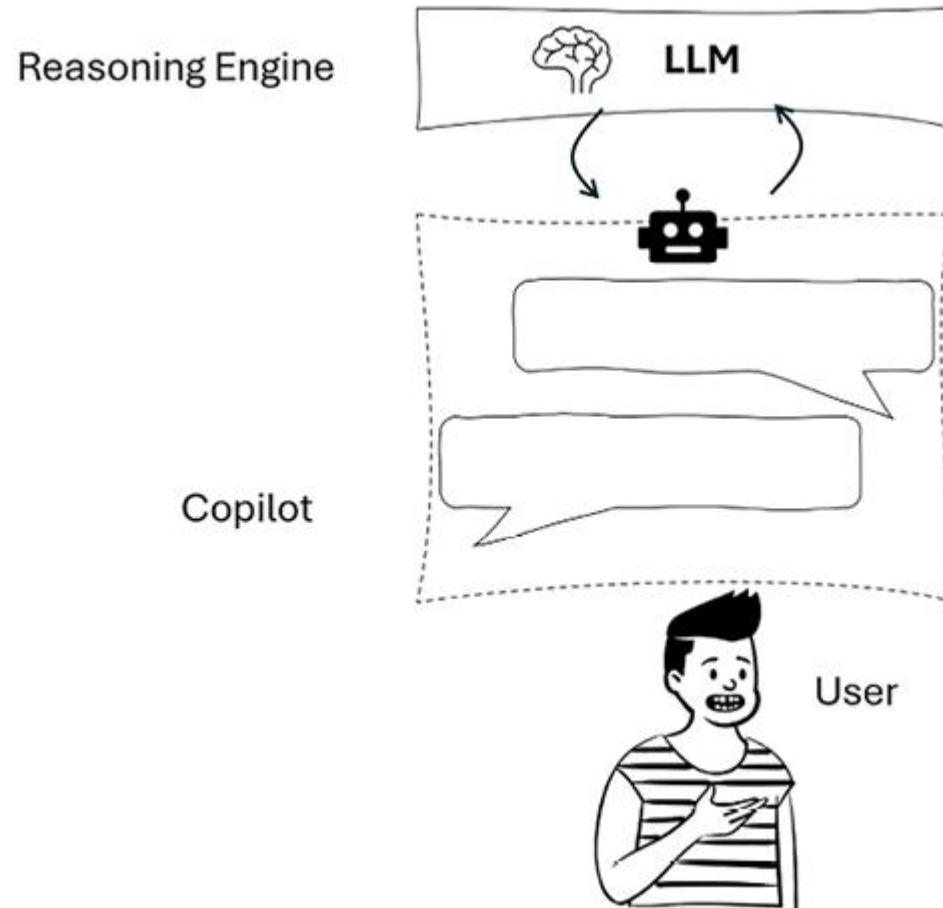


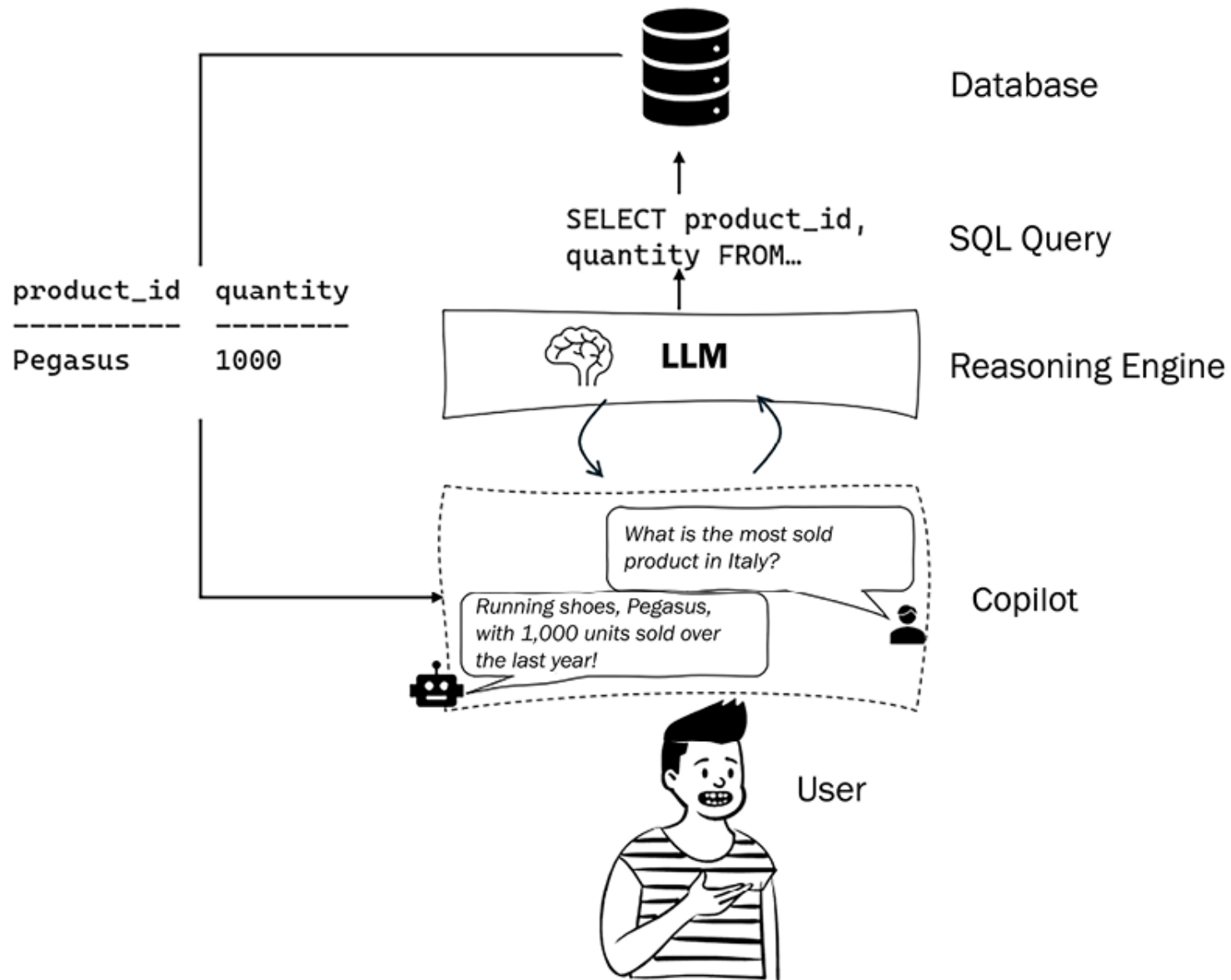
# Enterprise Essentials

## Generative AI – 1

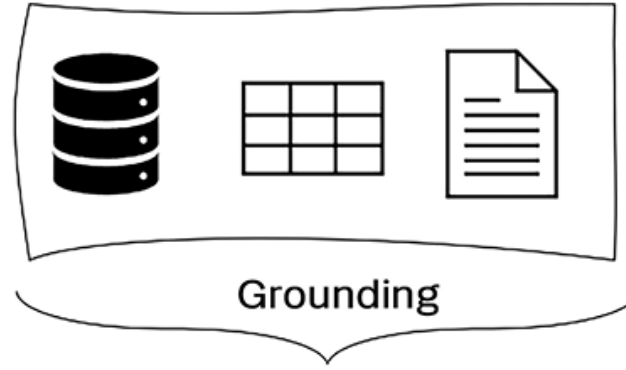


**A copilot is designed to have a conversational user interface,** allowing users to interact with it using natural language.

