The bridge to possible

Enhancing Collaboration with the Future: Webex and Generative AI Integration

Omer Ilyas
Technical Marketing Engineer Leader

AIHUB-1000

Cisco Webex App

Questions?

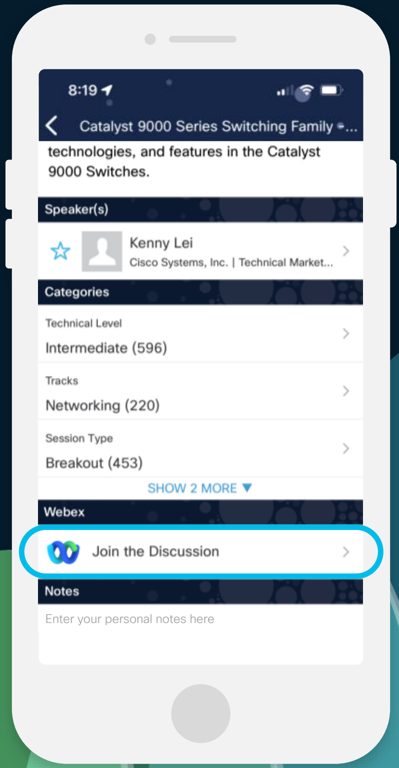
Use Cisco Webex App to chat
with the speaker after the session

How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install the Webex App or go directly to the Webex space
4. Enter messages/questions in the Webex space

Webex spaces will be moderated
by the speaker until June 7, 2024.

https://ciscolive.ciscoevents.com/
ciscolivebot/#AIHUB-1000



8:19

Catalyst 9000 Series Switching Family ...

technologies, and features in the Catalyst 9000 Switches.

Speaker(s)

Kenny Lei
Cisco Systems, Inc. | Technical Market...

Categories

Technical Level
Intermediate (596)

Tracks
Networking (220)

Session Type
Breakout (453)

SHOW 2 MORE ▼

Webex

Join the Discussion

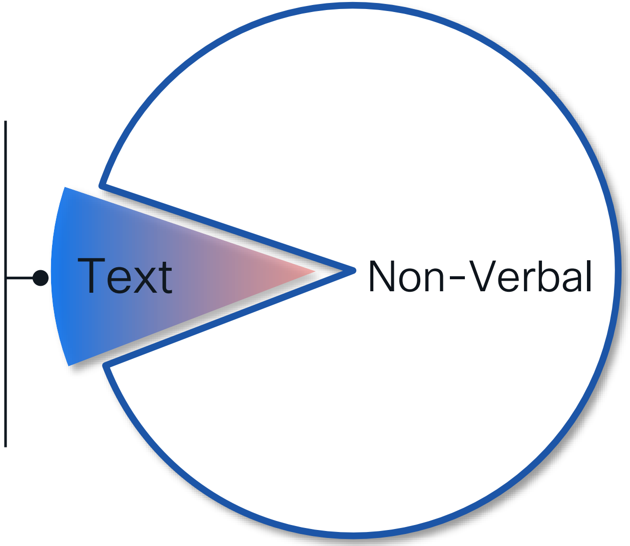
Notes

Enter your personal notes here

Agenda

- Introduction
  - Cisco AI Principles
- What is Neural Network
- Tokenization
- Embeddings & Vector Database
- RAG & Gen AI Framework
- Conclusion

Our Collaboration Strategy



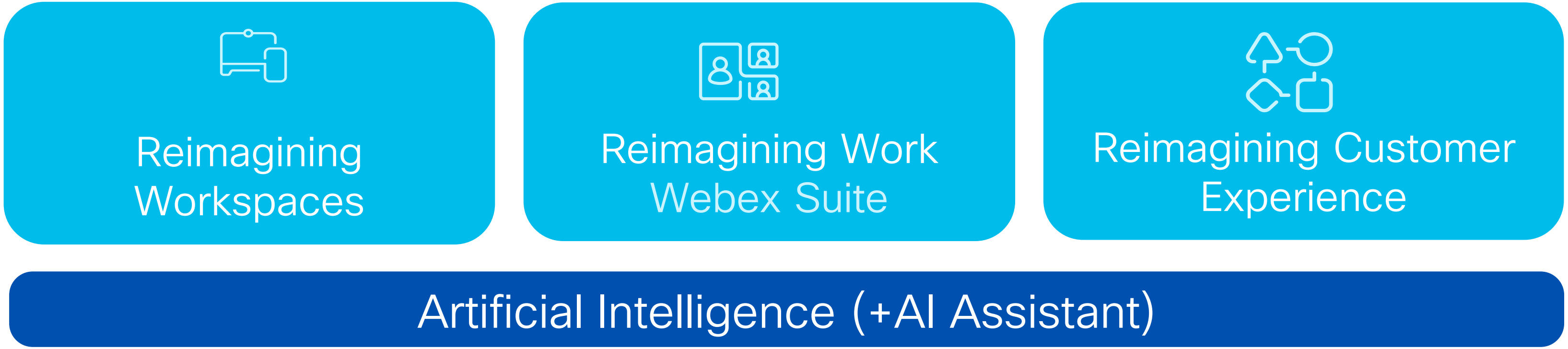
LLM

- Email
- Messages
- Transcripts
- Documents

RMM

- Audio
- Video
- Reactions
- Gestures
- Voice tones
- Inflections

Our Collaboration Strategy



Reimagining Workspaces

Reimagining Work Webex Suite

Reimagining Customer Experience

Artificial Intelligence (+AI Assistant)

Our Collaboration Strategy

Catch me up

Stay on top of what's going on

Meeting summary

Follow a meeting

Summarize in-meeting

Summarize all conversations

Why I was added

Be well prepared

Be ahead, effortlessly, for every interaction

Recommended action items

Prepare for the upcoming week

Communicate effectively

Speak with impact and confidence

Change tone & formatting (messaging)

Suggested reply to message

INTRODUCTION

Cisco AI Principles

Cisco's goal is to provide clarity and consistency in informing users when AI is employed in our technologies

