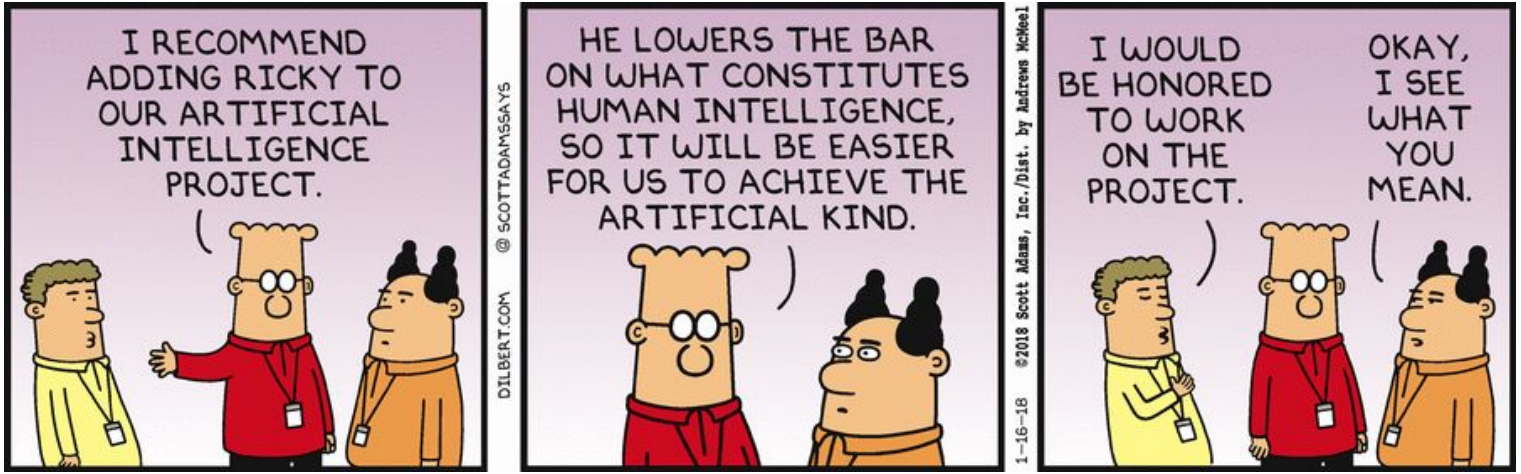


Scan me

Developing GenAI Applications with LangChain

chukmunnlee@gmail.com

https://bit.ly/genai_langchain

Updated the workshop template

- Either re clone
- Copy the files to your existing repository
  - 2_vector_store/2_llm_with_retrievers.py
  - ebook/*.pdf

https://bit.ly/genai_langchain

What are LLMs?

Model that is trained on text

- Eg books, code, chat conversations
- Learn statistical relationships between words and phrases
- Applications include code generation, text summary, translation, etc