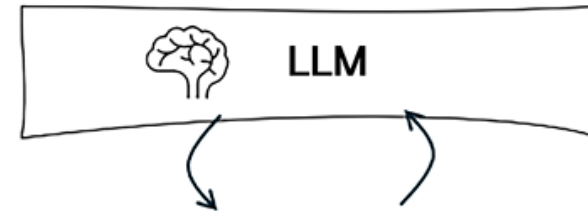




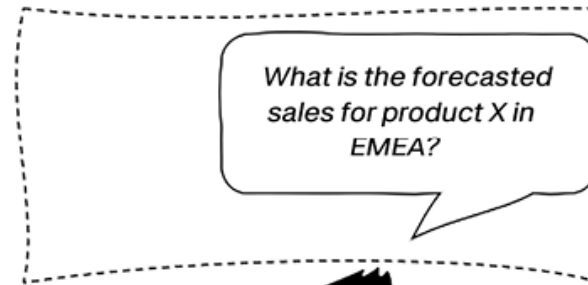
App data



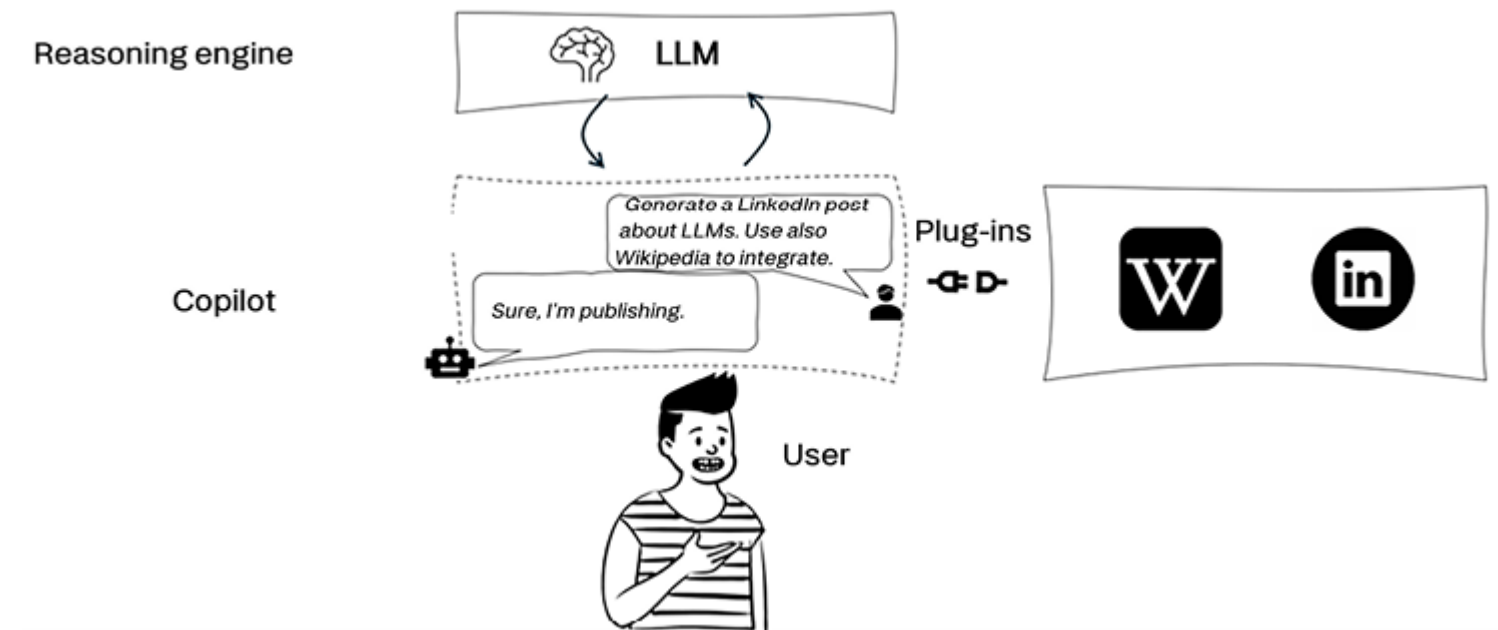
Reasoning engine



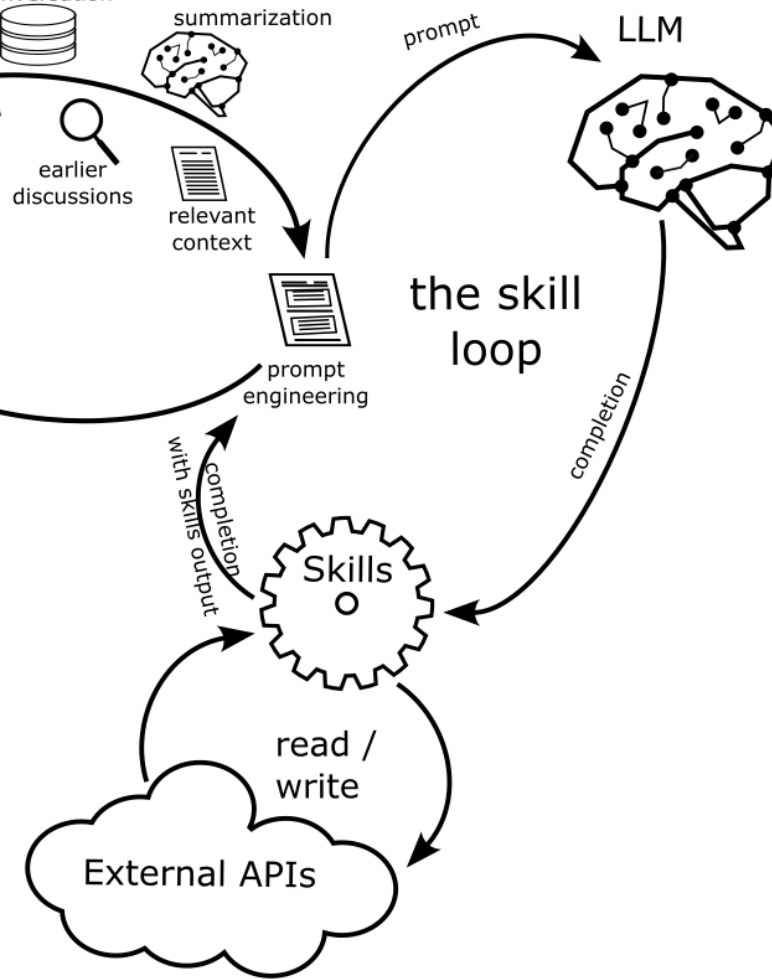
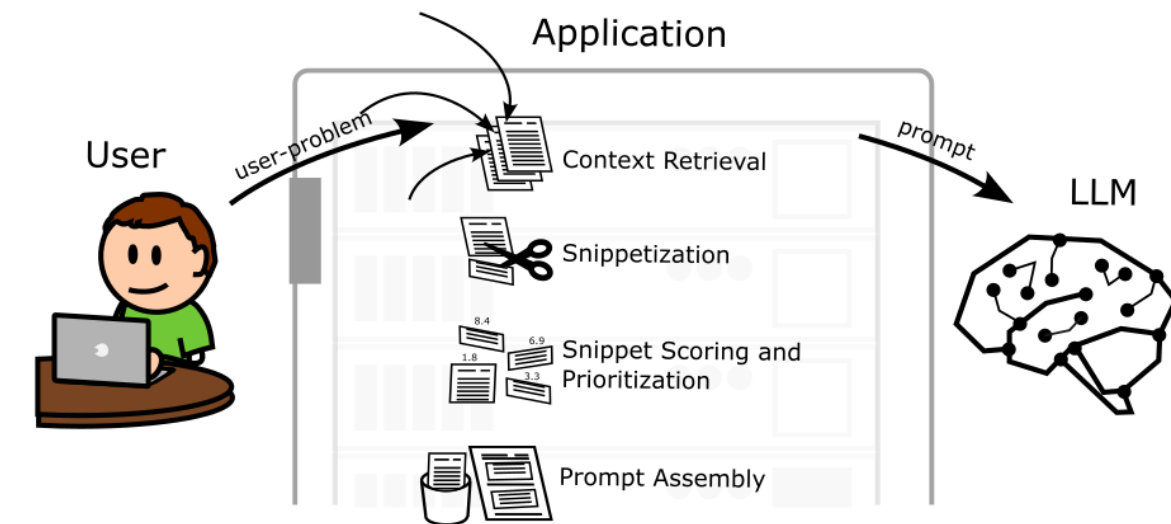
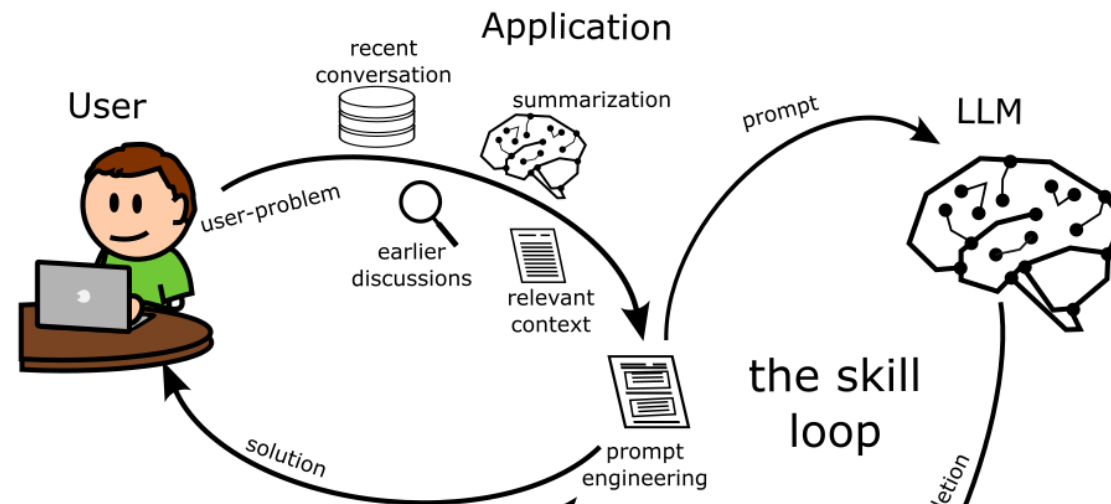
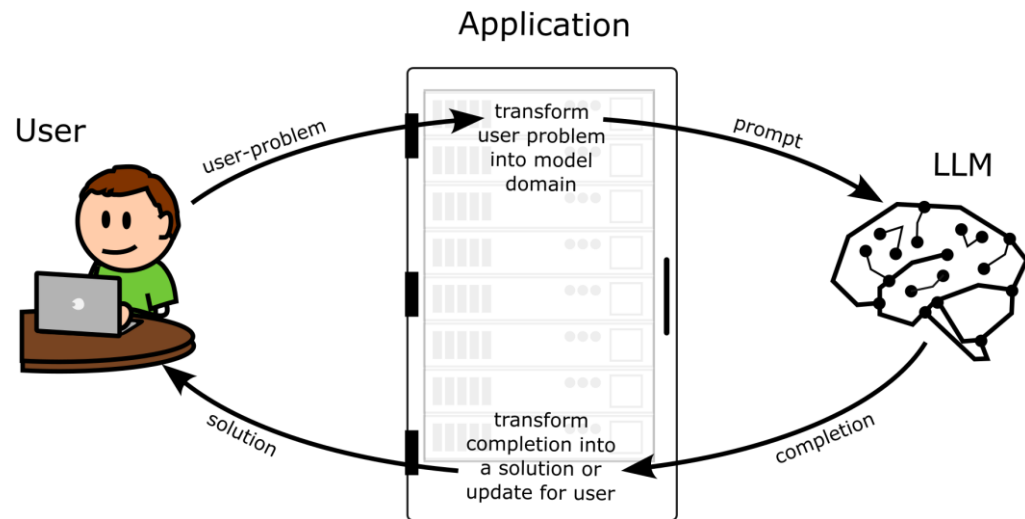
Copilot



User



*Wikipedia and LinkedIn plug-ins*



WU

Can you recommend a book to me? The last two books I read were "Moby Dick" and "Huckleberry Finn".



Based on the classic nature of the books you mentioned, I would recommend you read "To Kill a Mockingbird" by Harper Lee. It shares themes of adventure, friendship, and social issues, much like "Huckleberry Finn." "To Kill a Mockingbird" explores racial injustice and moral growth through the eyes of Scout Finch, a young girl living in the fictional town of Maycomb, Alabama. It has been widely acclaimed for its poignant storytelling and compelling characters. I believe you would find it engaging and thought-provoking, given your interest in classic literature.

WU

Can you recommend a book to me? The last two books I read were "Moby Dick" and "Huckleberry Finn".

Other facts about me: I'm 38, like to watch Vietnam War documentaries, for my last vacation I went to Thailand and I loved it there.



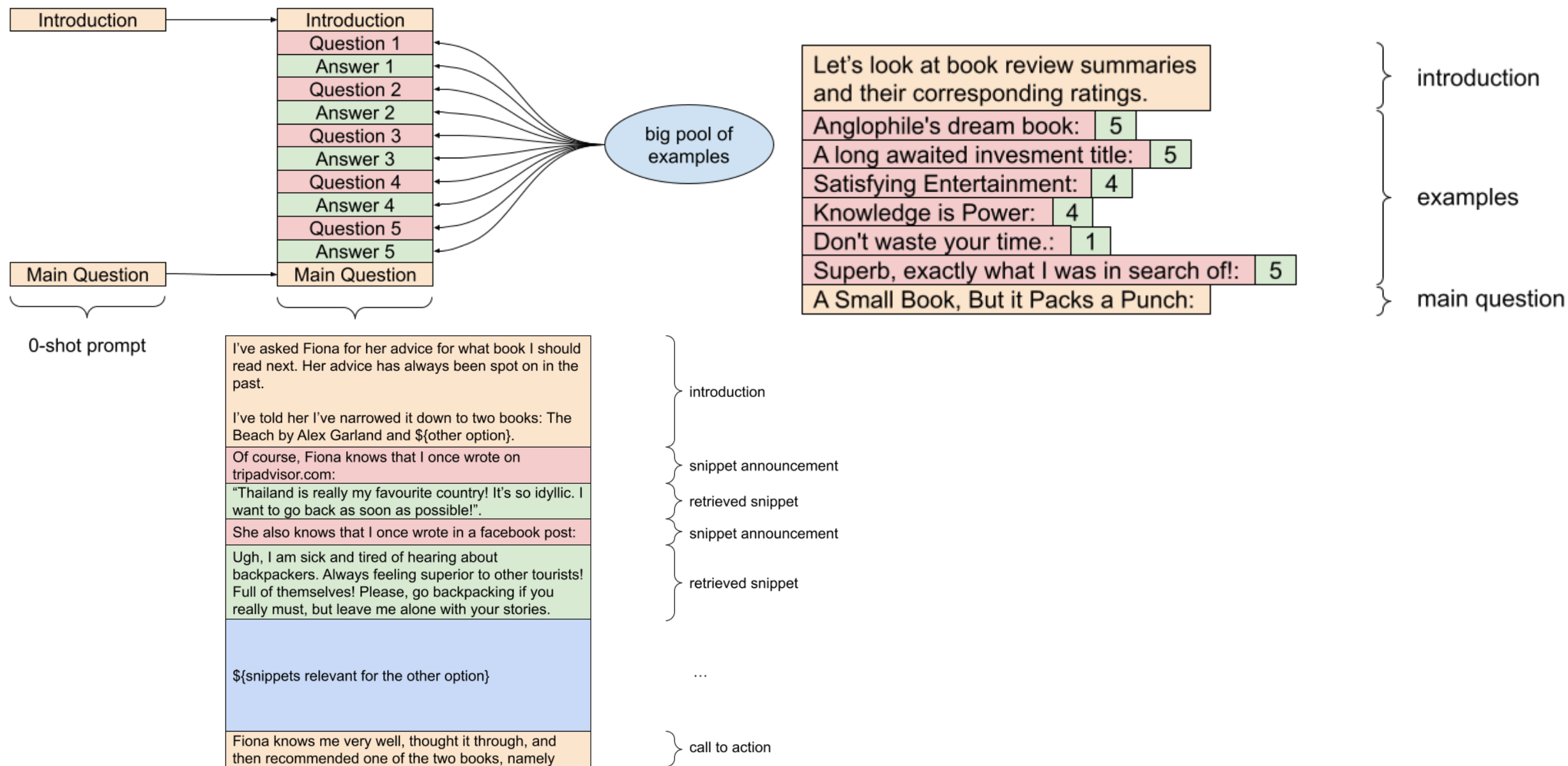
Based on your preferences and interests, I would recommend the book "The Things They Carried" by Tim O'Brien. This novel draws upon O'Brien's personal experiences as a soldier in the Vietnam War, offering a unique and compelling perspective on the conflict. It explores the physical and emotional burdens carried by soldiers and delves into themes of war, memory, truth, and storytelling. The book combines war narrative with elements of memoir and fiction, creating a powerful and thought-provoking read.

Additionally, considering your enjoyment of Thailand, you might find "The Beach" by Alex Garland intriguing. This novel follows the journey of a young traveler who discovers a hidden, idyllic beach in Thailand. It explores themes of adventure, escapism, and the dark side of paradise. It provides an engaging story set in a location you found appealing.

Both of these books offer captivating narratives and explore different aspects of human experiences and journeys. I hope you find them enjoyable!

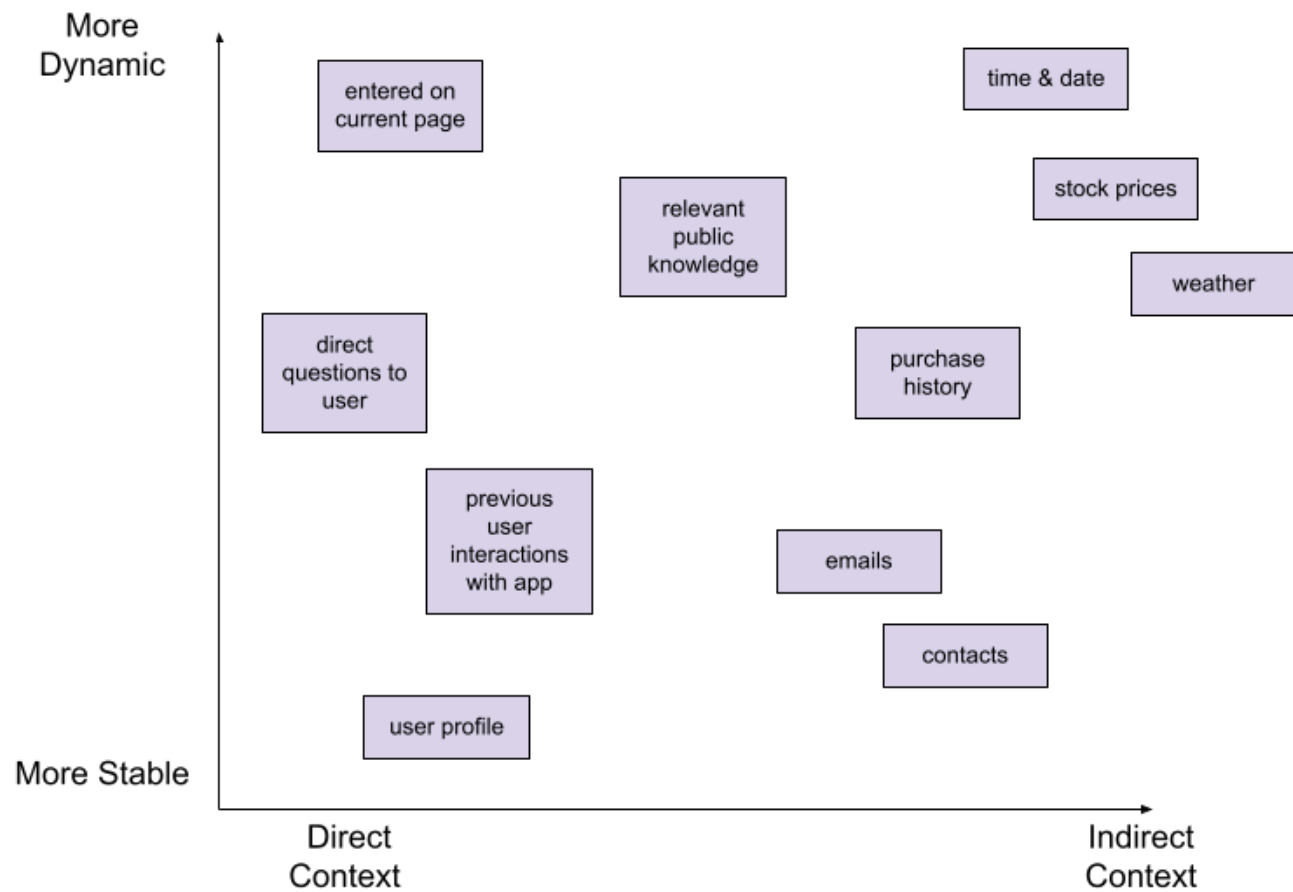
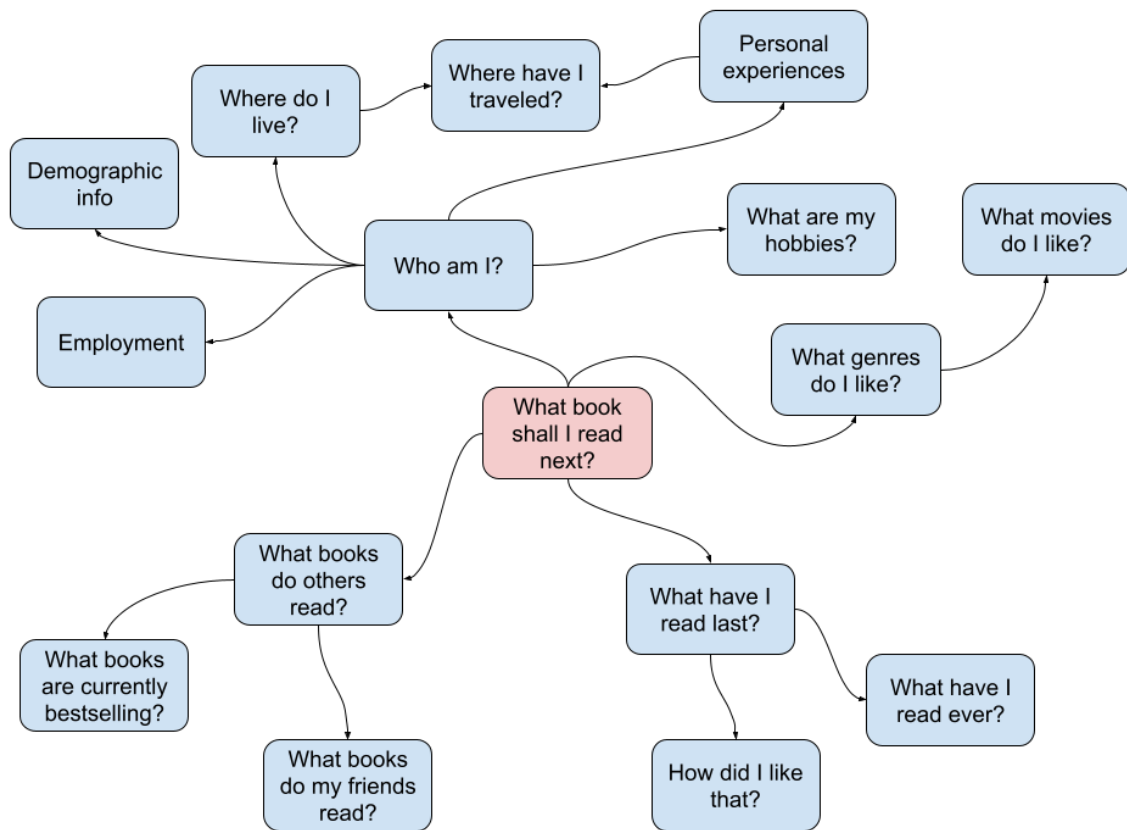
Asking ChatGPT for a book recommendation, first without context (left) and then with (right)