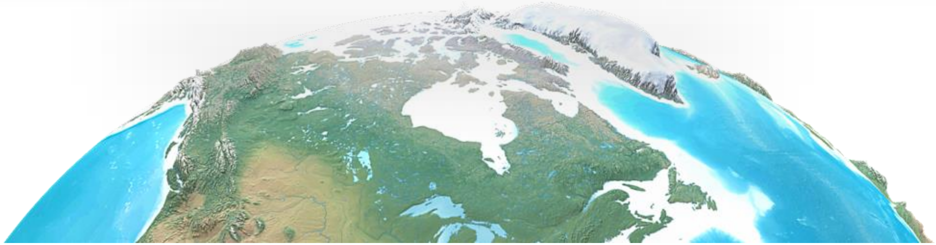
RESPONSIBLE AI- BUILT ON PRIVACY

CISCO PRINCIPLES FOR AI

CISCO AI FRAMEWORK

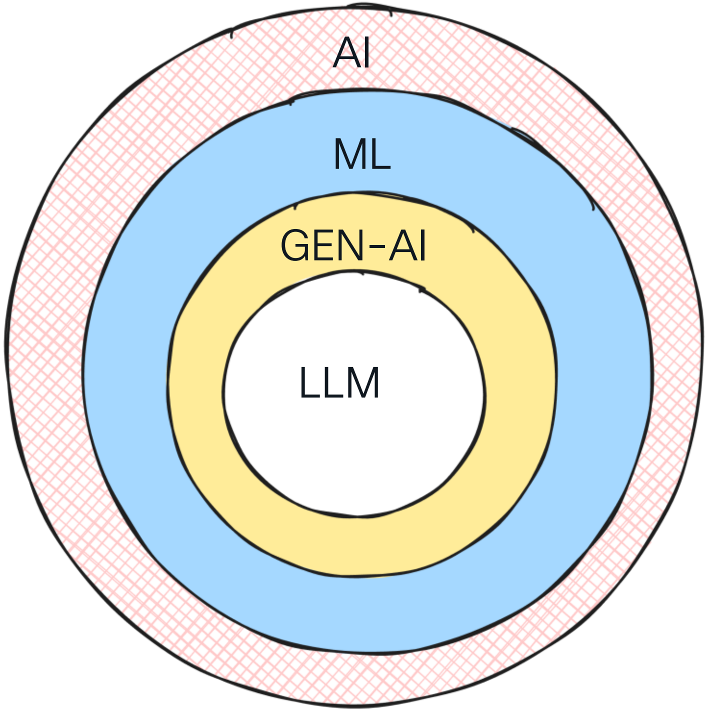
OUR RESPONSIBLE APPROACH

CISCO BLOGS ON AI



ARTIFICAL INTELLIGENCE - INTRODUCTION

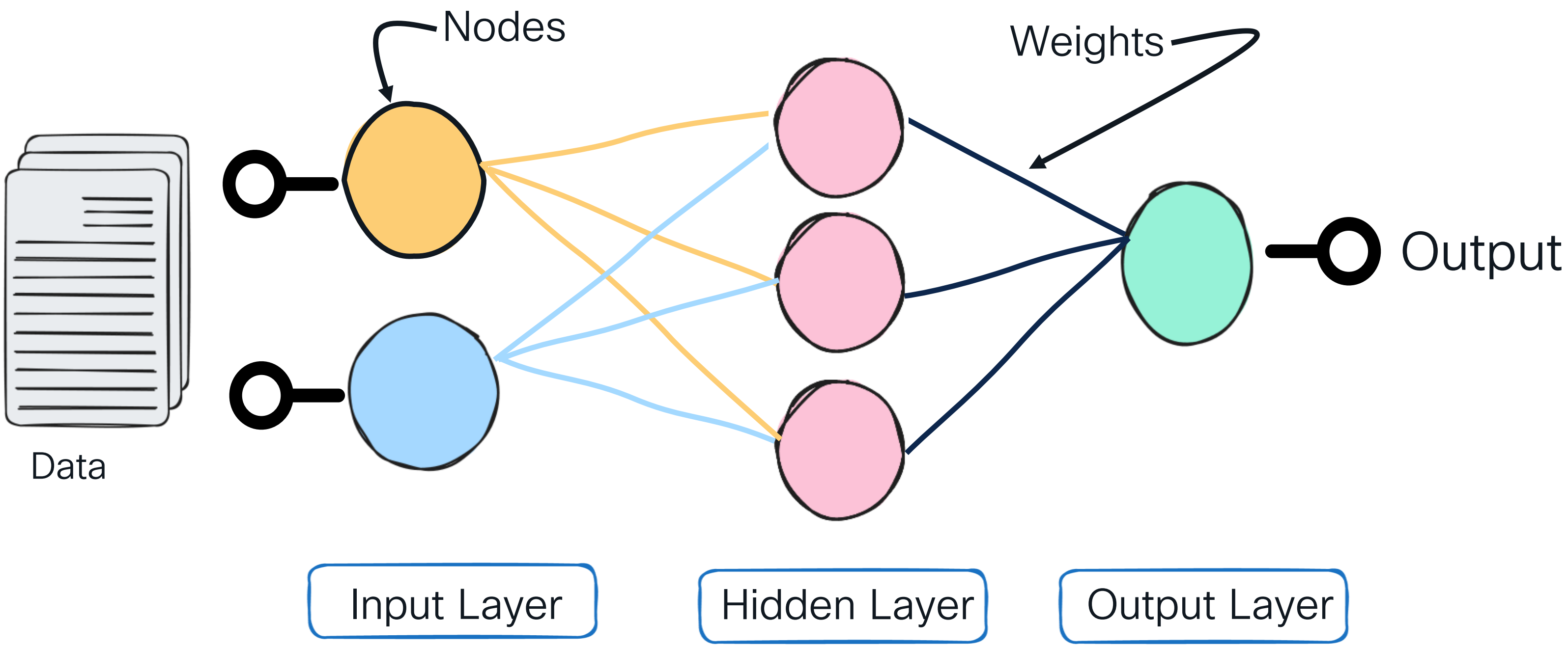
DIFFERENT TYPES OF AI



Neural Networks

Neural Network

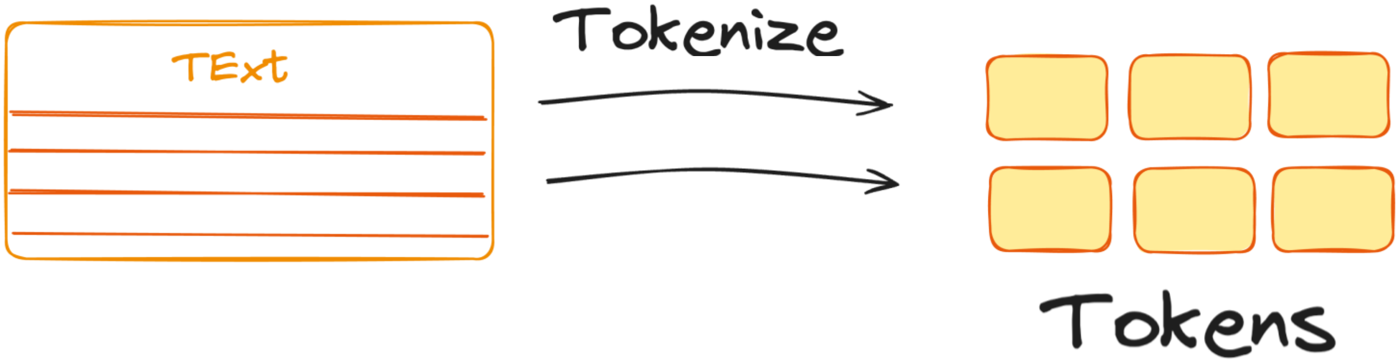
What is NN?