Key to LLMs is transformers with attention

- Attention mechanism allows the model to focus on key and relevant information

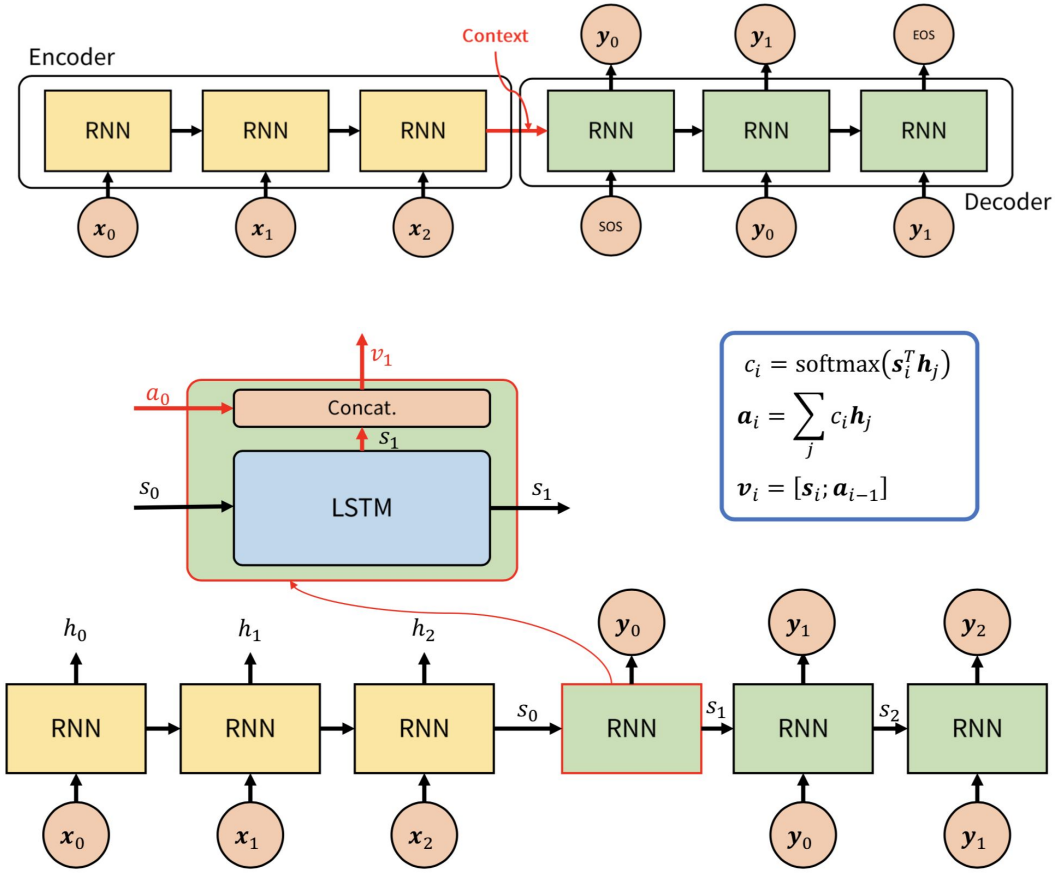


Image from https://heung-bae-lee.github.io/2020/01/21/deep_learning_10/

What is LangChain?

Framework for building LLM applications

- Applications can be written in Python, JavaScript, Java, C# (not official)

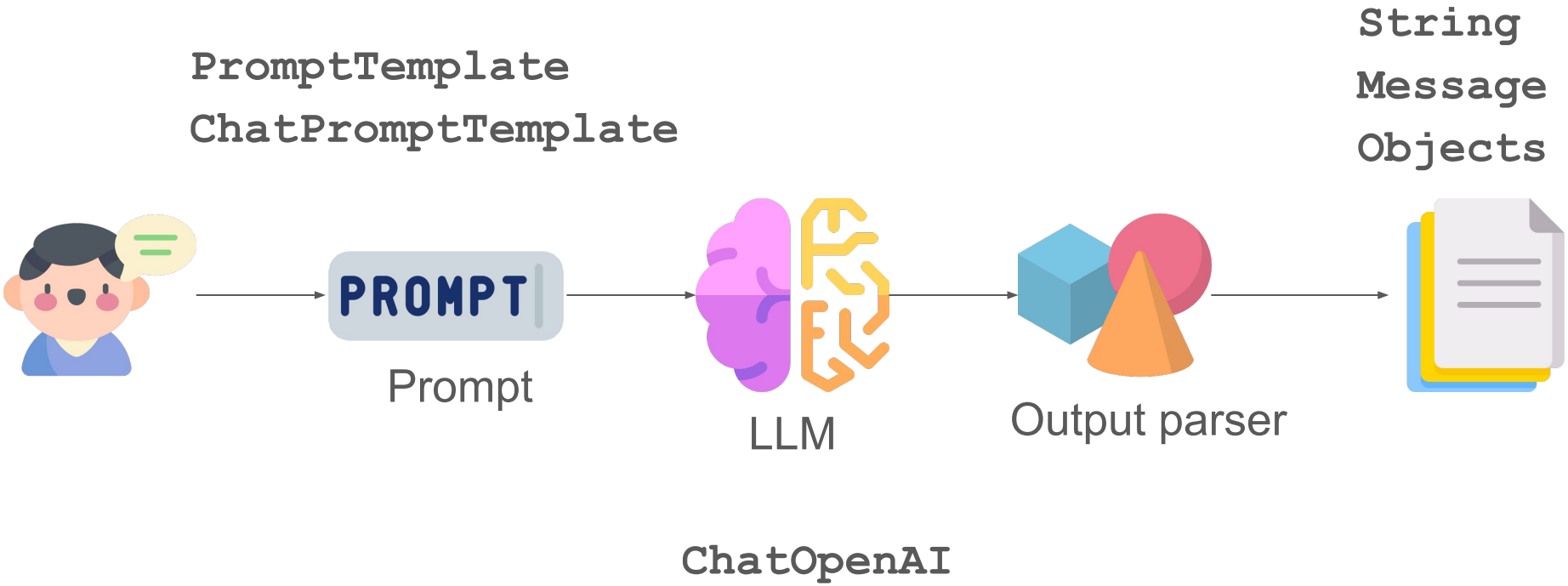
Single API for different LLMs - GPT, Anthropic, Ollama, Vertex, etc