## Few-Shot Prompting

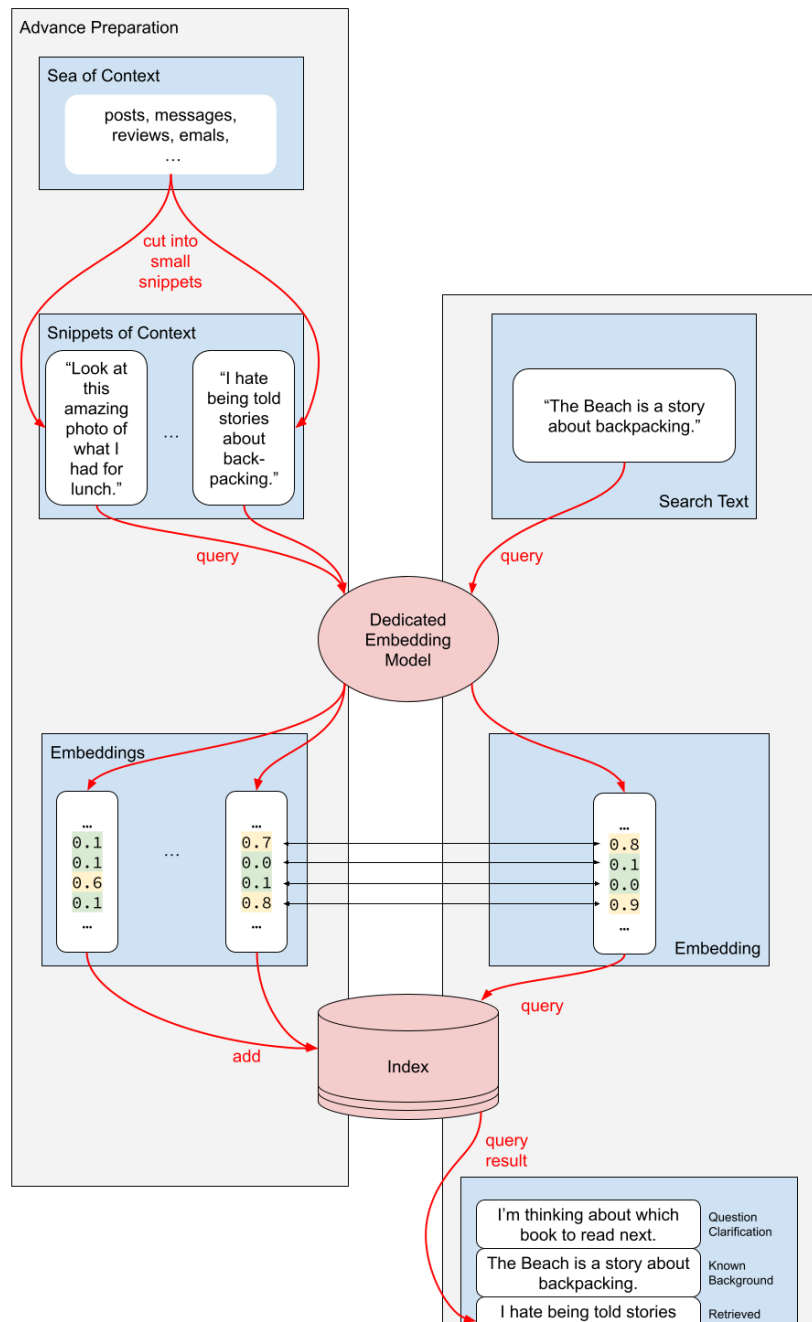




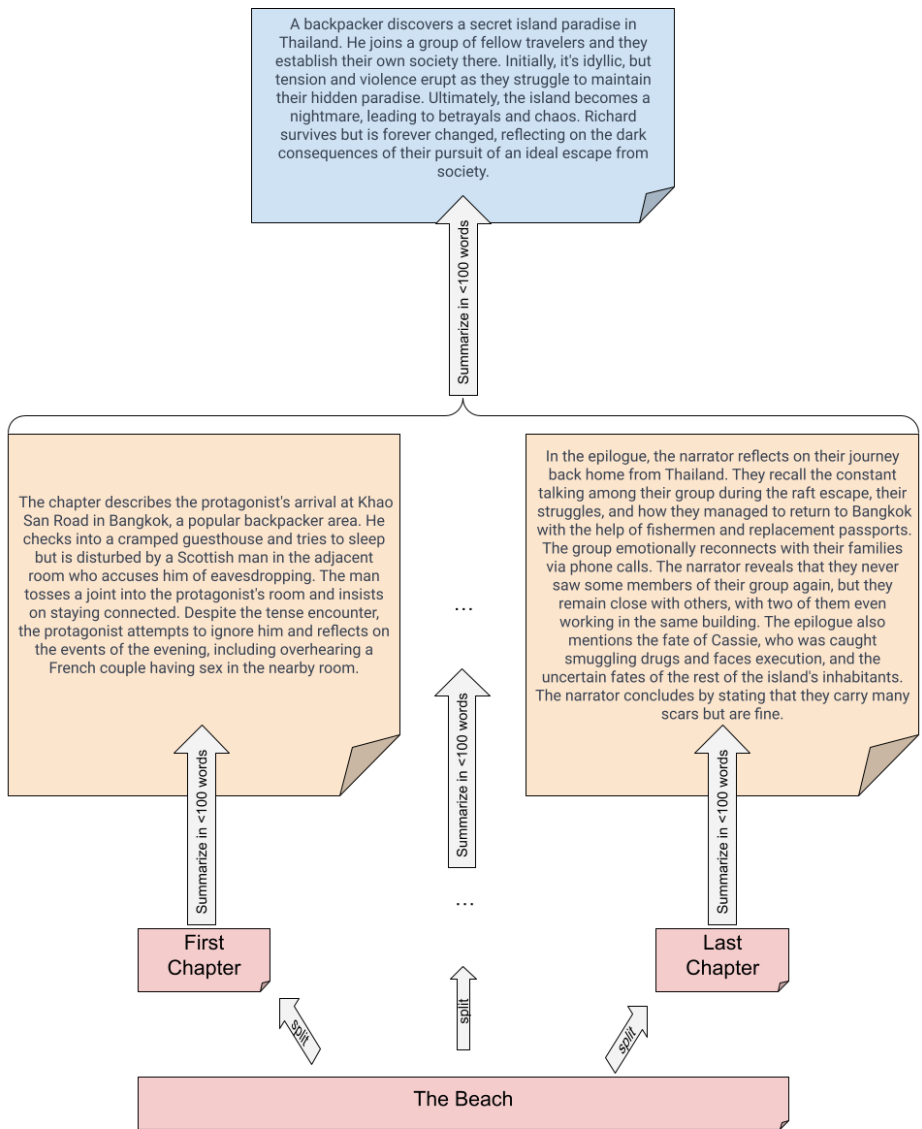
## Finding Context

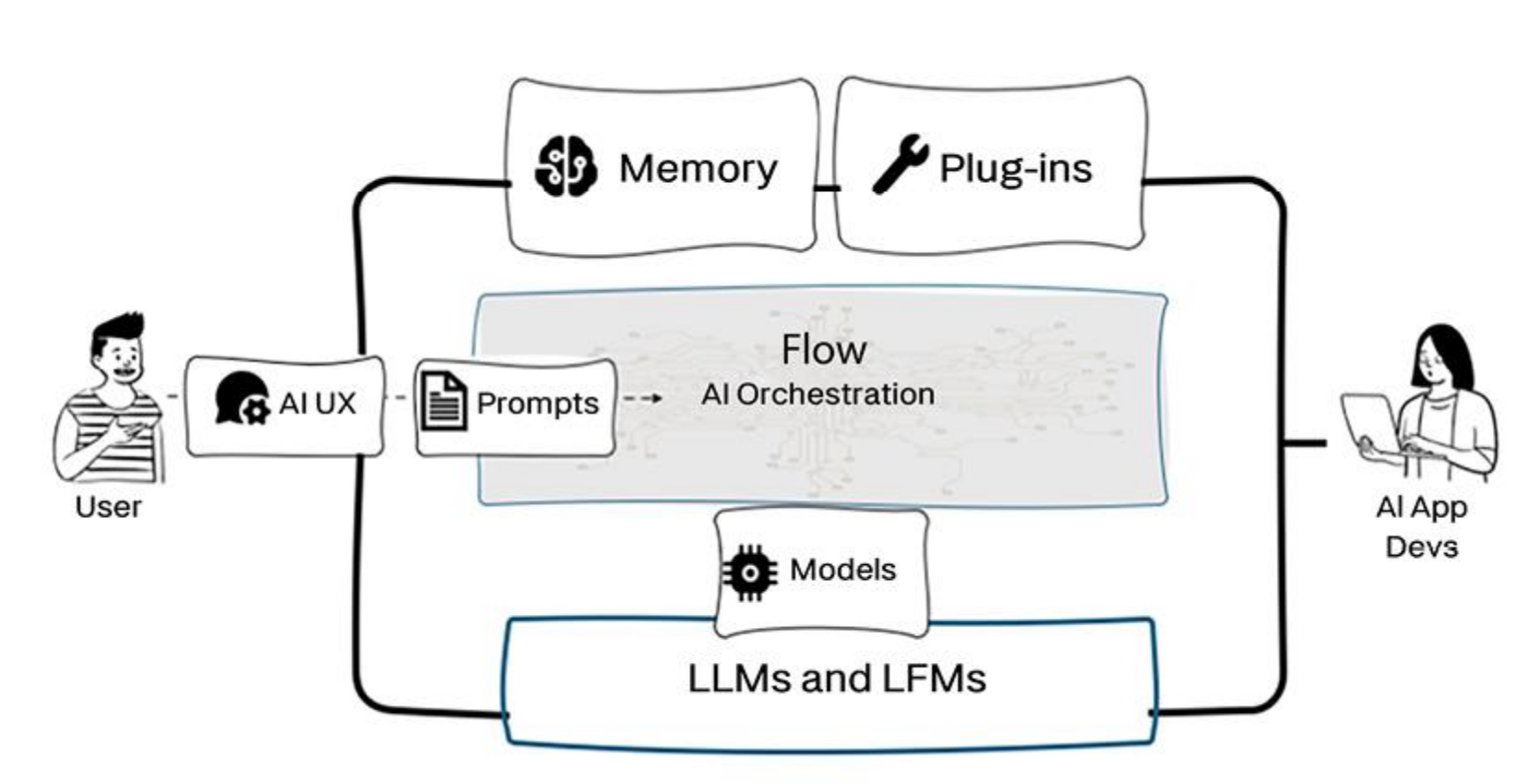


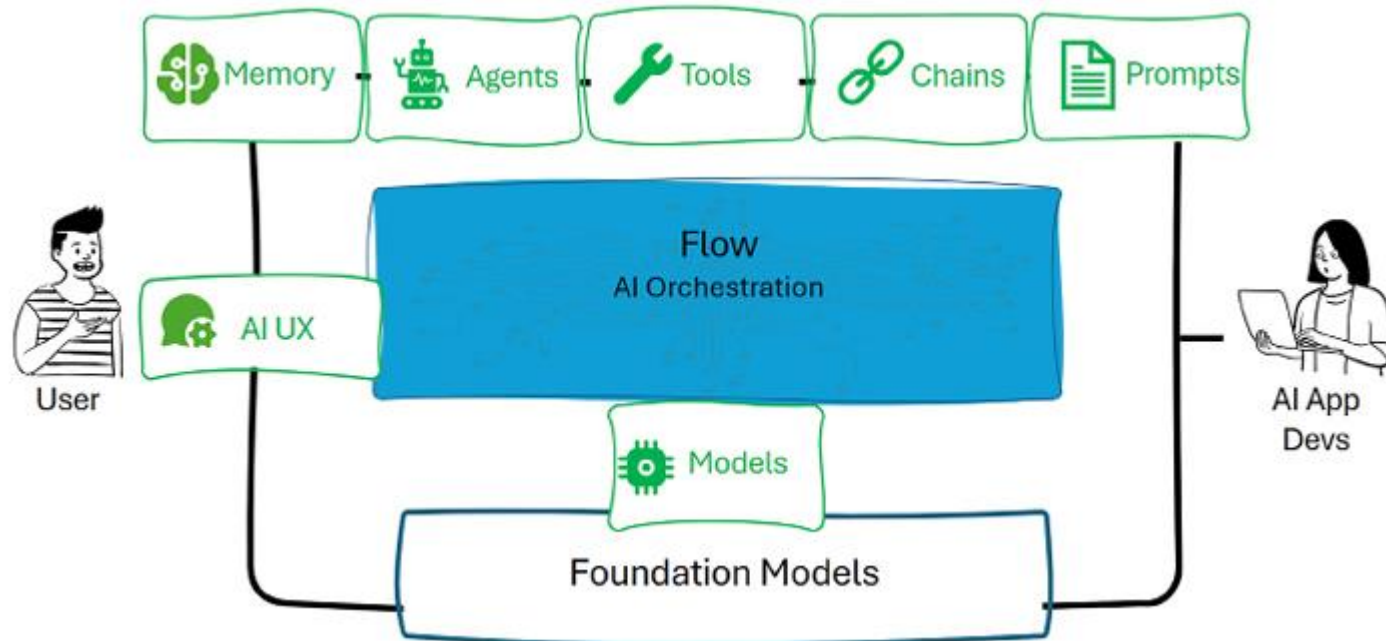
# Neural Retrieval



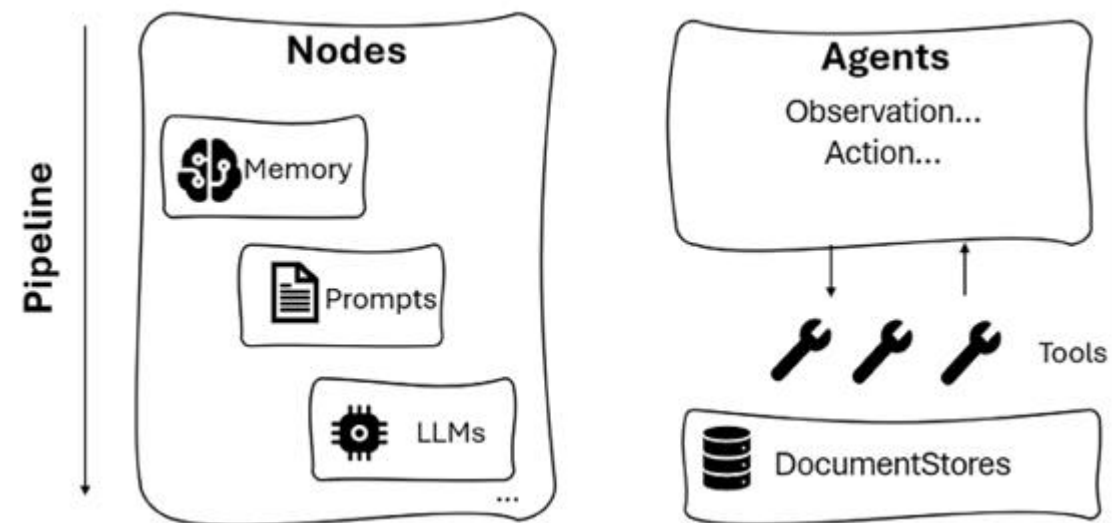
# Hierarchical Summarization



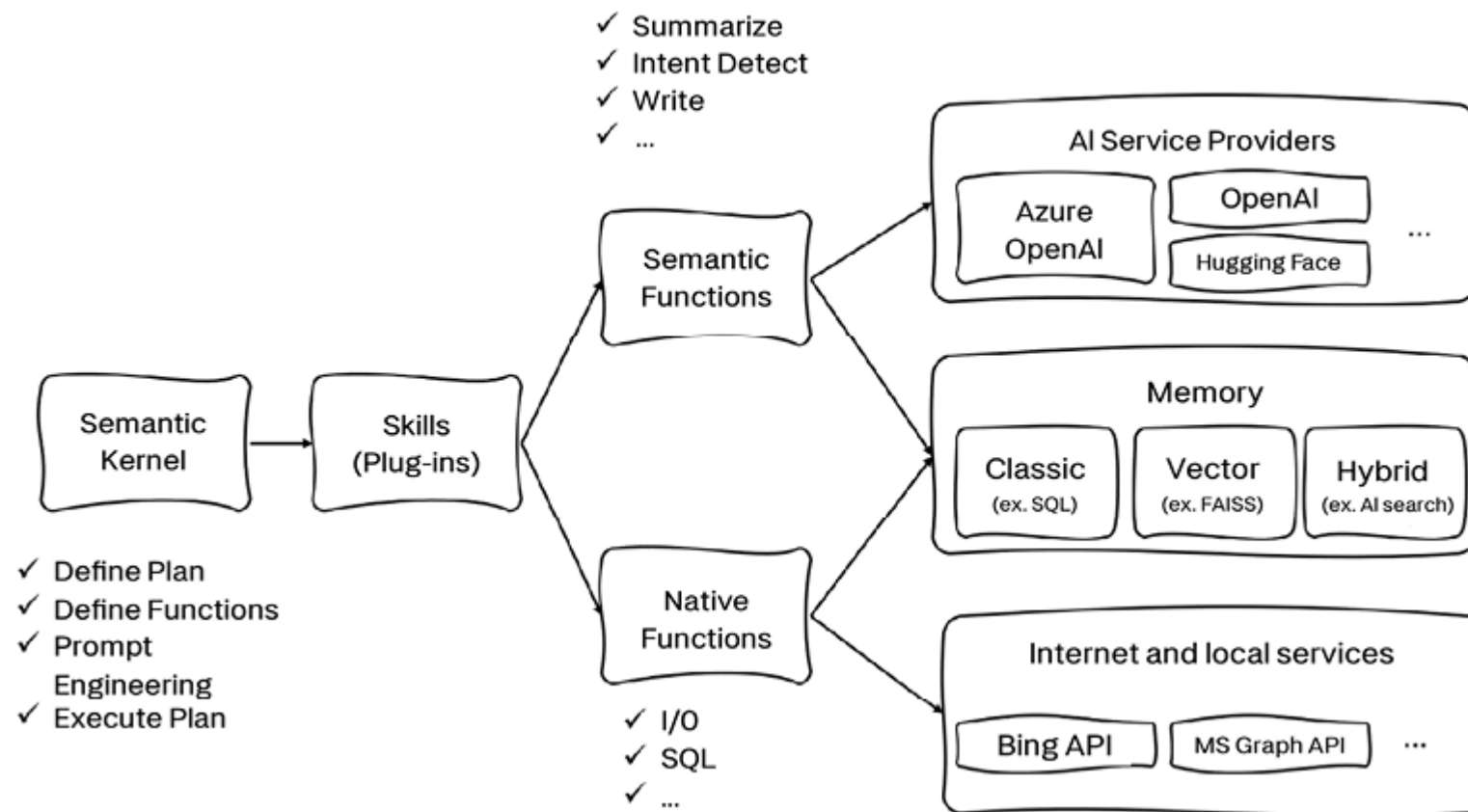




*LangChain's components*



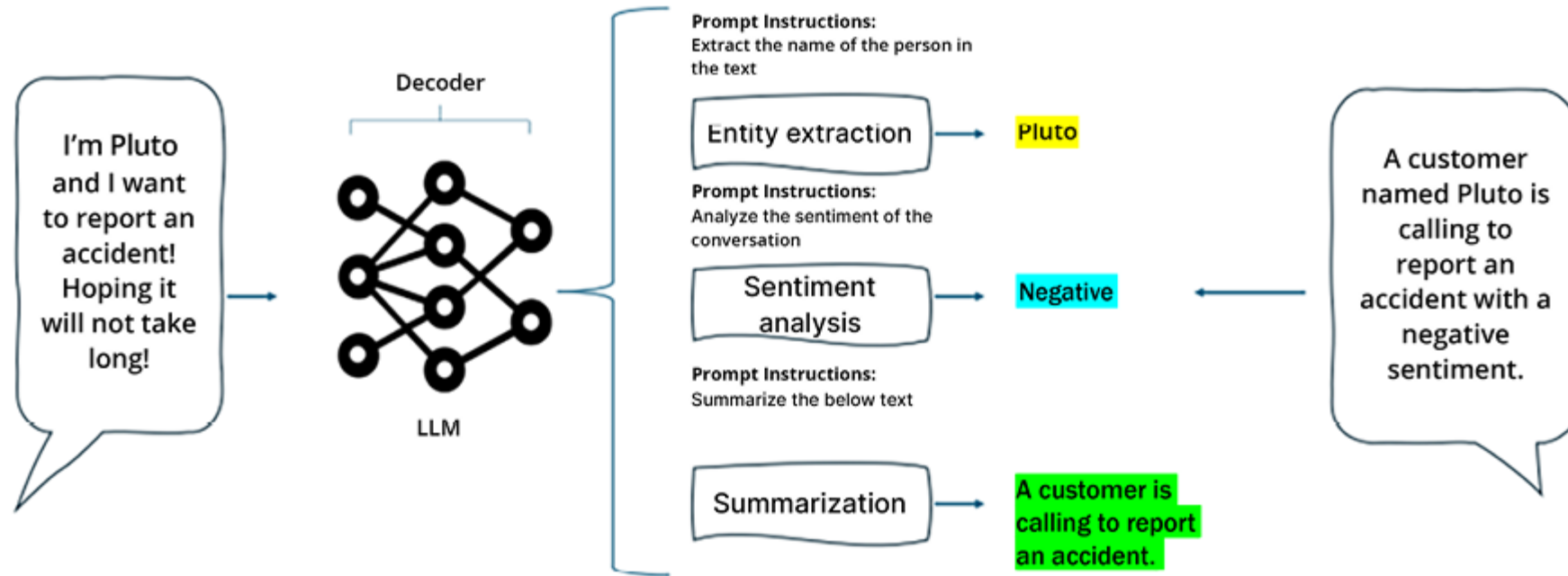
*Haystack's components*



*Semantic Kernel*

Feature	LangChain	Haystack	Semantic Kernel
LLM support	Proprietary and open-source	Proprietary and open source	Proprietary and open source
Supported languages	Python and JS/TS	Python	C#, Java, and Python
Process orchestration	Chains	Pipelines of nodes	Pipelines of functions
Deployment	No REST API	REST API	No REST API
Feature	LangChain	Haystack	Semantic Kernel

	GPT-4	Gemini	Claude 2
Company or institution	OpenAI	Google	Anthropic
First release	March 2023	December 2023	July 2023
Architecture	Transformer-based, decoder only	Transformer-based	Transformer-based
Sizes and variants	Parameters not officially specified Two context-length variants: GPT-4 8K tokens GPT-4 32K tokens	Three sizes, from smallest to largest: Nano, Pro, and Ultra	Not officially specified
How to use	REST API at OpenAI developer platforms Using OpenAI Playground at <a href="https://platform.openai.com/playground">https://platform.openai.com/playground</a>	REST API at Google AI Studio Using Gemini at <a href="https://gemini.google.com/">https://gemini.google.com/</a>	REST API after compiling the form at <a href="https://www.anthropic.com/claude">https://www.anthropic.com/claude</a>



Prompt engineering to specialize LLMs

```
import os
import openai
openai.api_key = os.getenv('OPENAI_API_KEY')
response = openai.ChatCompletion.create(
    model="gpt-3.5-turbo", # engine = "deployment_name".
    messages=[
        {"role": "system", "content": system_message},
        {"role": "user", "content": instructions},
    ]
)
```

```
system_message = """
You are an AI assistant that helps humans by generating tutorials.
You will be provided with a text. If the text contains any kind of recipe,
Otherwise, inform the user that the text does not contain any kind of recipe.
Text:
"""

instructions = """
To prepare the known sauce from Genova, Italy, you can start by chopping pine nuts.
Chop them in a kitchen mortar together with basil and garlic.
Finally, transfer the pesto to a bowl and stir in the grated Parmesan cheese.
"""
```

◀ ▶

```
print(response['choices'][0]['message']['content'])
```

Sure, I can generate a tutorial based on the given text. Here's a

- Toast the pine nuts on a pan until they are slightly browned.
- Coarsely chop the toasted pine nuts, along with basil leaves and olive oil.
- Add half of the olive oil into the kitchen mortar, and season it with salt and pepper.
- Grind the mixture until it becomes a smooth paste.
- Transfer the pesto to a bowl, and stir in the grated Parmesan cheese.
- You can add more olive oil to achieve the desired consistency if needed.