TOKENIZATION

Tokenization

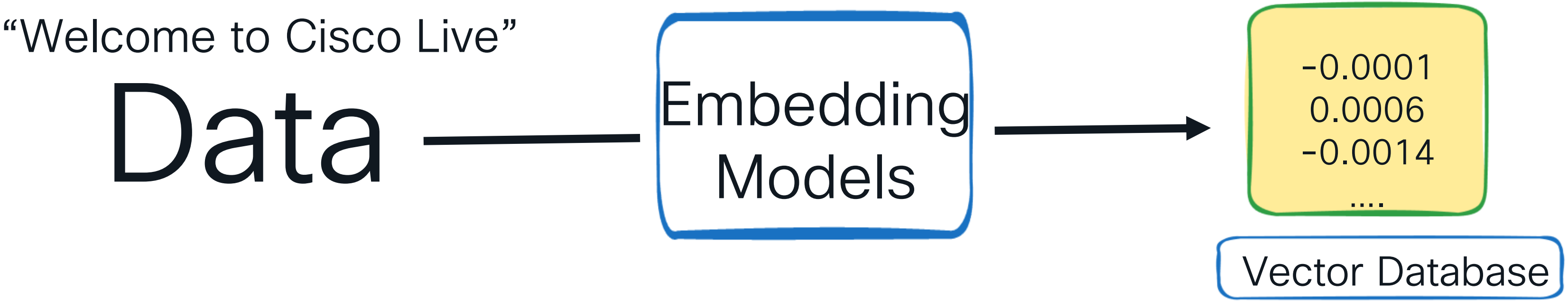
Different ways to Tokenize



Embeddings and Vector Database

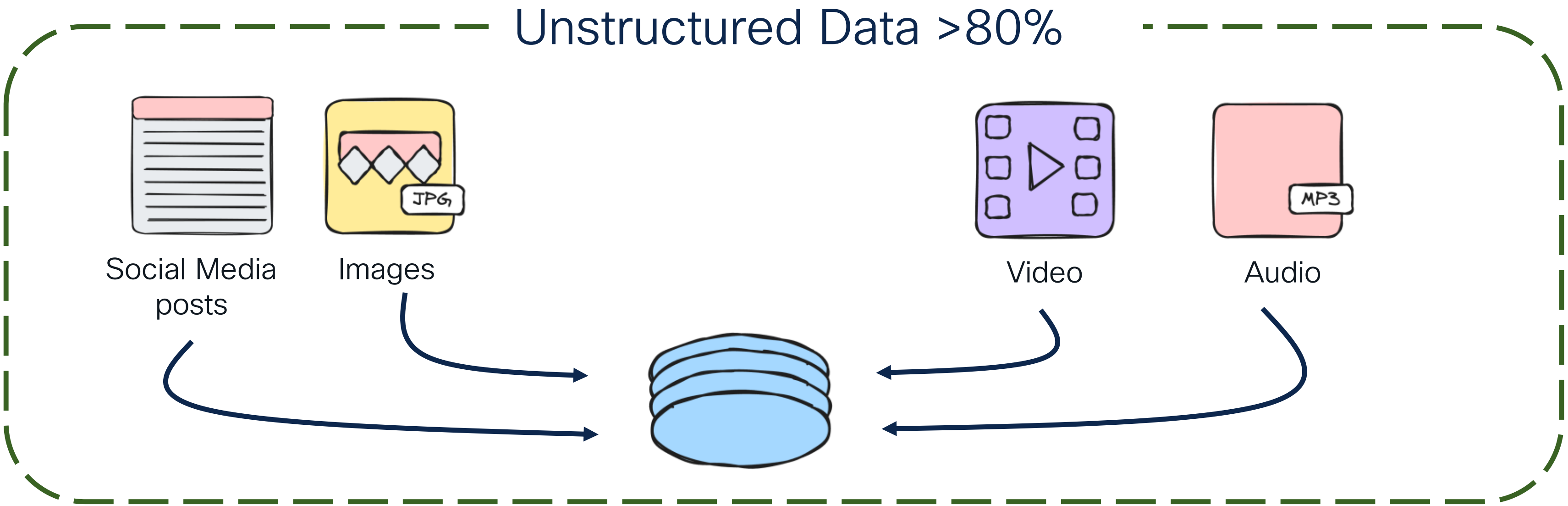
Embeddings and Vector Database

What are Embeddings?



Embeddings and Vector Database

Why Vector Database?



Embeddings and Vector Database

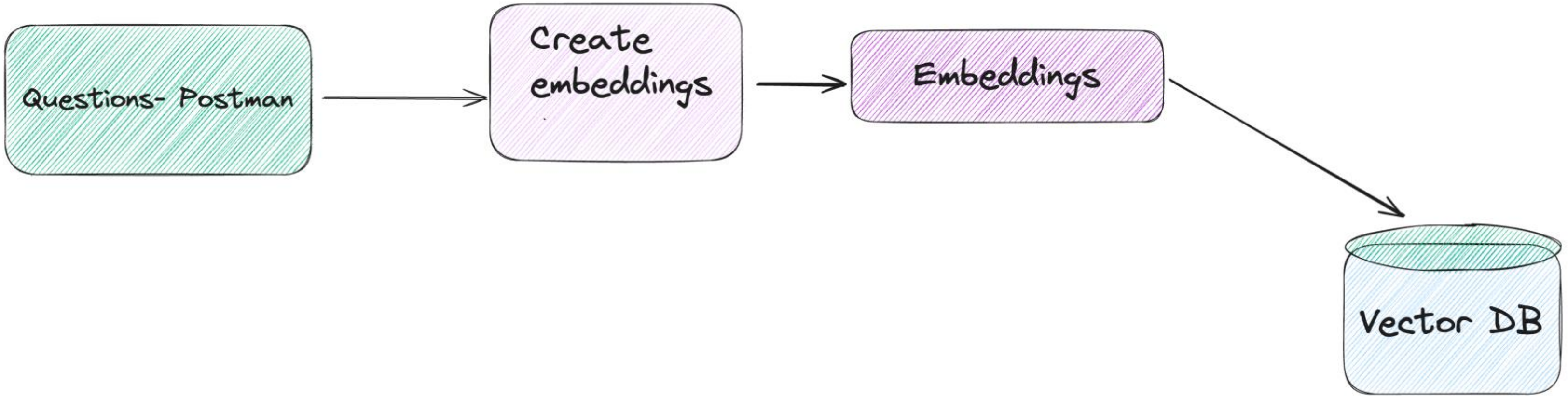
Why Vector Database?

Allow LLM to have long term memory.

Chroma   Pinecone   Single Store   Redis   Vespa

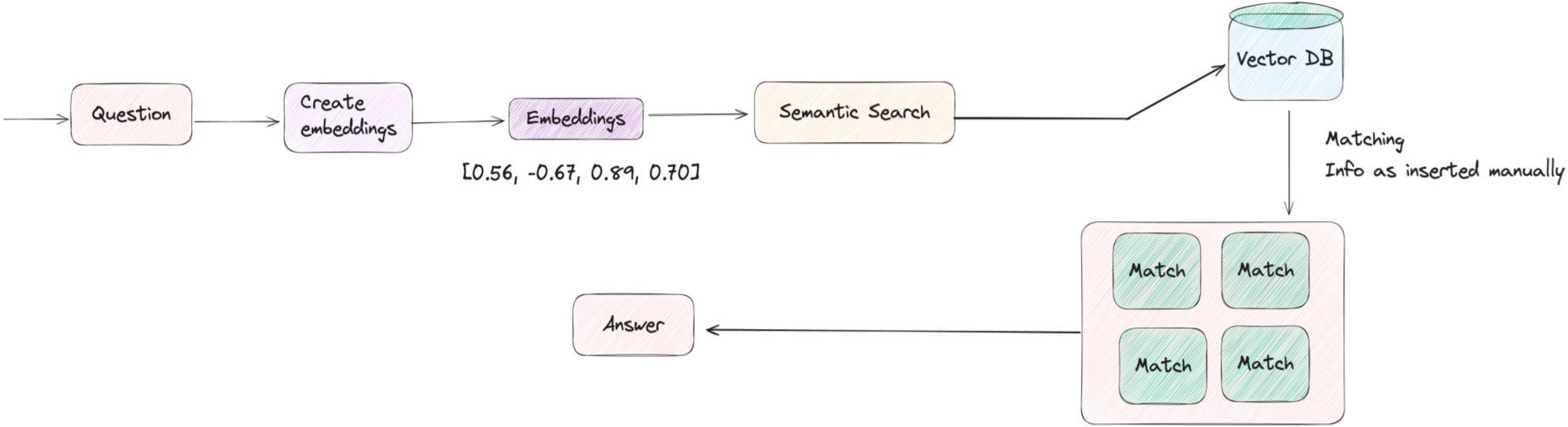
Embeddings and Vector Database

High Level Overview



Embeddings and Vector Database

High Level Overview



Retrieval Augmented Generation (RAG)

Retrieval Augmented Generation

Generation

↓

Response to user Query also known as Prompt

↓

Can have some undesirable behavior

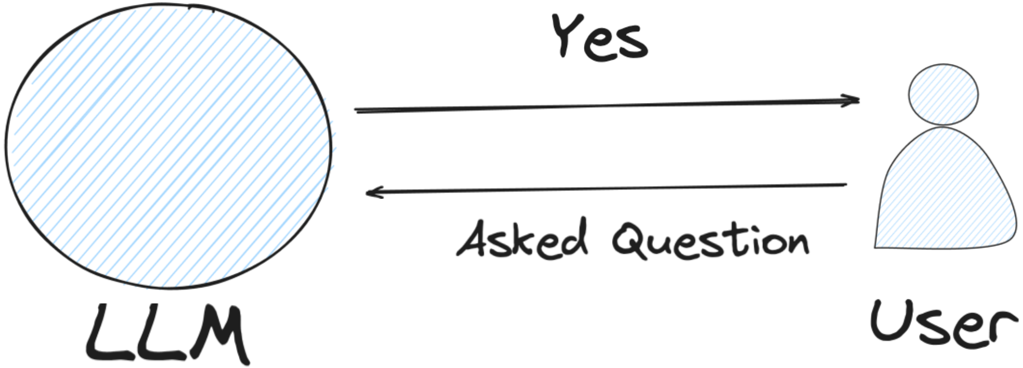
Cisco Devices

MTR?
ThousandEyes?

