



LangChain for AI Agents

+

•

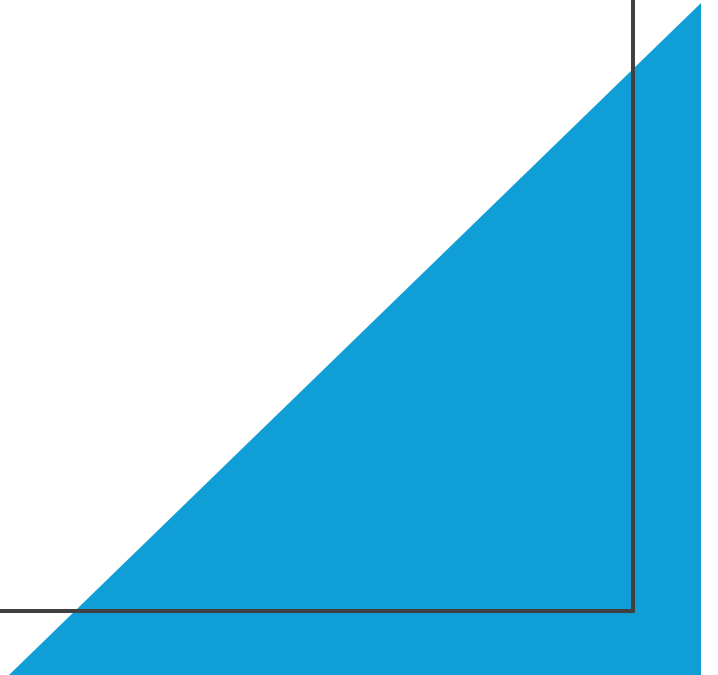
○

What to expect ?

- What is LangChain ?
- Why LangChain ?
- Common Terminologies
 - Prompt
 - Prompt Template
 - Parser
 - Moderation
 - Inference
- What is a Chain ?
- LangChain vs LangGraph vs LangSmith/LangFuse vs LangFlow

LangChain

- Simplifies the process of building applications using LLMs
- Open source framework
- Building chains
- Add a layer of abstraction
- Developer friendly
- Easy to scale
- Reduce dependency on any specific framework



Model Inferencing without LangChain

OpenAI Model Inferencing

```
1 from openai import OpenAI
2 client = OpenAI()
3
4 response = client.responses.create(
5     model="gpt-4o-mini",
6     input="Write a one-sentence bedtime story about a unicorn."
7 )
8
9 pprint(response.output_text)
```

```
('As a shimmering crescent moon rose high in the sky, Luna the unicorn spread '
'her sparkling wings and fluttered through the starlit forest, bringing sweet '
'dreams to all the creatures who slept below.')
```

OpenAI Model Inferencing

```
1 from groq import Groq
2
3 client = Groq()
4 completion = client.chat.completions.create(
5     model="meta-llama/llama-4-scout-17b-16e-instruct",
6     messages=[
7         {
8             "role": "user",
9             "content": "Write a one-sentence bedtime story about a unicorn."
10        }
11    ],
12    temperature=1,
13    max_completion_tokens=1024,
14    top_p=1,
15    stream=True,
16    stop=None,
17 )
18
19 chunks = ""
20 for chunk in completion:
21     # print(chunk.choices[0].delta.content or "", end="")
22     if chunk.choices[0].delta.content is None:
23         continue
24     chunks += chunk.choices[0].delta.content
```

```
1 pprint(chunks)
```

```
('As the moon rose high in the sky, a beautiful unicorn named Luna pranced '
'through a field of sparkling flowers, her shimmering coat and horn glowing '
'softly in the gentle night light.')
```

Model Inferencing with LangChain

```
1 from langchain_groq import ChatGroq
2
3 # creating LLM chatgroq instance for specific model
4 llm = ChatGroq(temperature=0, model_name="llama-3.1-8b-instant")
5
6 #Inferencing model
7 response = llm.invoke("hi")
8 print(response.content)
```

How can I assist you today?

```
1 from langchain_openai import ChatOpenAI
2
3 # Creating LLM OpenAI instance for specific OpenAI model
4 llm = ChatOpenAI(temperature=0, model_name="gpt-4o-mini")
5
6 #Inferencing model
7 response = llm.invoke("hi")
8 print(response.content)
```

Hello! How can I assist you today?

+

•

○

Prompt and Templates:

- Prompt:
 - input provided to the model to generate a specific output.
 - Can include question, instruction, context, examples, etc.
 - Structure of prompt is very important
- Templates:
 - Standard prompt with placeholders that can be updated on the fly
 - Makes the prompt more dynamic
 - Similar to f-string in python
- Types:
 - Prompt Template
 - Chat Prompt template
 - Etc.

Parser

Data extraction tool that extract the relevant information from any input content

write an email to my boss seeking 2 days leave.

Sure! Here's a professional and concise email you can use to request two days of leave:

Subject: Request for Leave on [Insert Dates]

Dear [Boss's Name],

I hope this message finds you well. I would like to request leave for two days, on [insert dates], due to [brief reason – optional, e.g., personal matters/family commitment].

Please let me know if any arrangements need to be made to ensure a smooth workflow during my absence. I will make sure all pending tasks are up to date before I leave.

Thank you for your consideration.