- Your Genovese pesto sauce is now ready. It can be served with pasta, bread, or as a dip.

◀ ▶



```
system_message = ""
```

You are an AI assistant that summarizes articles.

To complete this task, do the following subtasks:

Read the provided article context comprehensively and identify

Generate a paragraph summary of the current article context th

Print each step of the process.

Article:

```
"""
```

```
article = ""
```

Recurrent neural networks, long short-term memory, and gated r

in particular, [...]

```
"""
```

◀ ▶

```
system_message = ""
```

You are an AI assistant specialized in solving riddles.

Given a riddle, you have to generate three answers to the riddle.

For each answer, be specific about the reasoning you made.

Then, among the three answers, select the one that is most plausi

Riddle:

```
"""
```

```
riddle = ""
```

What has a face and two hands, but no arms or legs?

```
"""
```

◀ ▶

```
system_message = ""
```

You are a sentiment analyzer. You classify conversations into thr

Return only the sentiment, in lowercase and without punctuation.

Conversation:

```
"""
```

```
conversation = ""
```

Customer: Hi, I need some help with my order.

AI agent: Hello, welcome to our online store. I'm an AI agent and

Customer: I ordered a pair of shoes yesterday, but I haven't rece

```
[...]
```

```
"""
```

◀ ▶

```
system_message = ""
```

You are a Python expert who produces Python code as per the user'

```
==>START EXAMPLE
```

```
---User Query---
```

Give me a function to print a string of text.

```
---User Output---
```

Below you can find the described function:

```
```def my_print(text):
```

```
    return print(text)
```

```
```
```

```
<==END EXAMPLE
```

```
"""
```

query = "generate a Python function to calculate the nth Fibonacc

◀ ▶

```

system_message = """
You are a binary classifier for sentiment analysis.
Given a text, based on its sentiment, you classify it into one of
You can use the following texts as examples:
Text: "I love this product! It's fantastic and works perfectly."
Positive
Text: "I'm really disappointed with the quality of the food."
Negative
Text: "This is the best day of my life!"
Positive
Text: "I can't stand the noise in this restaurant."
Negative
ONLY return the sentiment as output (without punctuation).
Text:
"""

```

## Few-shot approach

```

system_message = """
To solve a generic first-degree equation, follow these steps:
1. **Identify the Equation:** Start by identifying the equation y
2. **Isolate the Variable:** Your goal is to isolate the variable

    a. **Add or Subtract Constants:** Add or subtract 'b' from bot

    b. **Divide by the Coefficient:** Divide both sides by 'a' to
3. **Simplify:** Simplify both sides of the equation as much as p
4. **Solve for 'x':** Once 'x' is isolated on one side, you have
5. **Check Your Solution:** Plug the found value of 'x' back into
6. **Express the Solution:** Write down the solution in a clear a
7. **Consider Special Cases:** Be aware of special cases where th
Equation:
"""