↓

Source

Retrieval Augmented Generation

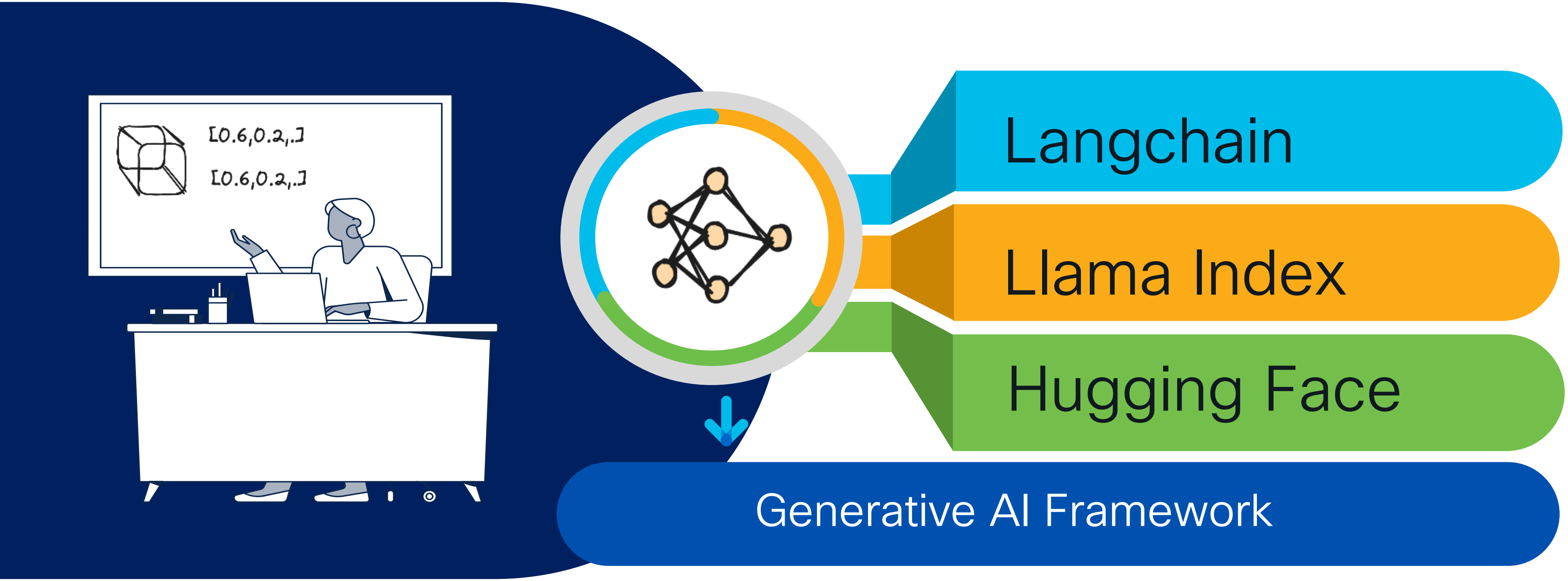


LLM Challenges

1. No Source
2. Out of date

Generative AI Framework

Generative AI Framework – LLM Models



Langchain

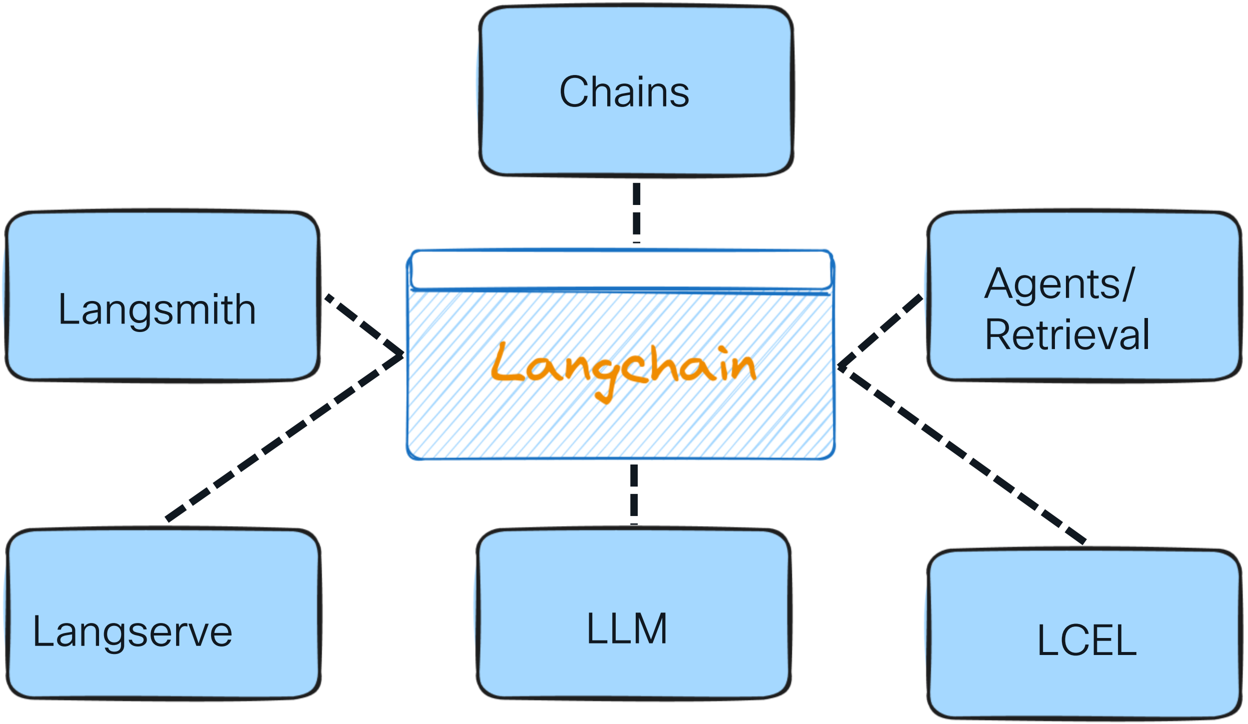
Llama Index

Hugging Face

Generative AI Framework

Langchain – Using Paid and Open Source Models

Generative AI Framework – Langchain


Chains

Langsmith

Agents/
Retrieval

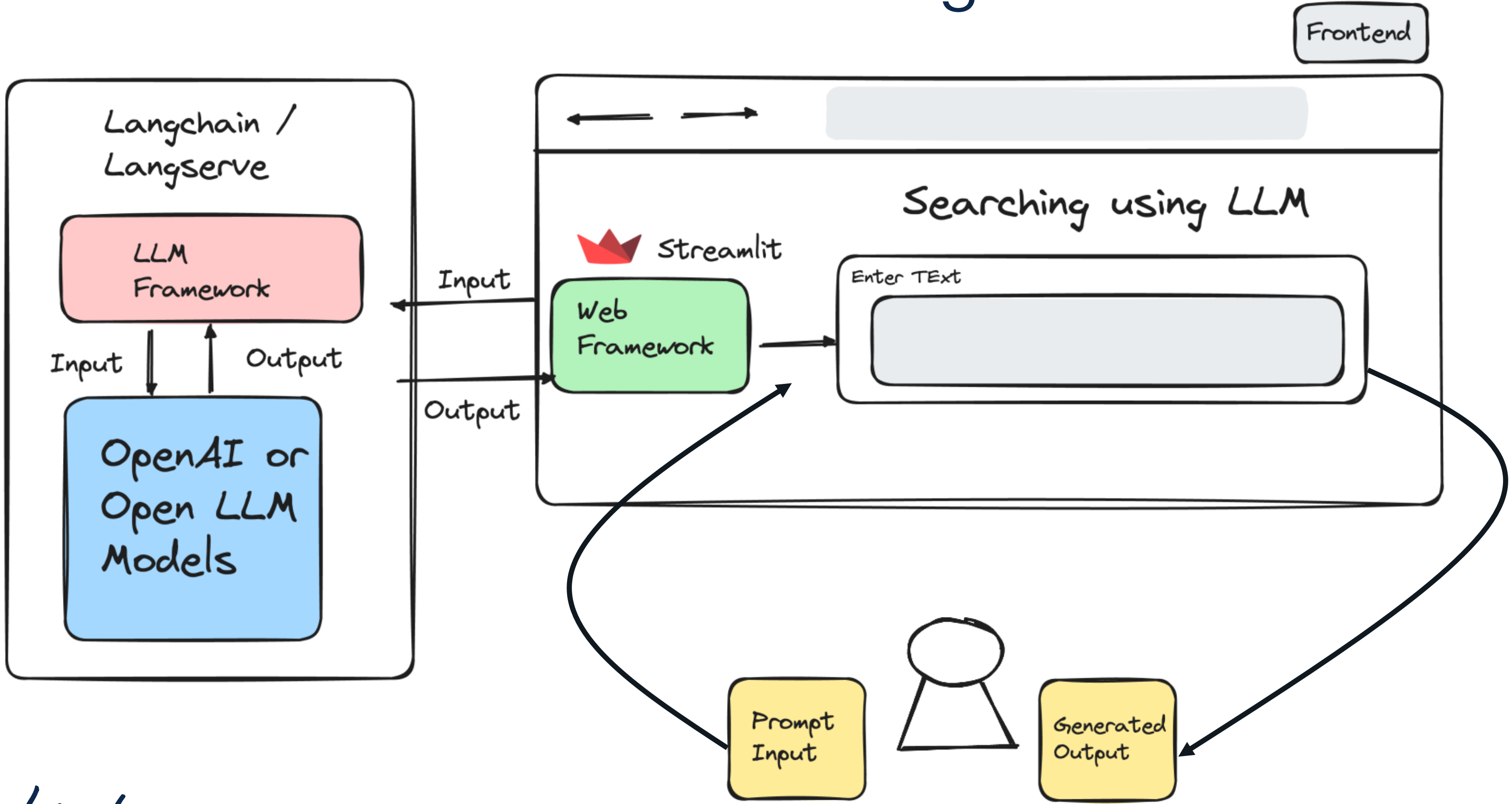
Langchain

Langserve

LLM

LCEL

Generative AI Framework – Langchain



Generative AI Framework – Langchain (OpenAI)

```
from langchain_openai import ChatOpenAI
# for chatbot
from langchain_core.prompts import ChatPromptTemplate
# Default ouput parser
from langchain_core.output_parsers import StrOutputParser