Alternative to LangChain

- LlamaIndex
- Flowise

https://bit.ly/genai_langchain

Basic LLM Application Structure

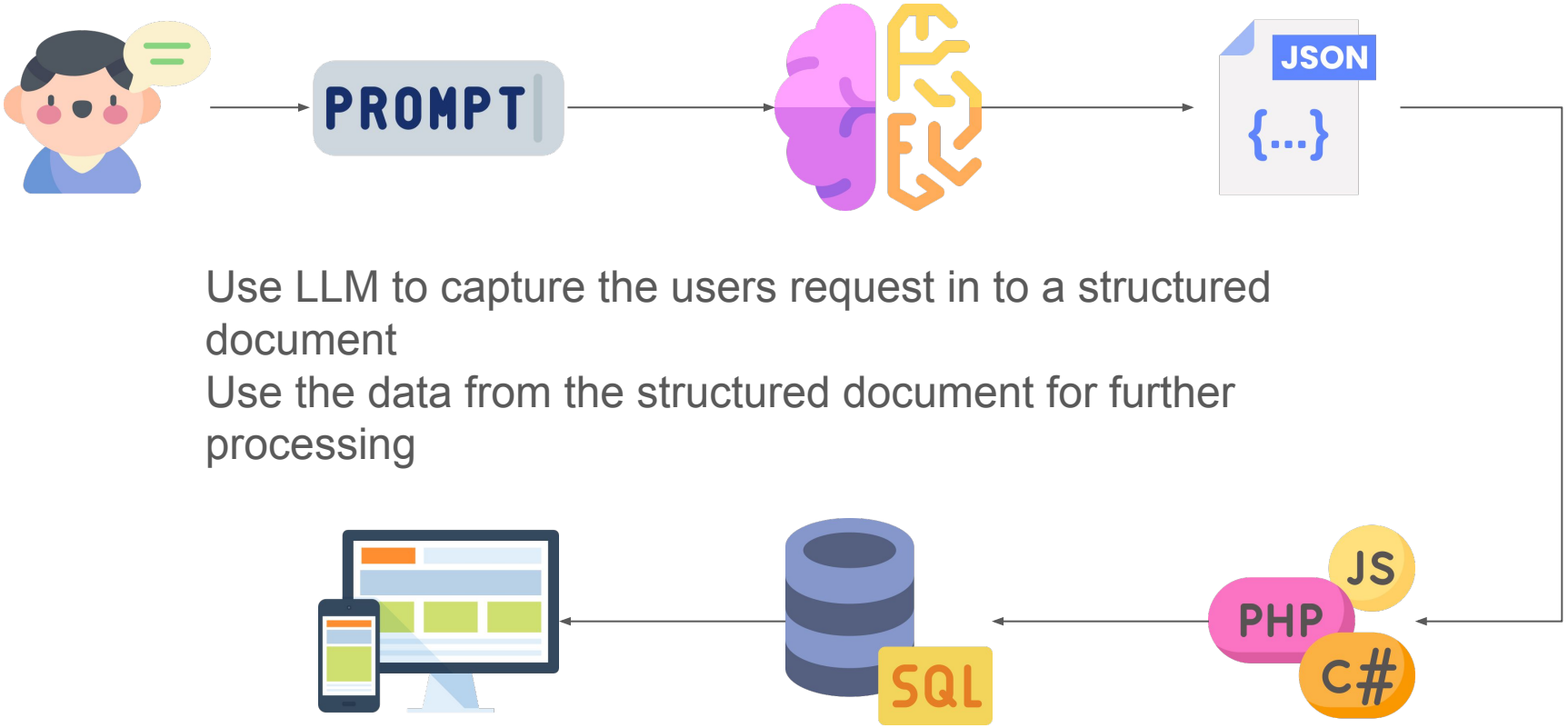


https://bit.ly/genai_langchain

Prompts and LLM

1. Your first prompt and LLM
   - **`0_basic_prompt.py`**
2. Cost of answering a question
3. Conditioning and alignment
   - **`1_chat_messages.py`**
4. Structuring outputs (illustration next slide)
   - **`2_generate_questions_and_answers.py`**