```

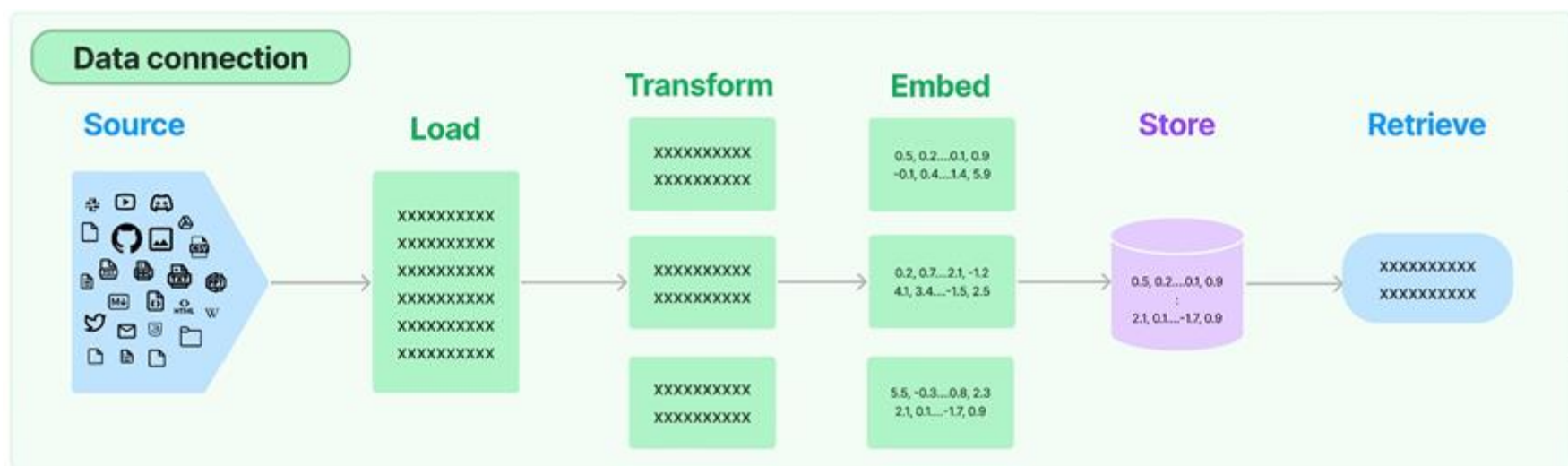
## Chain of thought

```

from langchain import PromptTemplate
template = """Sentence: {sentence}
Translation in {language}:"""
prompt = PromptTemplate(template=template, input_variables=["sent
print(prompt.format(sentence = "the cat is on the table", languag

```

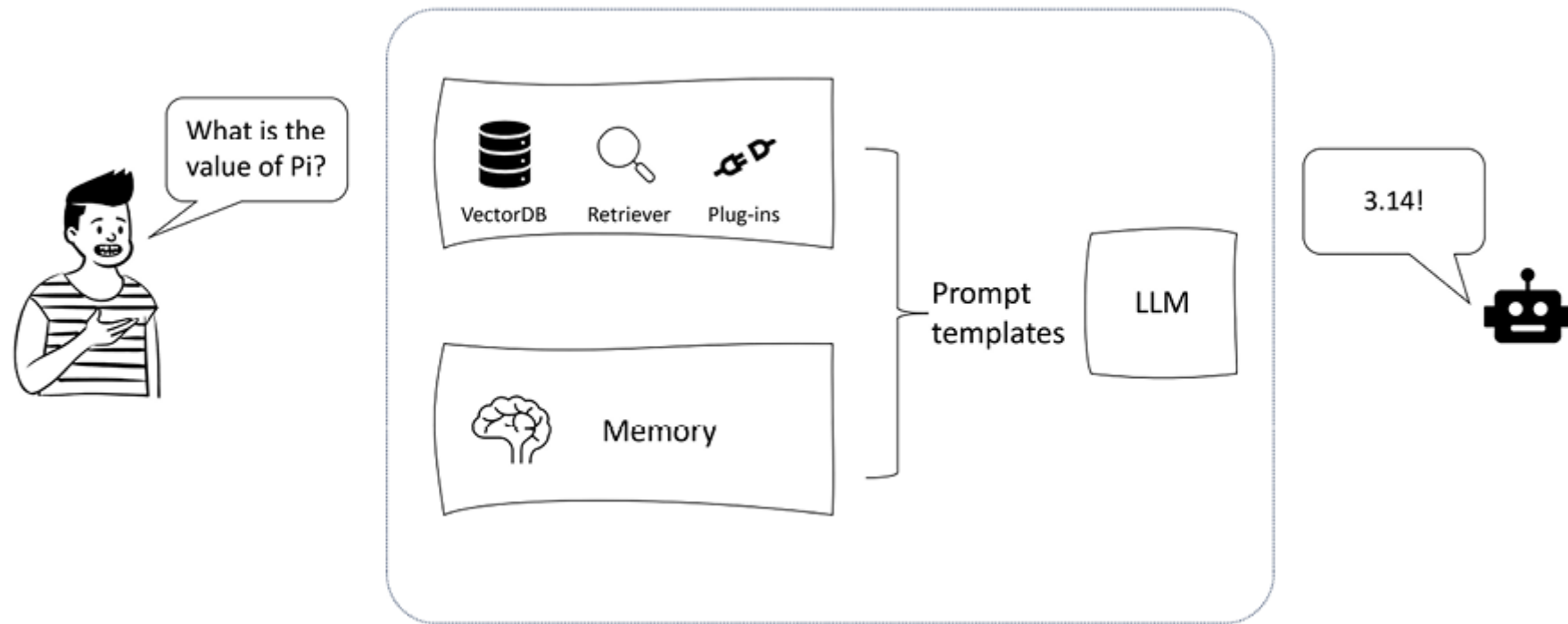
## LangChain



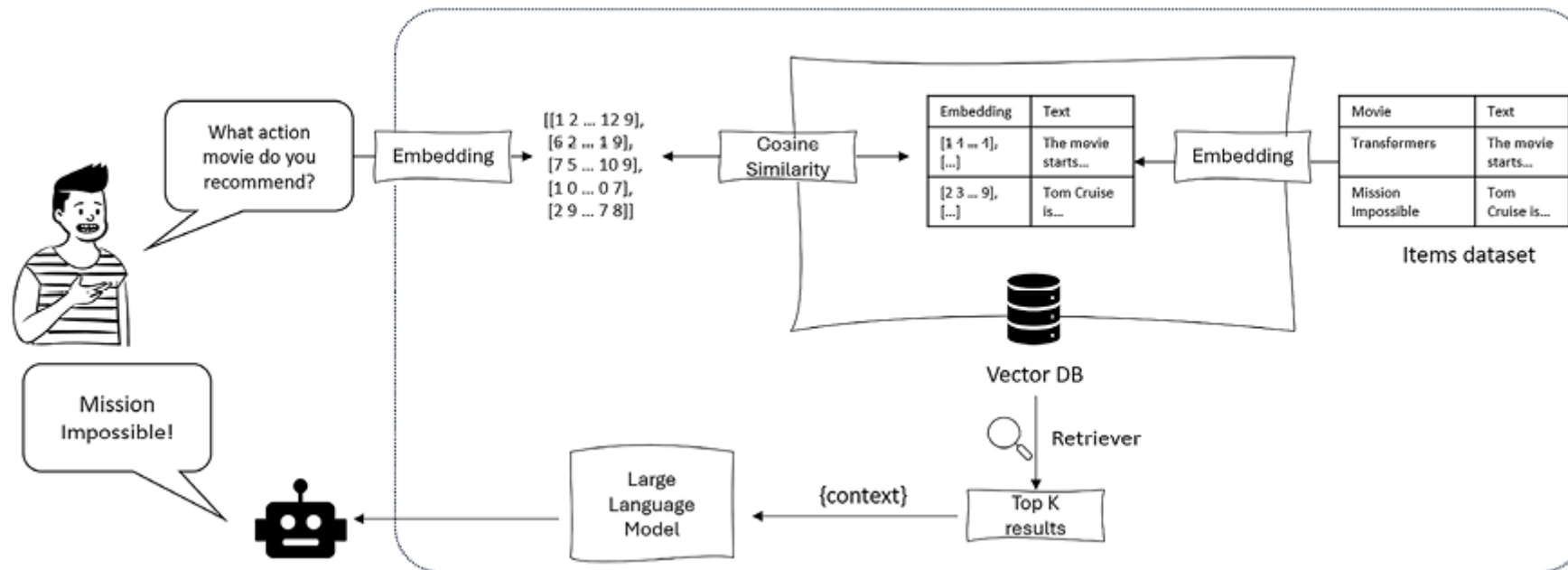
# Agents

```
from langchain import SerpAPIWrapper
from langchain.agents import AgentType, initialize_agent
from langchain.llms import OpenAI
from langchain.tools import BaseTool, StructuredTool, Tool, tool
import os
from dotenv import load_dotenv
load_dotenv()
os.environ["SERPAPI_API_KEY"]
search = SerpAPIWrapper()
tools = [Tool.from_function(
    func=search.run,
    name="Search",
    description="useful for when you need to answer questions
)]
agent = initialize_agent(tools, llm = OpenAI(), agent=AgentType.Z
agent.run("When was Avatar 2 released?")
```

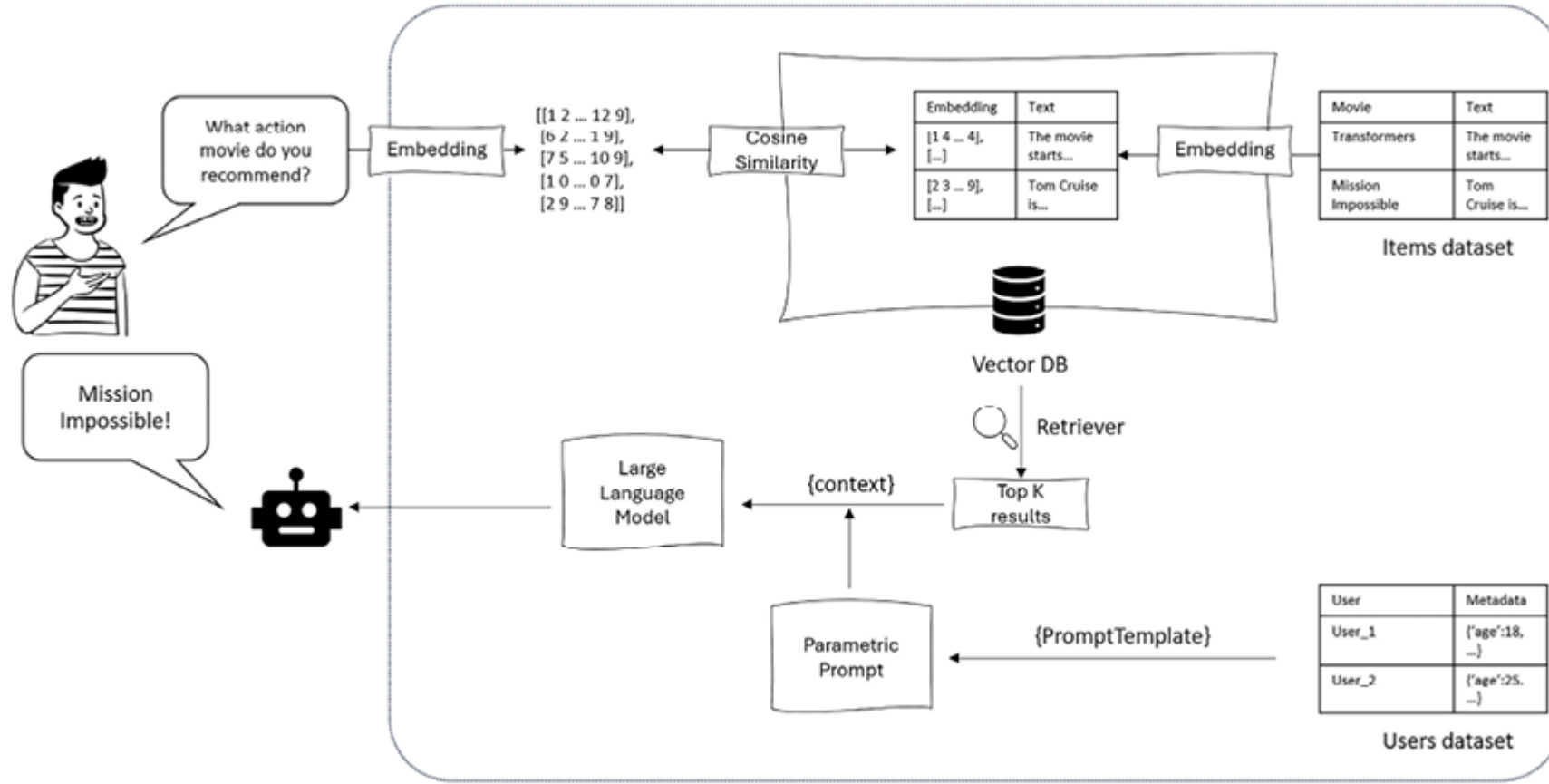
```
> Entering new AgentExecutor chain...
  I need to find out when Avatar 2 was released.
Action: Search
Action Input: "Avatar 2 release date"
Observation: December 16, 2022
Thought: I now know the final answer.
Final Answer: Avatar 2 was released on December 16, 2022.
> Finished chain.
'Avatar 2 was released on December 16, 2022.'
```



