34-dbe45900bbc4?timeModel=%7B"duration":

pr-minty-councilperson-79

Runs Threads Monitor Setup

1 filter Last 7 days Root Runs LLM Calls

Name Input

LangGraph Write about how small businesses can leverage AI to drive growth by enhancing customer service, optimizing operations, and driving innovation.

LangGraph Write about how small businesses can leverage AI to drive growth by enhancing customer service, optimizing operations, and driving innovation.

LangGraph Write about AI-Powered Chatbots

LangGraph Write about AI-Powered Chatbots

LangGraph Write about AI-Powered Chatbots

tavily_search_results_json AI automation for small businesses

tavily_search_results_json AI in small business

tavily_search_results_json AI tools for small business

JsonOutputToolsParser ai: ""

RunnableSequence human: Write me a brief summary of AI's impact on small business operations.

LangGraph

Run Feedback Metadata

AI

ReviseAnswer call_7QigW0ppLHAIQGIBQEz5upZI

answer: |

Small businesses can leverage AI to drive growth by enhancing customer service, optimizing operations, and driving innovation. AI-powered chatbots offer 24/7 customer support, improving engagement and satisfaction without the need for extensive customer service teams [1]. AI-driven analytics tools help businesses process large data volumes to extract insights on consumer behavior, market trends, and operational efficiencies, enabling informed decision-making about product offerings, pricing, and target audiences [2].

Additionally, automating repetitive tasks with AI frees employees to focus on strategic activities. For example, AI can optimize inventory management by predicting stock levels and streamlining supply chains, thus



smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects/p/9ed8e2e3-4ed6-42f6-9f34-dbe45900bbc4?timeModel=%7B"duration":

pr-minty-councilperson-79

Runs Threads Monitor Setup

1 filter Last 7 days Root Runs LLM Calls

Name Input

LangGraph Write about how small businesses can leverage AI to drive growth by enhancing customer service, optimizing operations, and driving innovation.

LangGraph Write about how small businesses can leverage AI to drive growth by enhancing customer service, optimizing operations, and driving innovation.

LangGraph Write about AI-Powered Chatbots

LangGraph Write about AI-Powered Chatbots

LangGraph Write about AI-Powered Chatbots

tavily_search_results_json AI automation for small businesses

tavily_search_results_json AI in small business

tavily_search_results_json AI tools for small business

JsonOutputToolsParser ai: ""

RunnableSequence human: Write me a brief summary of AI's impact on small business operations.

LangGraph

Run Feedback Metadata

AI

ReviseAnswer call_7QigW0ppLHAIQGIBQEz5upZI

Scaling with the company's growth [1] [2] [3] [4] [5] [6]

References:

- [1] <https://quantive.com/resources/articles/ai-in-business-strategy>
- [2] <https://aitglobalinc.com/ai-for-small-business/>
- [3] <https://rtslabs.com/top-ai-tools-for-small-businesses>
- [4] <https://www.dialpad.com/blog/ai-tools-for-small-business/>
- [5] <https://online.hbs.edu/blog/post/ai-business-strategy>
- [6] https://www.accenture.com/us-en/insights/strategy/ai-enabled-growth-search_queries

- case studies on small businesses using AI
- popular AI trends for small businesses
- AI integration strategies for business



smith.langchain.com/o/e4a9f3da-d95f-42b8-be4e-c631ca42cc3b/projects/p/9ed8e2e3-4ed6-42f6-9f34-dbe45900bbc4?timeModel=%7B"duration":

pr-minty-councilperson-79

Runs Threads Monitor Setup

1 filter Last 7 days Root Runs LLM Calls

Name Input

- LangGraph Write about how small businesses can benefit from AI
- LangGraph Write about how small businesses can benefit from AI
- LangGraph Write about AI-Powered Business Solutions
- LangGraph Write about AI-Powered Business Solutions
- tavily_search_results_json AI automation for small businesses
- tavily_search_results_json AI in small business
- tavily_search_results_json AI tools for small business
- JsonOutputToolsParser ai: ""
- RunnableSequence human: Write me a brief summary of the AI revolution in small business

TRACE

LangGraph 79.11s

draft 10.28s

gpt-4o 10.22s

execute_tools 2.75s

tavily_search_results_json 2.75s

tavily_search_results_json 2.75s

revisor 14.19s

gpt-4o 14.18s

event_loop 0.00s

execute_tools 2.58s

tavily_search_results_json 2.15s

tavily_search_results_json 2.58s

tavily_search_results_json 2.08s

revisor 27.05s

gpt-4o 27.05s

event_loop 0.00s

execute_tools 2.46s

LangGraph

Run Feedback Metadata

Run ID Trace ID

AI

ReviseAnswer call_7Q1gW0ppLHAIQGIBQEzSupZI

[3] <https://rtslabs.com/top-ai-tools-for-small-businesses>

[4] <https://www.dialpad.com/blog/ai-tools-for-small-business/>

[5] <https://online.hbs.edu/blog/post/ai-business-strategy>

[6] <https://www.accenture.com/us-en/insights/strategy/ai-enabled-growth-search-queries>:

- case studies on small businesses using AI
- popular AI trends for small businesses
- AI integration strategies for business reflection:
- missing: The revised answer includes pertinent and focused use cases of AI for small business growth, bolstered by concrete examples and backed up with citations, effectively bridging the gap noted in the initial critique.
- superfluous: The original broad discussion