import streamlit as st
import os
from dotenv import load_dotenv
```

```
os.environ["OPENAI_API_KEY"]=os.getenv("OPENAI_API_KEY")
```

Generative AI Framework – Langchain (OpenAI)

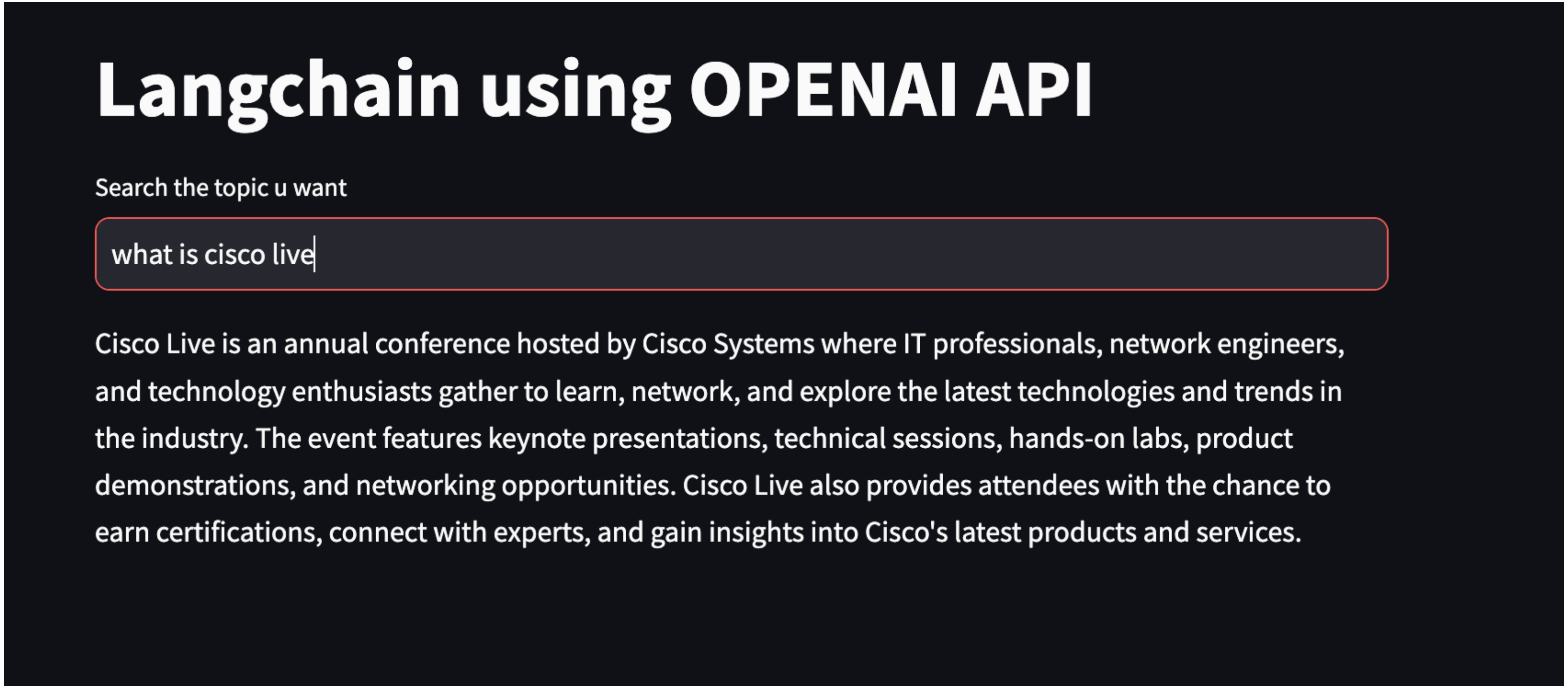
```
## Prompt Template
prompt=ChatPromptTemplate.from_messages(
[("system","You are a helpful Cisco Live assistant. Please respond to the user queries"),
("user","Question:{question}")])
```

```
## streamlit framework
st.title('Langchain using OPENAI API')
input_text=st.text_input("Search the topic u want")
```

```
# openAI LLm
llm=ChatOpenAI(model="gpt-3.5-turbo")
output_parser=StrOutputParser()
```

```
chain=prompt|llm|output_parser
if input_text:
st.write(chain.invoke({'question':input_text}))
```

Generative AI Framework – Langchain (OpenAI)