*architecture of a conversational bot*

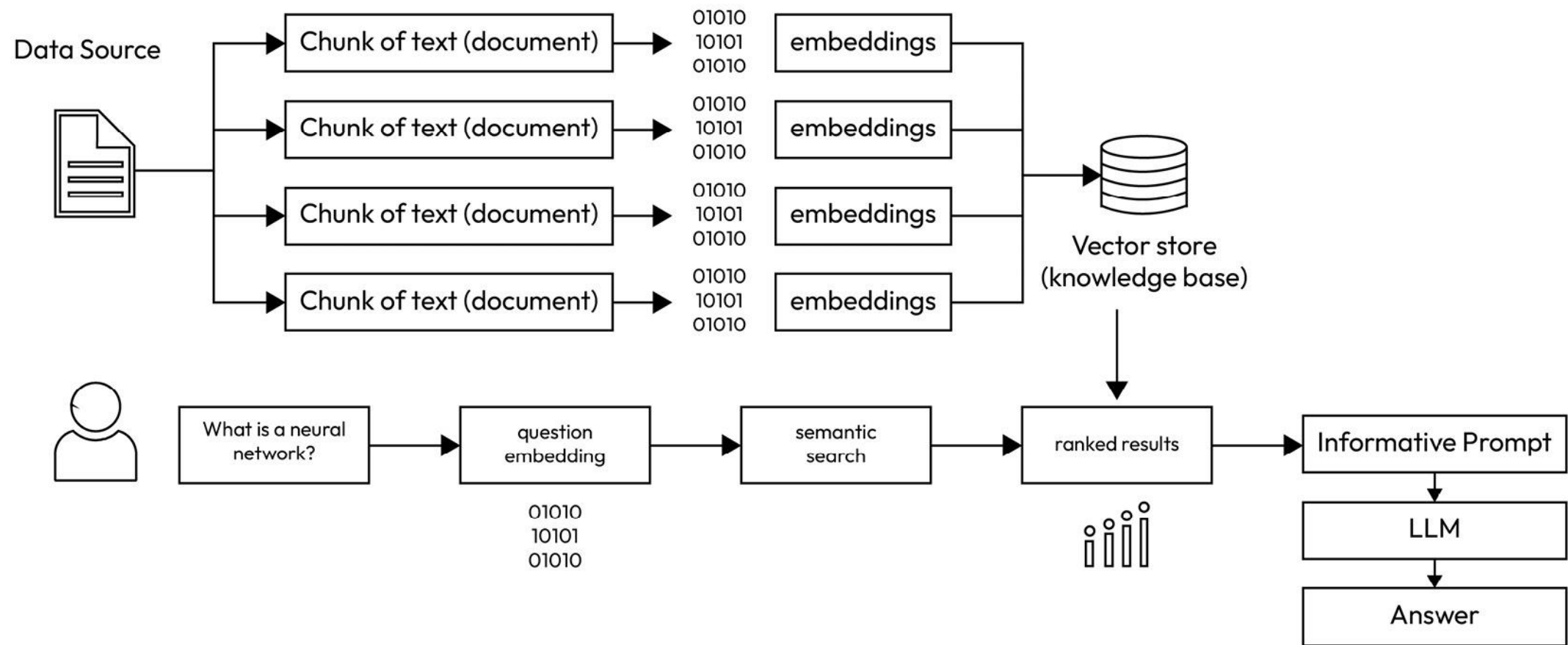


Building a QA recommendation chatbot in a cold-start scenario



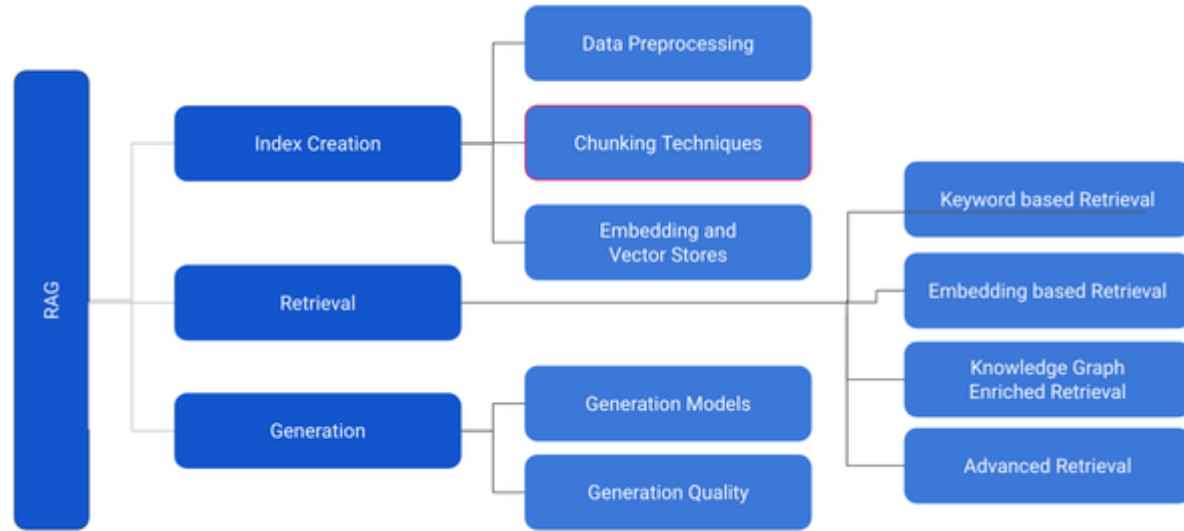
High-level architecture of a content-based recommendation system

LangChain pipeline

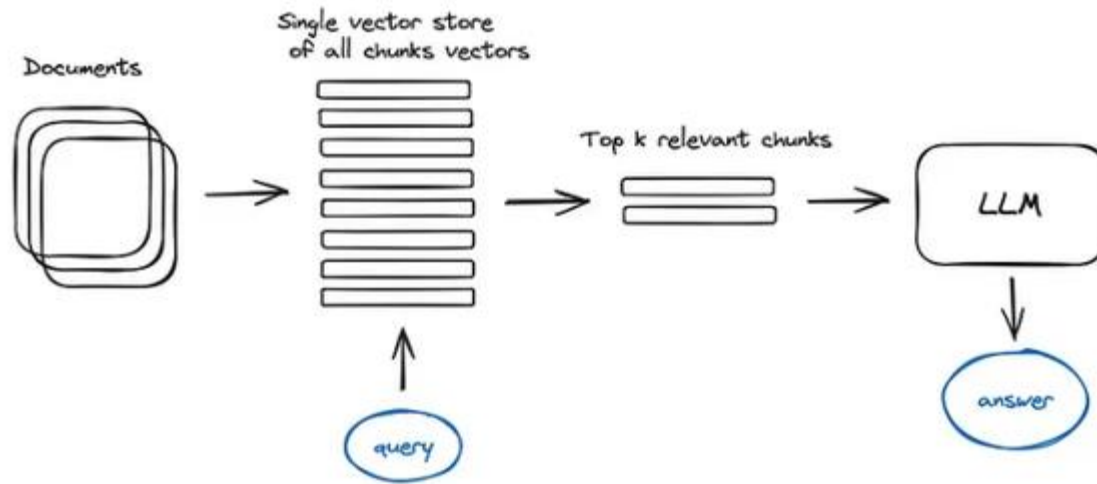




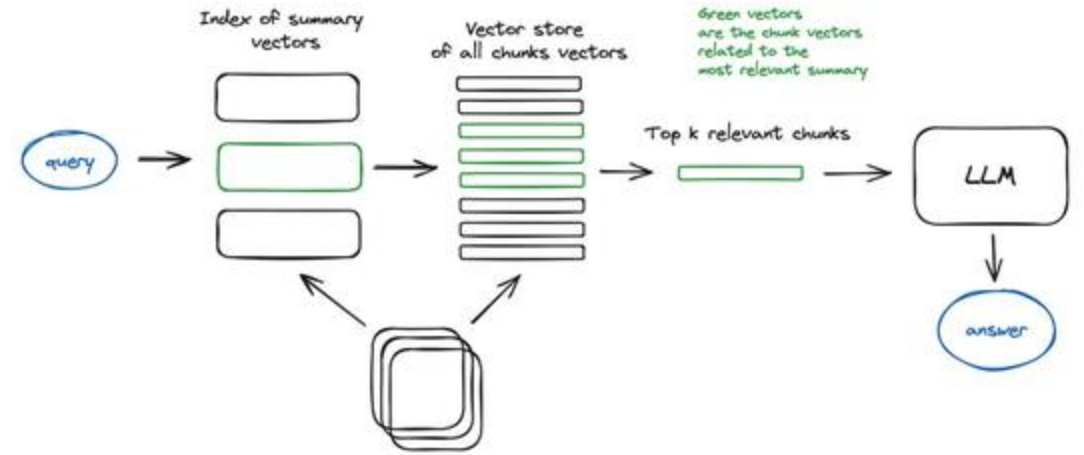
# RAG Components



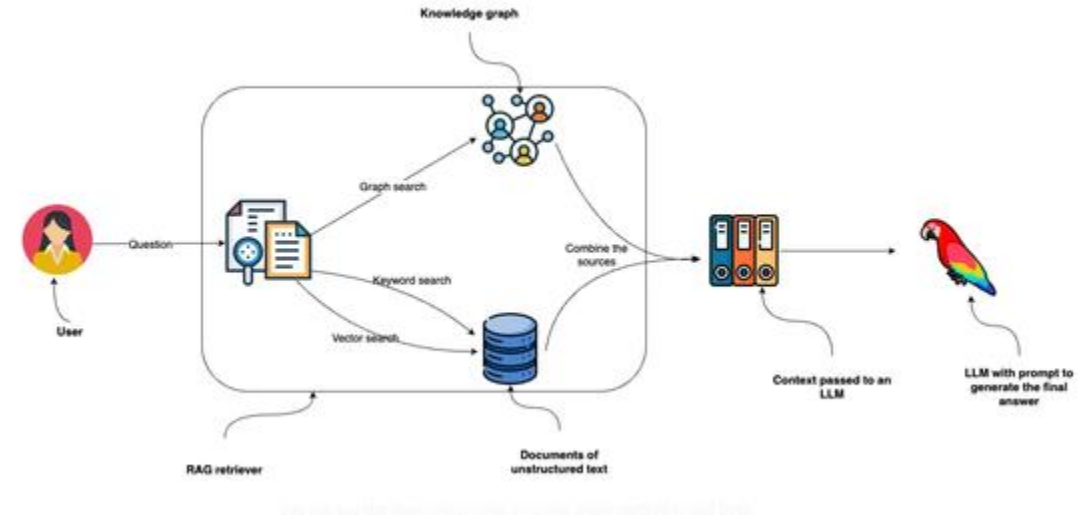
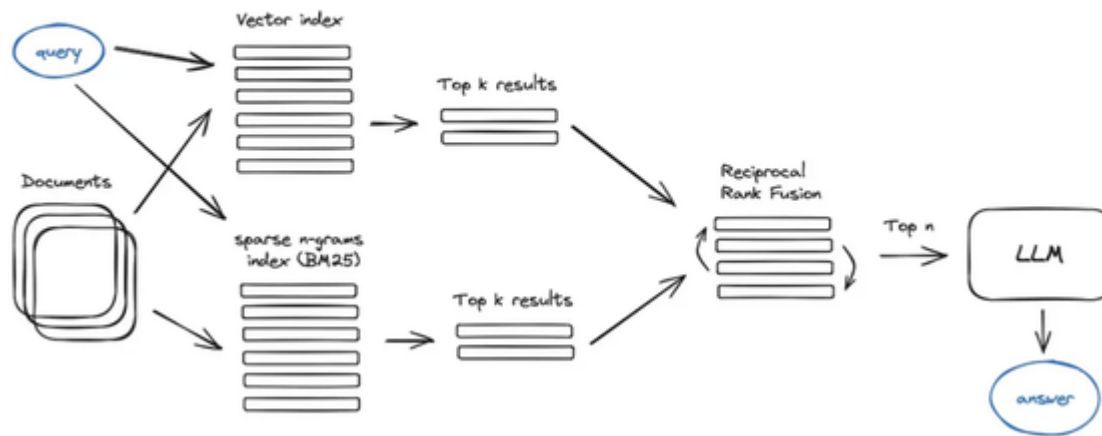
## Basic index retrieval



## Hierarchical index retrieval



## Fusion retrieval / hybrid search



# Using LLMs with Structured Data

The primary key in the Authors table becomes the foreign key in the Books table

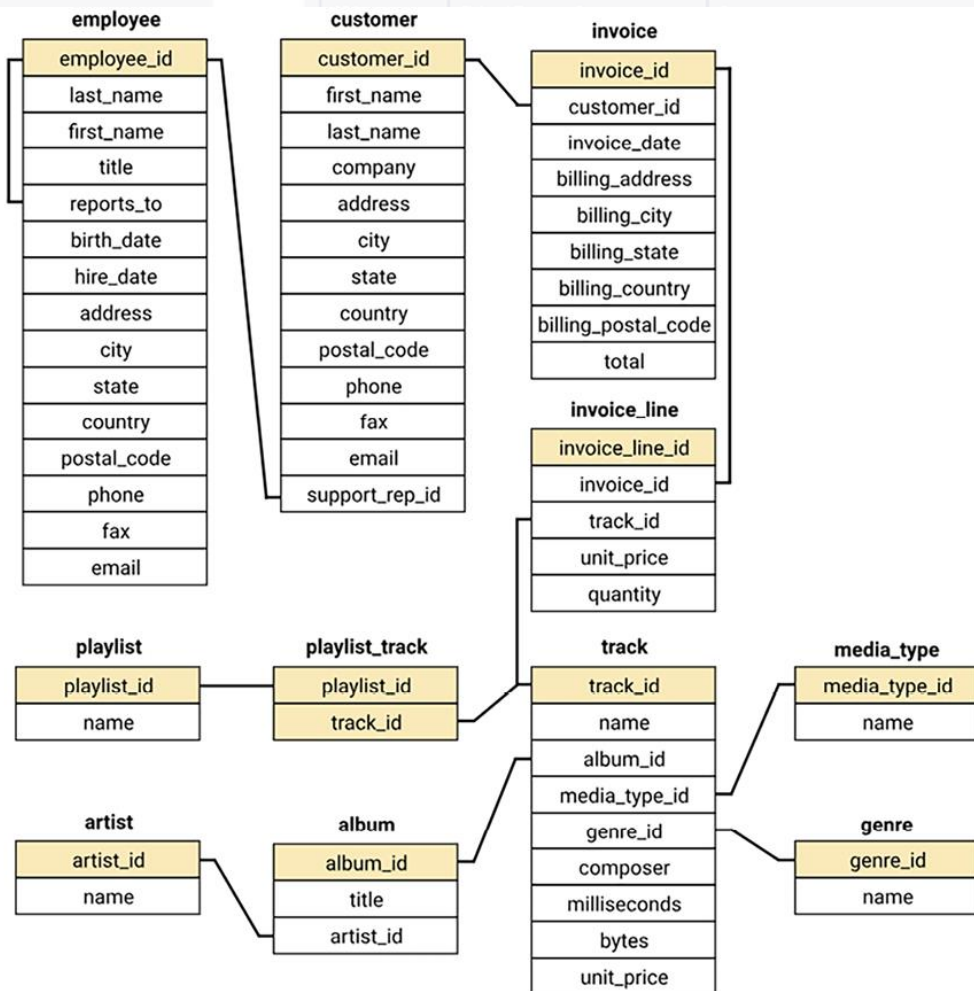
Primary key

Foreign key

| AuthorID | AuthorName     | BirthYear |
|----------|----------------|-----------|
| 1        | Jane Smith     | 1980      |
| 2        | John Johnson   | 1975      |
| 3        | Emily Williams | 1992      |

Authors table

| BookID | Title             | AuthorID (Foreign Key) |
|--------|-------------------|------------------------|
| 101    | "Mystery Mansion" | 1                      |
| 102    | "Adventure Quest" | 2                      |
| 103    | "Hidden Secrets"  | 1                      |



- **SQL databases:** These are **relational database management systems (RDBMS)** that use SQL for data manipulation and querying. Examples include MySQL, PostgreSQL, and Microsoft SQL Server.
- **Oracle Database:** A widely-used RDBMS that offers advanced features and scalability for large-scale applications.
- **SQLite:** A self-contained, serverless, and zero-configuration SQL database engine commonly used in embedded systems and mobile applications.
- **IBM Db2:** A family of data management products, including relational database servers, developed by IBM.
- **Amazon Web Services (AWS) RDS:** A managed relational database service offered by Amazon, providing options for various databases like MySQL, PostgreSQL, SQL Server, and more.
- **Google Cloud SQL:** A managed database service by Google Cloud Platform, supporting MySQL, PostgreSQL, and SQL Server.
- **Redis:** An open-source, in-memory data structure store that can be used as a database, cache, and message broker.

```
from langchain.chat_models import ChatOpenAI
llm = OpenAI()
db = SQLiteDatabase.from_uri('sqlite:///chinook.db')
toolkit = SQLiteDatabaseToolkit(db=db, llm=llm)
agent_executor = create_sql_agent(
    llm=llm,
    toolkit=toolkit,
    verbose=True,
    agent_type=AgentType.ZERO_SHOT_REACT_DESCRIPTION,
)
```

```
agent_executor.run("Describe the playlisttrack table")
```

```
> Entering new AgentExecutor chain...
Action: sql_db_list_tables
Action Input:
Observation: album, artist, customer, employee, genre, invoice, i
Thought: The table I need is playlist_track
Action: sql_db_schema
Action Input: playlist_track
Observation:
CREATE TABLE playlist_track (
[...]
> Finished chain.
'The playlist_track table contains the playlist_id and track_id c
◀ ▶
```

You are an agent designed to interact with a SQL database.