Best regards,

[Your Full Name]

[Your Position]

Let me know if you'd like the tone to be more casual, formal, or if you want to include specific details.



```
1 prompt = "write an email to my boss seeking 2 days leave. "  
2  
3 response = llm.invoke(prompt).content  
4 print(response)
```

Here's a sample email you can use as a template:

Subject: Request for Leave – [Date]

Dear [Boss's Name],

I am writing to request two days of leave from [Date] to [Date]. I have some

I will ensure that all my tasks are completed and my work is up to date before

I would greatly appreciate it if you could approve my leave request. If there

Thank you for your understanding and I look forward to hearing back from you.

Best regards,

[Your Name]

Please note that you should customize this email to fit your specific needs.

- Clearly state the dates you are requesting off
- Mention that you will ensure your work is up to date before leaving
- Show appreciation for your boss's time and approval
- Include your name and title (if applicable)

Why Parser?

- Clean response to be consumed directly
- Easy to integrate

```
6 # creating llm object with structured output parser
7 llm_parser = llm.with_structured_output(EmailParser)
8
9 response = llm_parser.invoke(prompt)
10
11 print(response.email)
```

Subject: Request for 2 Days Leave

Dear [Boss's Name],

I am writing to request for 2 days leave on [Date] and [Date]. I have s

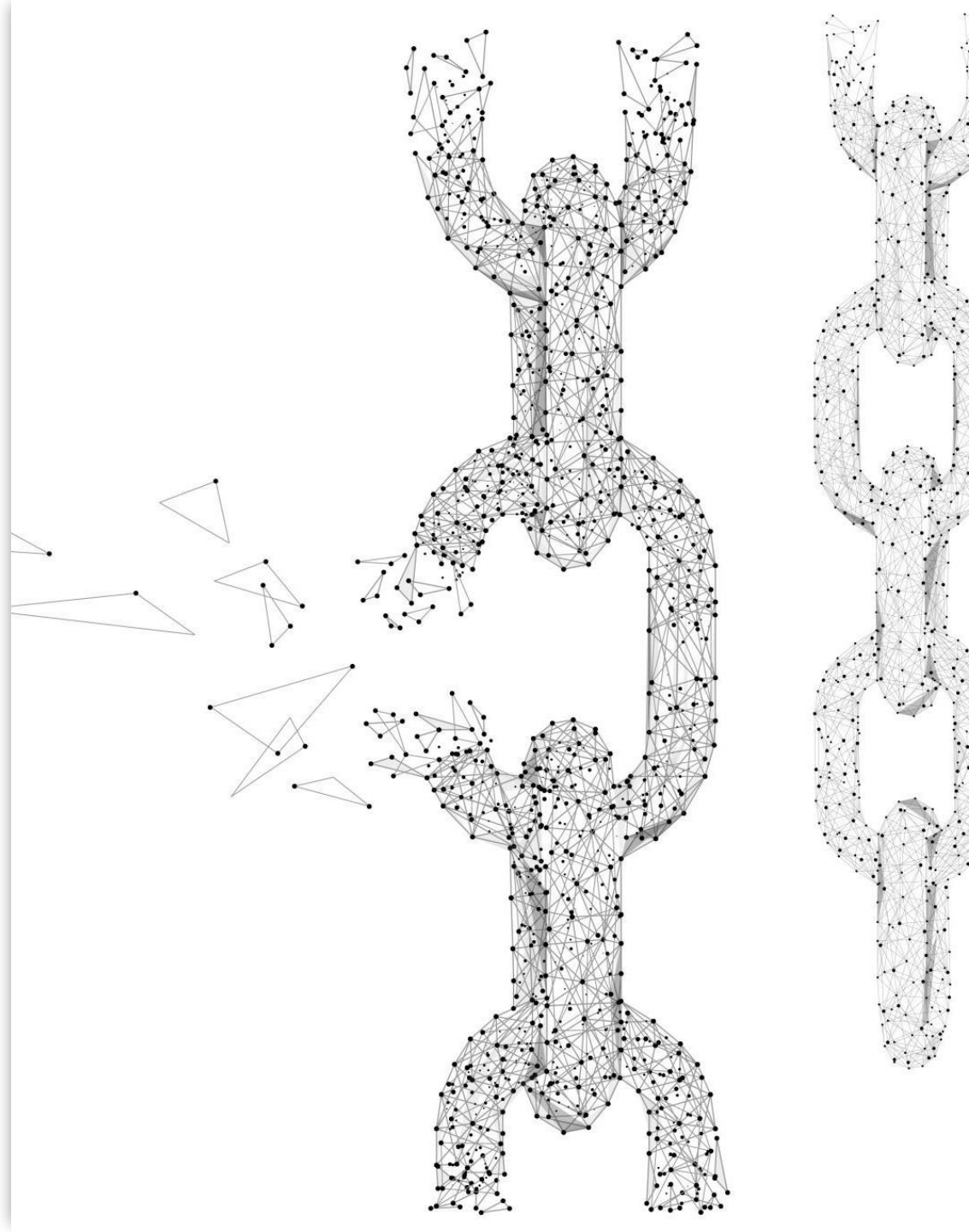
I will ensure that all my tasks are completed and my work is up to date

Thank you for considering my request.

Best regards,
[Your Name]

Chain

- A sequence of call to different component like language model, data preprocessing steps, external tools, executed in a specific order
- Like a pipeline where output of one step becomes the input to the next step
- Fundamental building block of LangChain
- Eg. Chain: Prompt – LLM – Parser





Questions ?