Langchain using OPENAI API

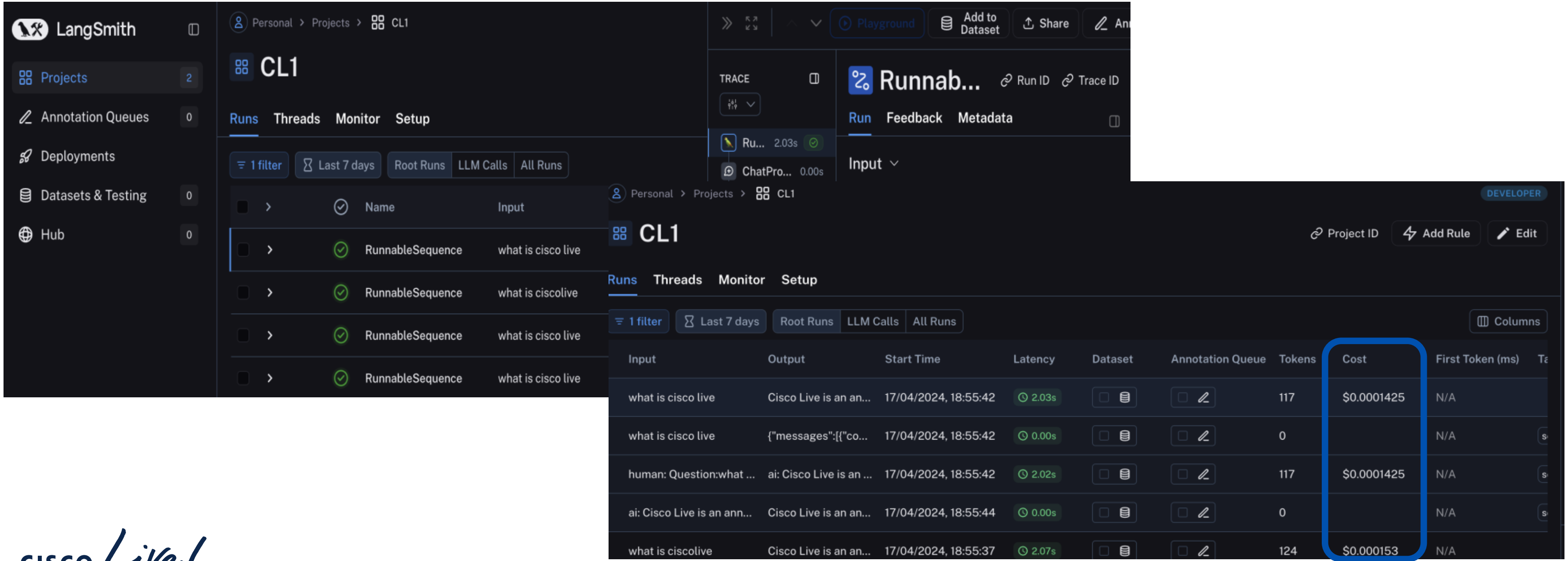
Search the topic u want

what is cisco live

Cisco Live is an annual conference hosted by Cisco Systems where IT professionals, network engineers, and technology enthusiasts gather to learn, network, and explore the latest technologies and trends in the industry. The event features keynote presentations, technical sessions, hands-on labs, product demonstrations, and networking opportunities. Cisco Live also provides attendees with the chance to earn certifications, connect with experts, and gain insights into Cisco's latest products and services.

Generative AI Framework – Langsmith

```
## Langmith tracking
os.environ["LANGCHAIN_TRACING_V2"]="true"
os.environ["LANGCHAIN_API_KEY"]=os.getenv("LANGCHAIN_API_KEY")
```



Name	Input
RunnableSequence	what is cisco live
RunnableSequence	what is ciscolive
RunnableSequence	what is cisco live
RunnableSequence	what is cisco live

Input	Output	Start Time	Latency	Dataset	Annotation Queue	Tokens	Cost	First Token (ms)	Ta
what is cisco live	Cisco Live is an an...	17/04/2024, 18:55:42	2.03s			117	$0.0001425	N/A	
what is cisco live	{"messages":[{"co...	17/04/2024, 18:55:42	0.00s			0		N/A	
human: Question:what ...	ai: Cisco Live is an ...	17/04/2024, 18:55:42	2.02s			117	$0.0001425	N/A	
ai: Cisco Live is an ann...	Cisco Live is an an...	17/04/2024, 18:55:44	0.00s			0		N/A	
what is ciscolive	Cisco Live is an an...	17/04/2024, 18:55:37	2.07s			124	$0.000153	N/A	

Generative AI Framework - Langchain (OpenSource Model)

```
from langchain_community.llms import Ollama
# for chatbot
from langchain_core.prompts import ChatPromptTemplate
# Default ouput parser
from langchain_core.output_parsers import StrOutputParser
import streamlit as st
import os
from dotenv import load_dotenv
```

```
## Langmith tracking
os.environ["LANGCHAIN_TRACING_V2"]="true"
os.environ["LANGCHAIN_API_KEY"]=os.getenv("LANGCHAIN_API_KEY")
```

Generative AI Framework – Langchain (OpenSource Model)

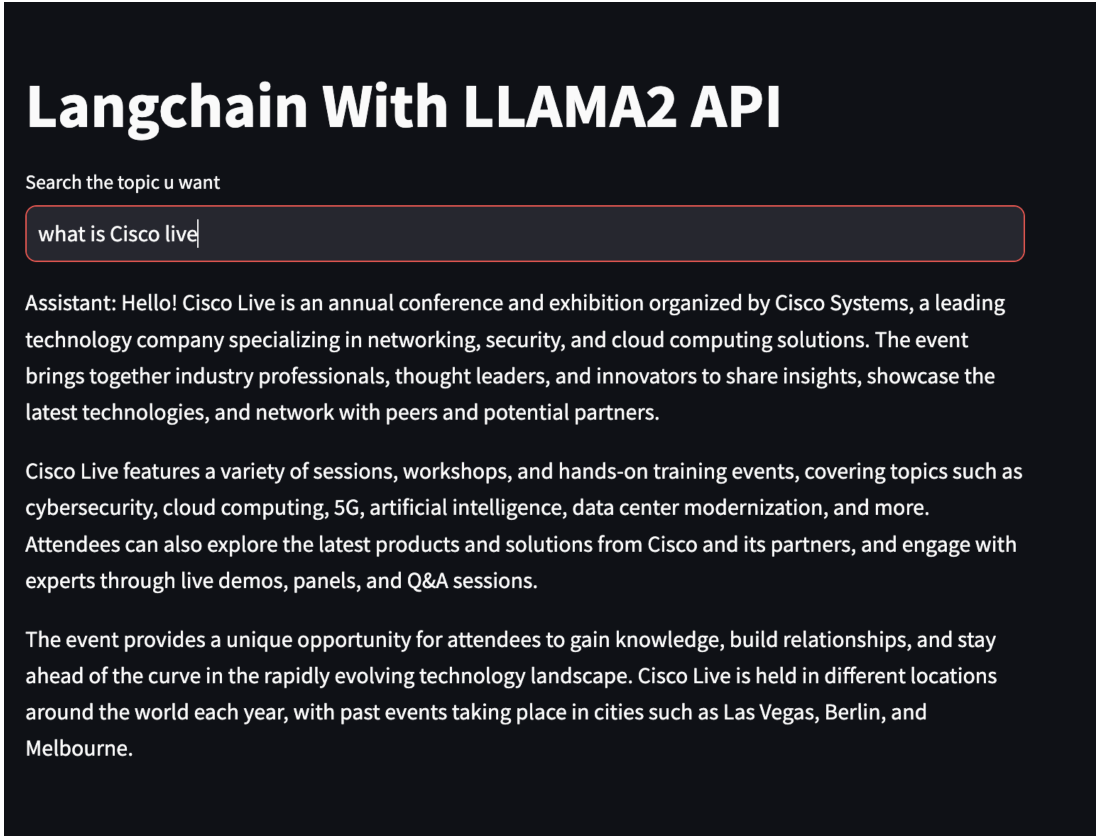
```
## Prompt Template
prompt=ChatPromptTemplate.from_messages(
[("system","You are a helpful Cisco Live assistant. Please respond to the user queries"),
("user","Question:{question}")])
```

```
## streamlit framework
st.title('Langchain using Llama2')
input_text=st.text_input("Search the topic u want")
```

```
# openAI LLm
llm=Ollama(model="llama2")
output_parser=StrOutputParser()
```

```
chain=prompt|llm|output_parser
if input_text:
st.write(chain.invoke({'question':input_text}))
```

Generative AI Framework – Langchain (OpenSource Model)