Given an input question, create a syntactically correct sqlite query. Unless the user specifies a specific number of examples they wish, you can order the results by a relevant column to return the most recent results. Never query for all the columns from a specific table, only ask for the relevant columns. You have access to tools for interacting with the database.

Only use the below tools. Only use the information returned by the tools. You MUST double check your query before executing it. If you get an error, you MUST make sure you have the right query. DO NOT make any DML statements (INSERT, UPDATE, DELETE, DROP etc.) to the database. If the question does not seem related to the database, just return the answer.

sql\_db\_query: Input to this tool is a detailed and correct SQL query. It will return the results of the query.

sql\_db\_schema: Input to this tool is a comma-separated list of table names. It will return the schema for the tables.

Be sure that the tables actually exist by calling sql\_db\_list\_tables.

sql\_db\_list\_tables: Input is an empty string, output is a comma-separated list of table names.

sql\_db\_query\_checker: Use this tool to double check if your query is syntactically correct.

Use the following format:

Question: the input question you must answer

Thought: you should always think about what to do

Action: the action to take, should be one of [sql\_db\_query, sql\_db\_schema, sql\_db\_list\_tables, sql\_db\_query\_checker]

Action Input: the input to the action

...

Question: {input}

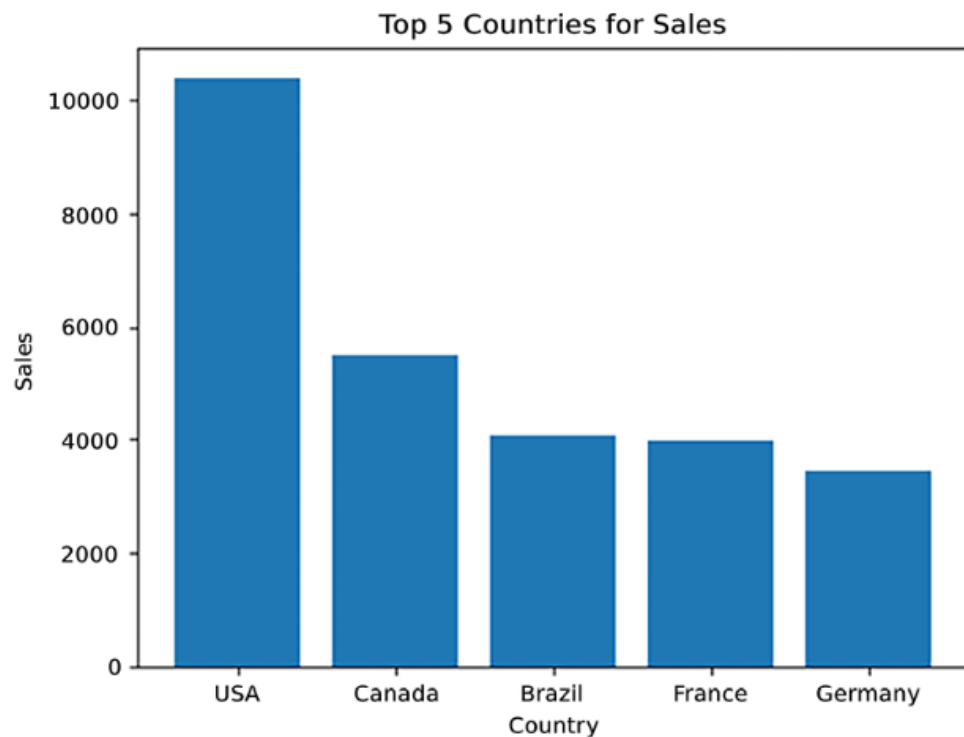
Thought: I should look at the tables in the database to see what tables are available.

{agent\_scratchpad}

```

from langchain.chat_models import ChatOpenAI
from langchain.agents import initialize_agent, Tool
from langchain.agents import AgentType
model = ChatOpenAI()
agent = initialize_agent(
    tools, model, agent= AgentType.STRUCTURED_CHAT_ZERO_SHOT_REAC
)
agent.run("generate a matplotlib bar chart of the top 5 countries

```



```

> Entering new AgentExecutor chain...
Action:
...
{
  "action": "sql_db_query",
  "action_input": "SELECT billing_country as Country, SUM(total)
}
...
[...]
Observation: [('USA', 10405.889999999999), ('Canada', 5489.549999
[...]
We have successfully retrieved the top 5 countries for sales. We
Action:
...
{
  "action": "Python_REPL",
  "action_input": "import matplotlib.pyplot as plt\nsales_data =
}
...
[...]
> Finished chain.
'Here is the bar chart of the top 5 countries for sales from the

```



What are the top 5 best-selling albums and their artists?



✓ sql\_db\_list\_tables:



✓ sql\_db\_schema: album, artist, invoice, invoice\_line



✓ sql\_db\_query: SELECT a.title AS Album, ar.name AS Artist, SUM(il.quantity)...



The top 5 best-selling albums and their artists are: Jagged Little Pill by Alanis Morissette, The Best Of Billy Cobham by Billy Cobham, Load by Metallica, Plays Metallica By Four Cellos by Apocalyptica, and Supernatural by Santana.

Explanation: I joined the invoice\_line, invoice, album, and artist tables on the invoice\_id, album\_id, and artist\_id columns, which are the common keys between them. This allowed me to access the quantity, title, and name columns from all four tables. Then I grouped the records by the album title and artist name and calculate the sum of the quantity column for each album/artist combination, ordered them in descending order and limited the SELECT to the top 5.

```
SELECT a.title AS Album, ar.name AS Artist, SUM(il.quantity) AS Sales
FROM invoice_line il
JOIN invoice i ON il.invoice_id = i.invoice_id
JOIN album a ON a.album_id = il.track_id
JOIN artist ar ON ar.artist_id = a
```

Ask me anything!



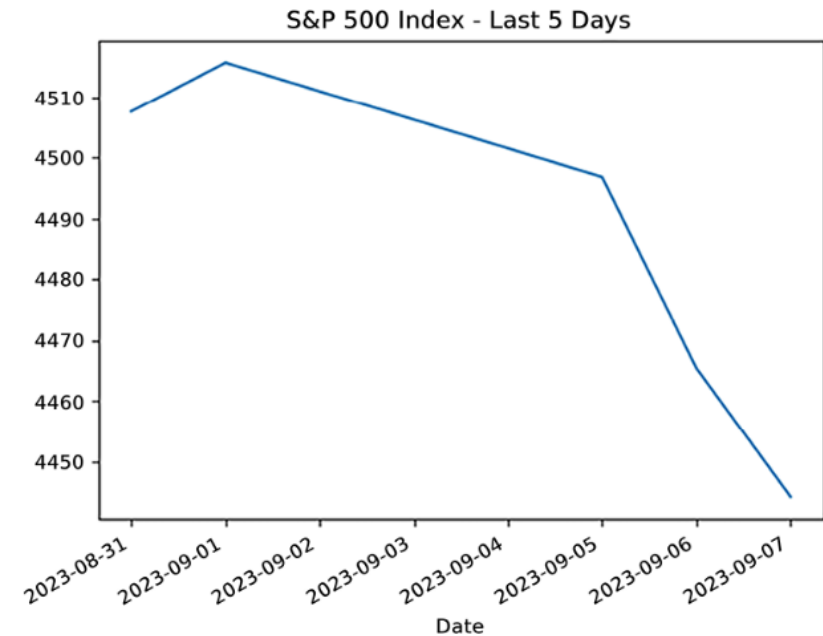
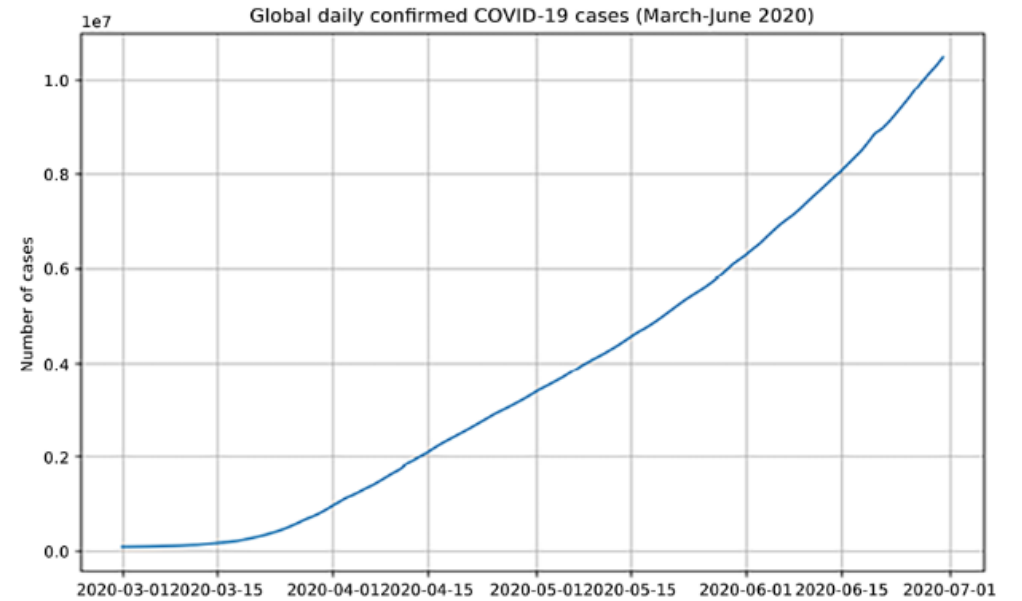
*Streamlit  
front-end DBCopilot*



# Code Interpreter API

```
from codeinterpreterapi import CodeInterpreterSession
import os
from dotenv import load_dotenv
load_dotenv()
api_key = os.environ['OPENAI_API_KEY']
# create a session
async with CodeInterpreterSession() as session:
    # generate a response based on user input
    response = await session.generate_response(
        "Generate a plot of the evolution of Covid-19 from March
    )
    # output the response
    print("AI: ", response.content)
    for file in response.files:
        file.show_image()
```

```
async with CodeInterpreterSession() as session:
    # generate a response based on user input
    response = await session.generate_response(
        "Generate a plot of the price of S&P500 index in the last
    )
    # output the response
    print("AI: ", response.content)
    for file in response.files:
        file.show_image()
```

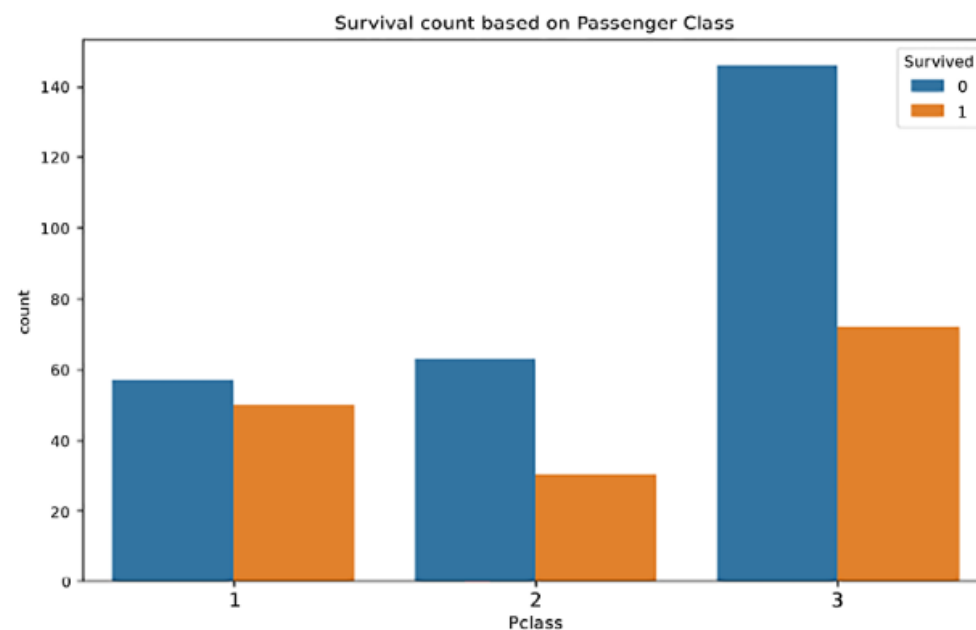
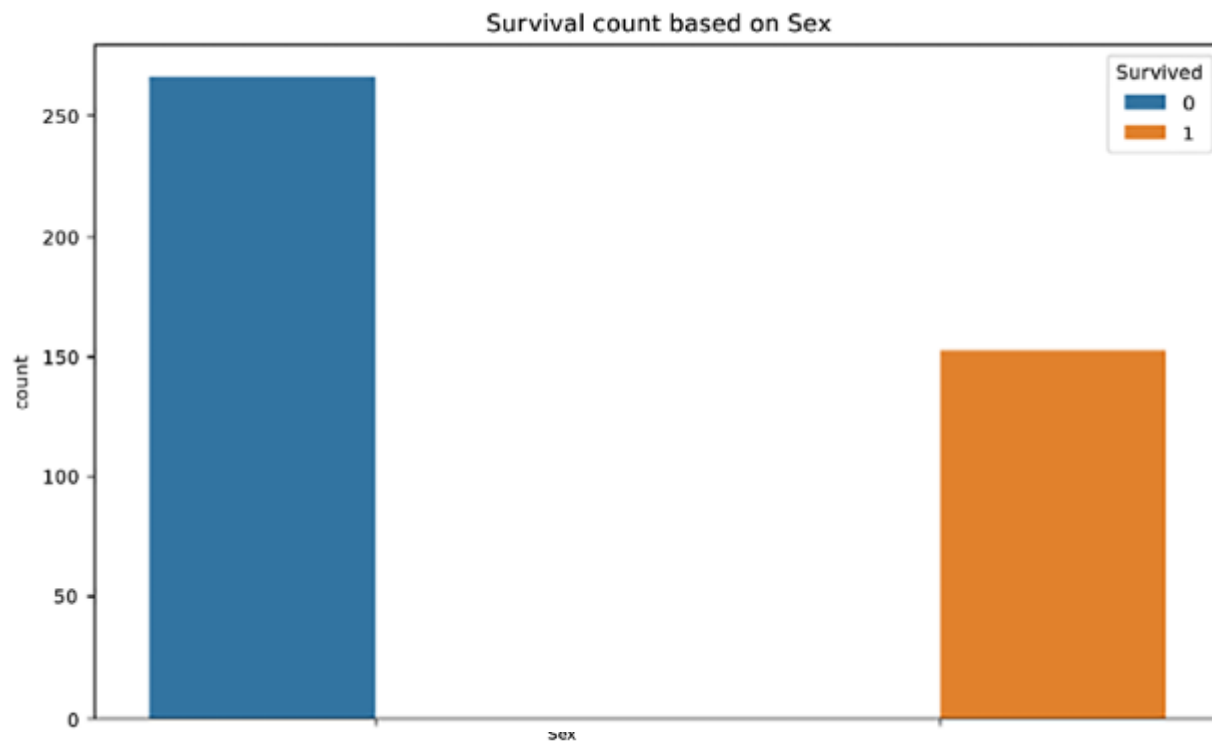