Langchain With LLAMA2 API

Search the topic u want

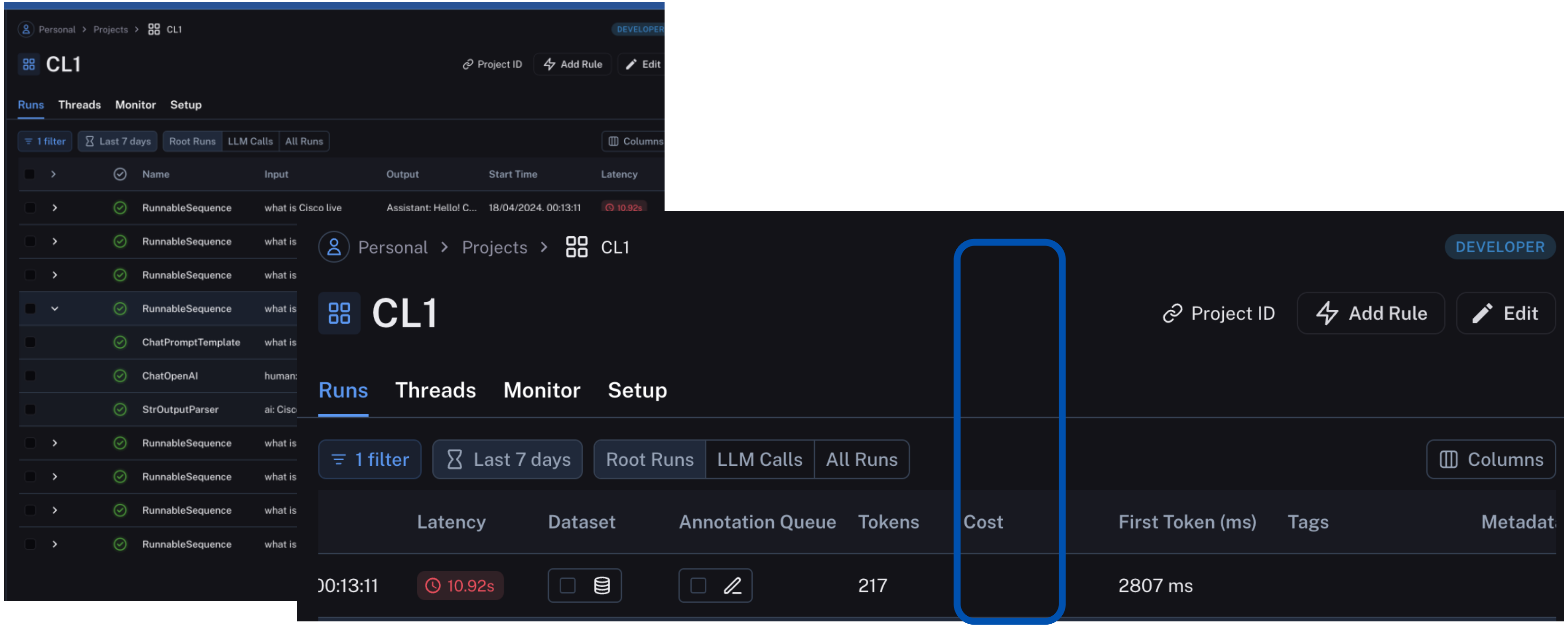
what is Cisco live

Assistant: Hello! Cisco Live is an annual conference and exhibition organized by Cisco Systems, a leading technology company specializing in networking, security, and cloud computing solutions. The event brings together industry professionals, thought leaders, and innovators to share insights, showcase the latest technologies, and network with peers and potential partners.

Cisco Live features a variety of sessions, workshops, and hands-on training events, covering topics such as cybersecurity, cloud computing, 5G, artificial intelligence, data center modernization, and more. Attendees can also explore the latest products and solutions from Cisco and its partners, and engage with experts through live demos, panels, and Q&A sessions.

The event provides a unique opportunity for attendees to gain knowledge, build relationships, and stay ahead of the curve in the rapidly evolving technology landscape. Cisco Live is held in different locations around the world each year, with past events taking place in cities such as Las Vegas, Berlin, and Melbourne.

Generative AI Framework – Langsmith

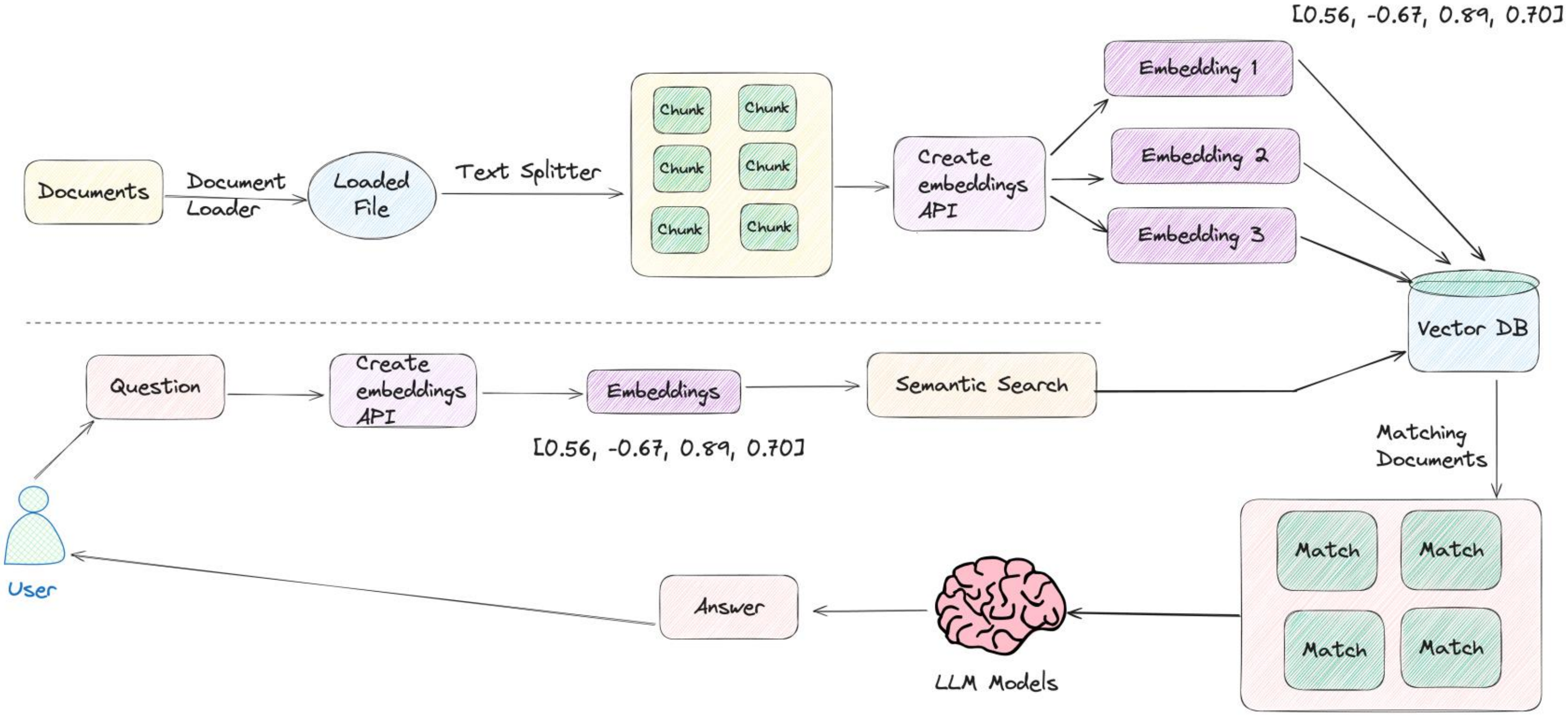


Name	Input	Output	Start Time	Latency
RunnableSequence	what is Cisco live	Assistant: Hello! C...	18/04/2024, 00:13:11	10.92s
RunnableSequence	what is			
RunnableSequence	what is			
RunnableSequence	what is			
ChatPromptTemplate	what is			
ChatOpenAI	human:			
StrOutputParser	ai: Cisc			
RunnableSequence	what is			
RunnableSequence	what is			
RunnableSequence	what is			
RunnableSequence	what is			

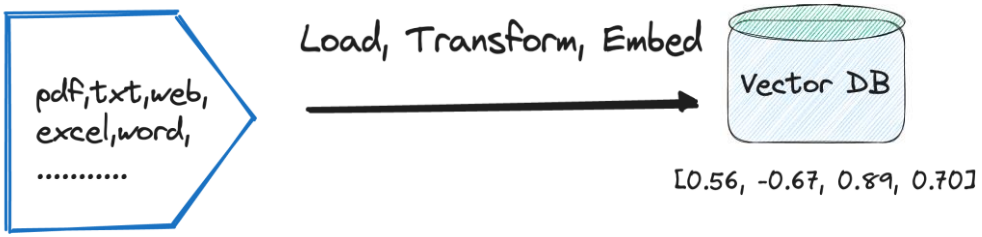
	Latency	Dataset	Annotation Queue	Tokens	Cost	First Token (ms)	Tags	Metadat
)0:13:11	10.92s	☐	☐	217		2807 ms		