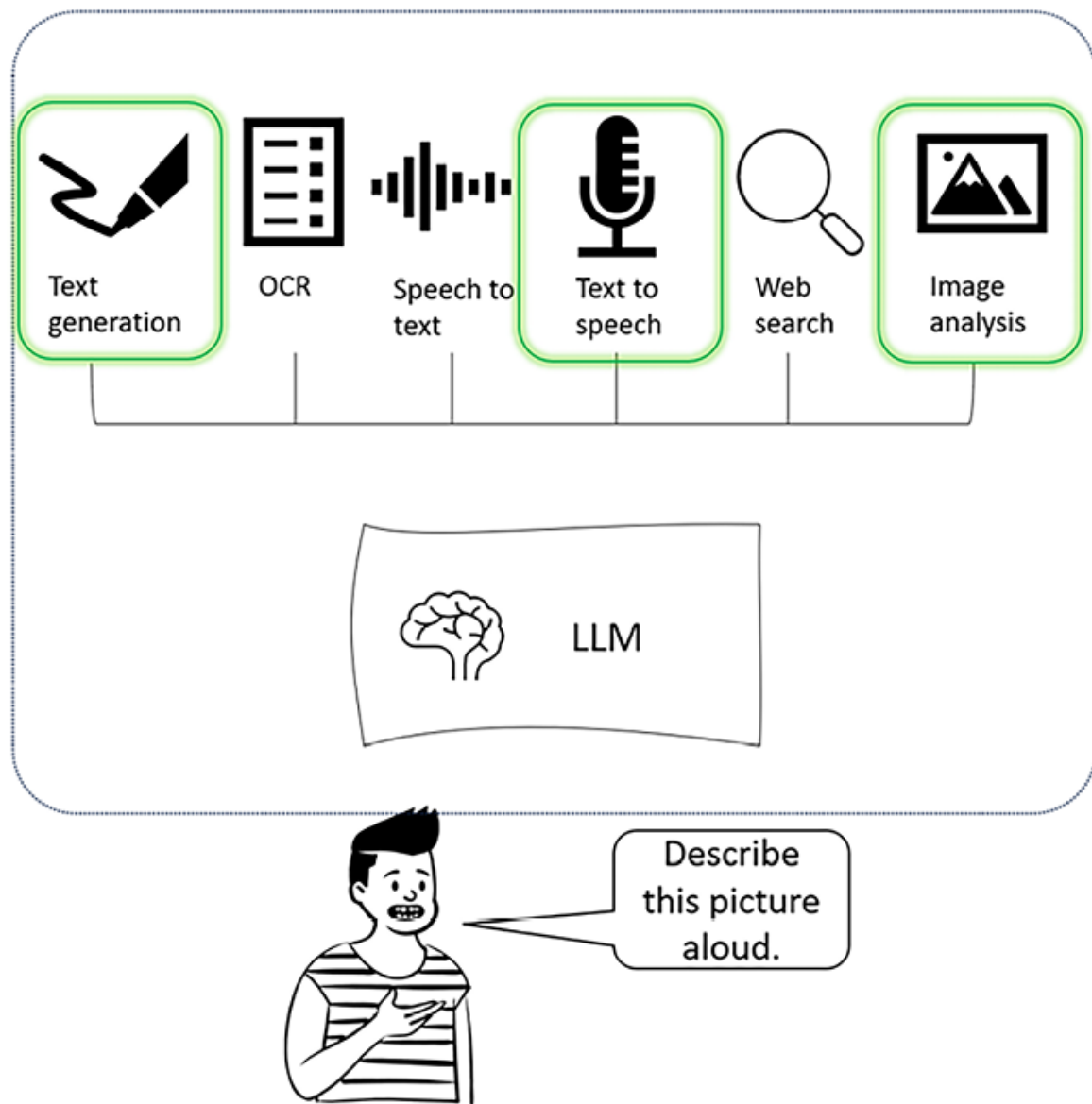
```

from codeinterpreterapi import CodeInterpreterSession, File
#os.environ["HUGGINGFACEHUB_API_TOKEN"]
os.environ['OPENAI_API_KEY'] = "sk-YIN03tURjJRYmhcmv0yIT3BlbkFJvO
os.environ['VERBOSE'] = "True"
async with CodeInterpreterSession() as session:
    # define the user request
    user_request = "Analyze this dataset and plot something i
    files = [
        File.from_path("drive/MyDrive/titanic.csv"),
    ]
    # generate the response
    response = await session.generate_response(
        user_request, files=files
    )
    # output to the user
    print("AI: ", response.content)
    for file in response.files:
        file.show_image()

```



## multimodal application with single-modal tools



- **AzureCogsImageAnalysisTool**: Used to analyze and extract metadata from images.
- **AzureCogsSpeech2TextTool**: Used to convert speech to text.
- **AzureCogsText2SpeechTool**: Used to synthesize text to speech with neural voices.
- **AzureCogsFormRecognizerTool**: Used to perform **optical character recognition (OCR)**.

```
description = agent.run("what shows the following image?:"  
    "https://www.stylo24.it/wp-content/uploads/2020/03/fionda.jpg"  
print(description)
```

```
> Entering new AgentExecutor chain...  
Action:  
...  
{  
    "action": "azure_cognitive_services_image_analysis",  
    "action_input": "https://www.stylo24.it/wp-content/uploads/2020  
}  
...  
Observation: Caption: a person holding a slingshot  
Tags: person, tool, nail, hand, holding, needle  
Thought: I know what the image is.  
Action:  
...  
{  
    "action": "Final Answer",  
    "action_input": "The image is of a person holding a slingshot."  
}  
...  
> Finished chain.  
The image is of a person holding a slingshot.
```



User What is funny about this image? Describe it panel by panel.



Source: [hmmm \(Reddit\)](#)

GPT-4 The image shows a package for a "Lightning Cable" adapter with three panels.

Panel 1: A smartphone with a VGA connector (a large, blue, 15-pin connector typically used for computer monitors) plugged into its charging port.

Panel 2: The package for the "Lightning Cable" adapter with a picture of a VGA connector on it.

Panel 3: A close-up of the VGA connector with a small Lightning connector (used for charging iPhones and other Apple devices) at the end.

The humor in this image comes from the absurdity of plugging a large, outdated VGA connector into a small, modern smartphone charging port.

## Leveraging multiple tools

```
1("Tell me a story related to the following picture and read the s
```

```
Thought:The human requested a story related to the picture. I wil
Action:
...
{
  "action": "azure_cognitive_services_image_analysis",
  "action_input": {
    "query": "https://i.redd.it/diawvlriobq11.jpg"
  }
}
...

Observation: Caption: a dog standing on a snowy hill
Objects: dog
Tags: outdoor, animal, mammal, snow, freezing, dog breed, dog, mo
Thought:Based on the Azure Cognitive Services Image Analysis tool
Action:
...
{
  "action": "azure_cognitive_services_text2speech",
  "action_input": {
    "query": "Once upon a time, in a snowy winter wonderland, the
  }
}
...

Observation: C:\Users\vaalt\AppData\Local\Temp\tmp_cecwqht.wav
Thought:The story involving a dog in a snowy winter wonderland se
Action:
...
{
  "action": "Final Answer",
  "action_input": "I hope you enjoyed the story of Snowy the Savi
```

PURCHASE ORDER TEMPLATE

YOUR LOGO

PURCHASE ORDER

Company Name  
123 Main Street  
Hamilton, OH 44416  
(321) 456-7890  
Email Address  
Point of Contact

DATE

01/18/2022

PURCHASE ORDER NO.

A246

CUSTOMER NO.

114H

BILL TO  
ATTN: Name / Dept  
Company Name  
123 Main Street  
Hamilton, OH 44416  
(321) 456-7890  
Email Address

SHIP TO  
ATTN: Name / Dept  
Company Name  
123 Main Street  
Hamilton, OH 44416  
(321) 456-7890

| SHIPPING METHOD | SHIPPING TERMS | SHIP VIA | PAYMENT | DELIVERY DATE |
|-----------------|----------------|----------|---------|---------------|
|                 |                |          |         |               |

| ITEM NO. | DESCRIPTION      | QTY | UNIT PRICE | TOTAL    |
|----------|------------------|-----|------------|----------|
| A111     | Women's Tall - M | 10  | \$10.00    | \$100.00 |
| 8222     | Men's Tall - M   | 5   | \$20.00    | \$100.00 |
| C333     | Children's - S   | 10  | \$5.00     | \$50.00  |
| D444     | Men's - XL       | 5   | \$10.00    | \$50.00  |
|          |                  |     |            | \$0.00   |
|          |                  |     |            | \$0.00   |
|          |                  |     |            | \$0.00   |
|          |                  |     |            | \$0.00   |
|          |                  |     |            | \$0.00   |
|          |                  |     |            | \$0.00   |
|          |                  |     |            | \$0.00   |

Remarks/Instructions:

|                    |                        |        |
|--------------------|------------------------|--------|
|                    | SUBTOTAL               | 300.00 |
| enter total amount | DISCOUNT               | 50.00  |
|                    | SUBTOTAL LESS DISCOUNT | 250.00 |
| enter percentage   | TAX RATE               | 7.214% |
|                    | TOTAL TAX              | 18.04  |
|                    | SHIPPING/HANDLING      | 50.00  |
|                    | OTHER                  | 50.00  |
|                    | TOTAL \$               | 368.04 |

Please make check payable to Your Company Name.

THANK YOU

AUTHORIZED SIGNATURE

DATE

```
agent.run("what are all men's skus?"  
  "https://www.whiteelysee.fr/design/wp-content/uploads/2022,
```

```
en's SKUs in the following invoice, then read it aloud:"  
elysee.fr/design/wp-content/uploads/2022/01/custom-t-shirt-orde
```

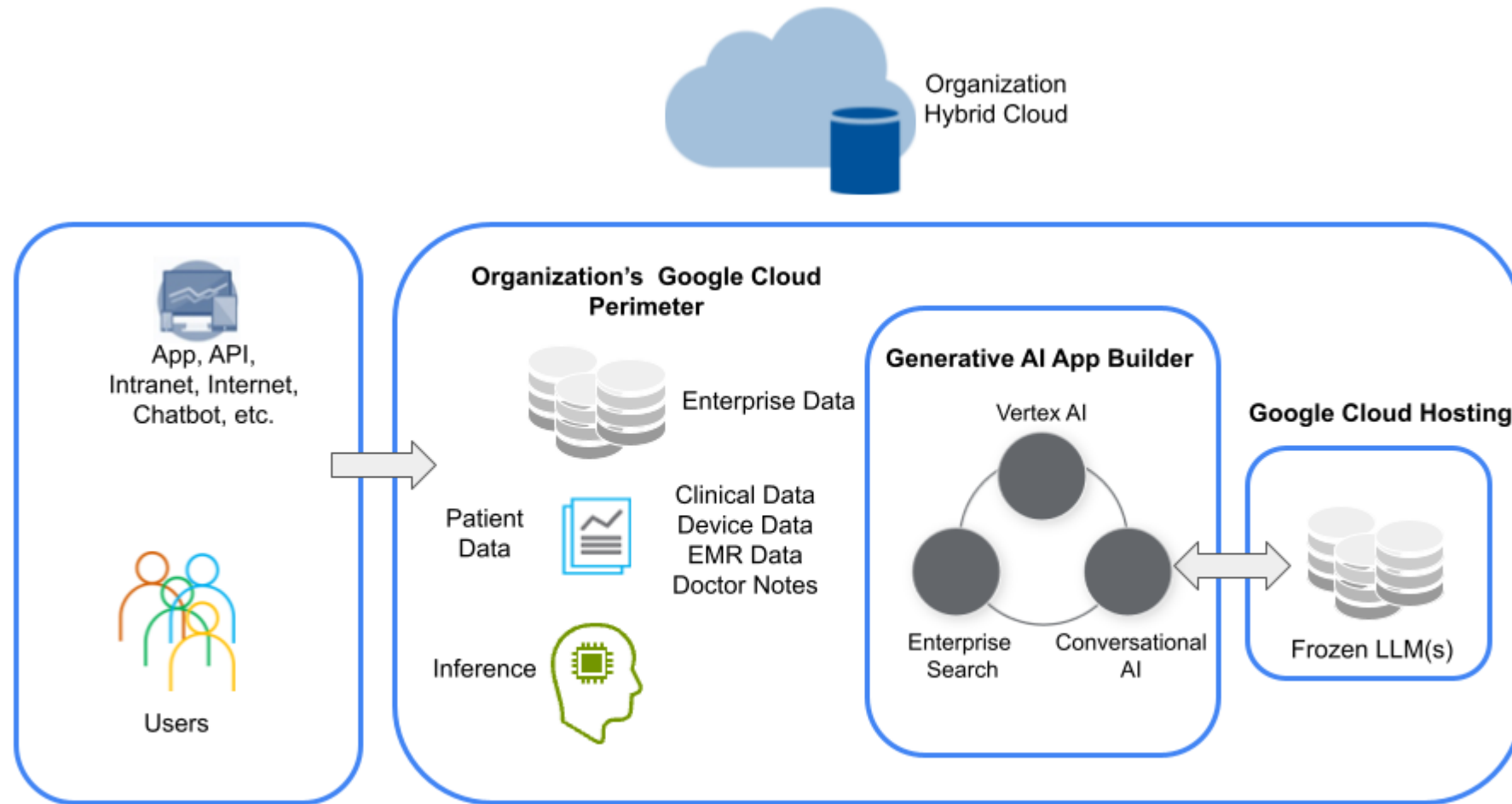
# DALL·E and text generation

```
from langchain.agents import load_tools
from langchain.agents import initialize_agent
tools = load_tools(['dalle-image-generator'])
agent = initialize_agent(tools, model, AgentType.ZERO_SHOT_REACT_
agent.run("Create an image of a halloween night. Return only the
```

```
agent.run("search a video trailer of Avatar: the way of water. Return only 1 video.
transcribe the youtube video
and return a review of the trailer. Generate an image based on the video transcription")
```



## LLM App Enterprise Architecture





## Approach for Building an LLM