Langchain - Demo2

Generative AI Framework – Call Flow



Generative AI Framework – Langchain



Step 1: Load data source also called Data ingestion

Step 2: Transform, where we break data into small chunks

Step 3: Convert Chunks into vectors also called Embeddings

Step 4: Save in Vector Database

Entire RAG pipeline

Generative AI Framework – Langchain

```
# Data ingestion Technique #1
from
langchain_community.document_loaders
import TextLoader
loader = TextLoader("calling.txt")
text_documents = loader.load()
text_documents
```

```
# web based loader - Data ingestion Technique #2
from langchain_community.document_loaders import WebBaseLoader
import bs4
## load,chunk and index the content of the html page
loader=WebBaseLoader(web_paths=("https://github.com/WebexSamples",)
,
bs_kwargs=dict(parse_only=bs4.SoupStrainer(
class_=("heading-element","markdown-heading"))))
text_documents=loader.load()
text_documents
```

```
# pdf based loader - Data ingestion Technique #3
from langchain_community.document_loaders import PyPDFLoader
loader=PyPDFLoader("webex_calling.pdf")
docs=loader.load()
docs
```

Generative AI Framework – Langchain

```
# Lets now move to the Transform part
from langchain.text_splitter import RecursiveCharacterTextSplitter
text_splitter=RecursiveCharacterTextSplitter(chunk_size=1000, chunk_overlap=200)
documents=text_splitter.split_documents(docs)
documents
```

Generative AI Framework – Langchain

```
# Lets now move to Embeddings, Convert text into vectors –
We can do Embeddings with respect to Openai or Llama
from langchain_openai import OpenAIEmbeddings
from langchain_community.vectorstores import Chroma
db = Chroma.from_documents(documents,OpenAIEmbeddings())
db
```

OR

```
# Lets now move to Embeddings, Convert text into vectors –
We can do Embeddings with respect to Openai or Llama
from langchain_openai import OpenAIEmbeddings
from langchain_community.vectorstores import FAISS
db1 = FAISS.from_documents(documents,OpenAIEmbeddings())
db1
```

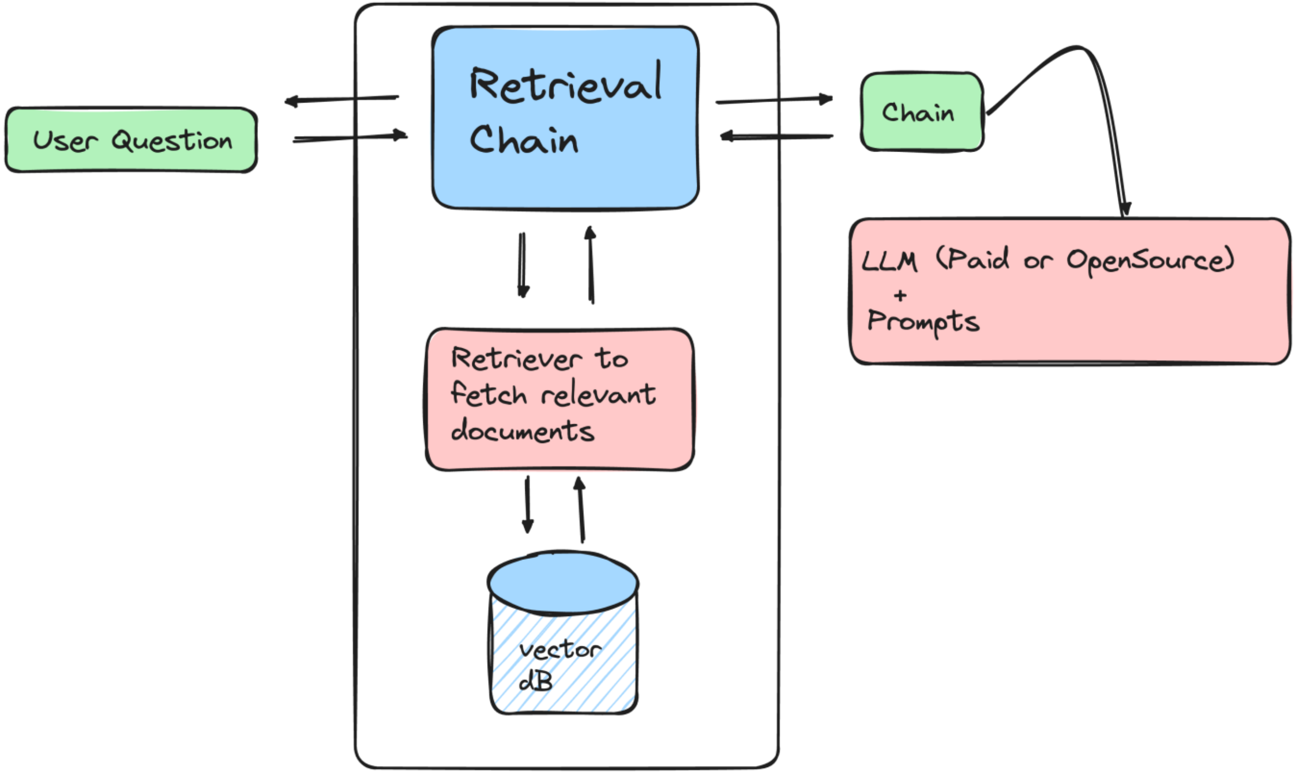
Generative AI Framework – Langchain

```
# Query dB
query = "The Private Network Connect (PNC) feature allows"
result = db.similarity_search(query)
print(result[0].page_content)
```

Query using
Similarity Search

Langchain - Demo3

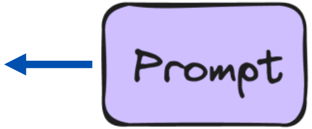
Generative AI Framework – Chains and Retrievers



Generative AI Framework – Langchain

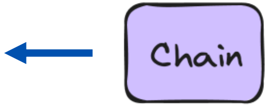
```
from langchain_community.llms import Ollama
## Load Ollama LAMA2 LLM model
llm=Ollama(model="llama2")
```

```
from langchain_core.prompts import ChatPromptTemplate
prompt= ChatPromptTemplate.from_template("""
Answer the following question based only on the provided context. If
no answer is available just say I don’t know.
<context>
{context}
</context>
Question: {input}""")
```



Generative AI Framework – Langchain

```
## Chain Introduction
## Create Docment Chain
from langchain.chains.combine_documents import create_stuff_documents_chain
document_chain=create_stuff_documents_chain(llm,prompt)
```



```
"""
Retrievers: A retriever is an interface that returns documents given
an unstructured query. It is more general than a vector store.
A retriever does not need to be able to store documents, only to
return (or retrieve) them. Vector stores can be used as the backbone
of a retriever, but there are other types of retrievers as well.
https://python.langchain.com/docs/modules/data_connection/retrievers/
"""

retriever=db.as_retriever()
retriever
```

Generative AI Framework – Langchain

```
"""
Retrieval chain:This chain takes in a user inquiry, which is then
passed to the retriever to fetch relevant documents. Those documents
(and original inputs) are then passed to an LLM to generate a response
https://python.langchain.com/docs/modules/chains/
"""
from langchain.chains import create_retrieval_chain
retrieval_chain=create_retrieval_chain(retriever,document_chain)
```

Retriever Chain

```
response=retrieval_chain.invoke({"input":"Webex Calling Customer Direct Connect"})
```

Question

Continue your education

- Visit the Cisco Showcase for related demos
- Book your one-on-one Meet the Engineer meeting
- Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs
- Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand

Contact me at: oilyas@cisco.com

The bridge to possible

Thank you

#CiscoLive