Structure Output



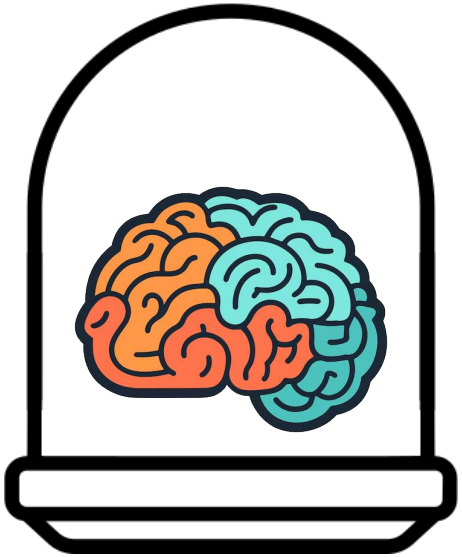
Use LLM to capture the users request in to a structured document

Use the data from the structured document for further processing

https://bit.ly/genai_langchain

Language Models are Self Contained

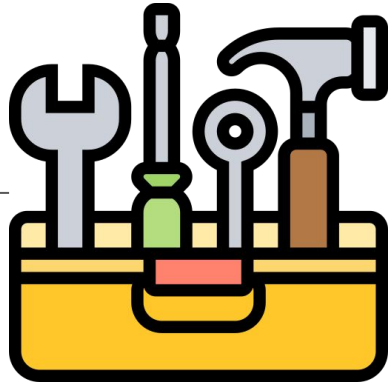
I don’t know



Make up an answer

https://bit.ly/genai_langchain

LLM with External Knowledge



Documents

Lookup proprietary information

Tools

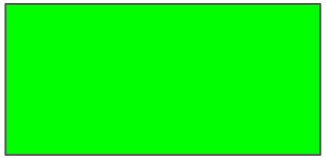
Interact with the real-world to get information

Embeddings

RGBA embedding describe the intensity of each of the component

[199, 21, 133]

[55, 192, 203]



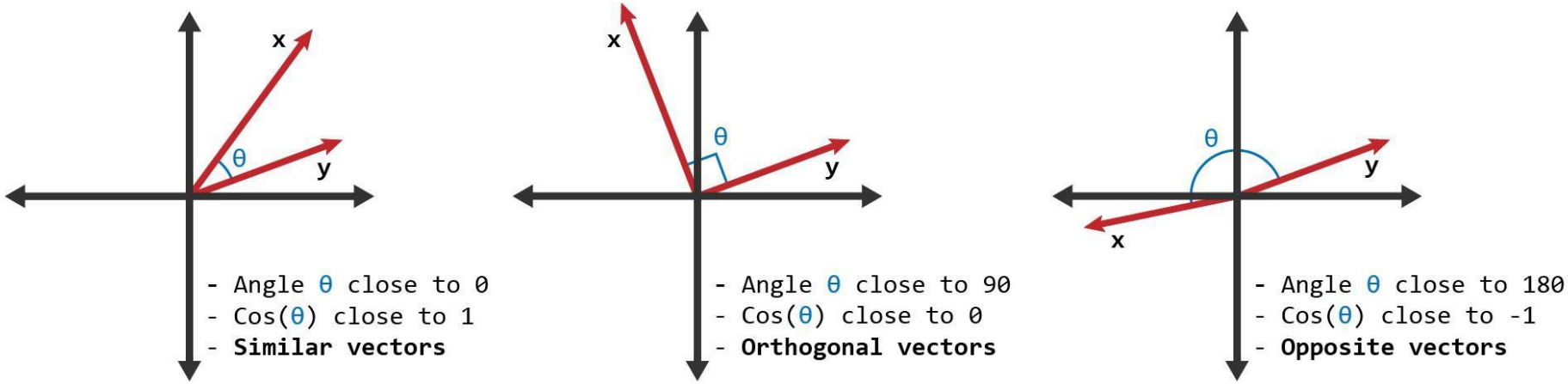
[0, 255, 0]

[199, 21, 133, 20]

More dimension, more information

https://bit.ly/genai_langchain

Cosine Similarity



$$cosine(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

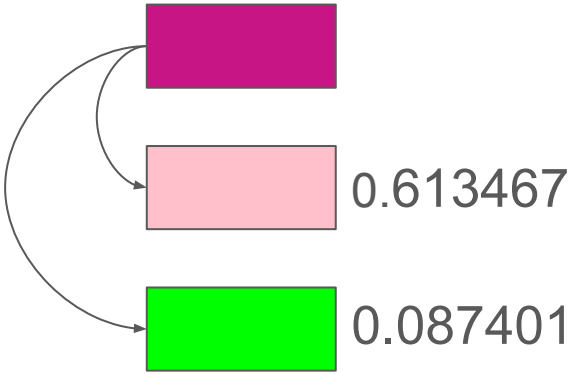
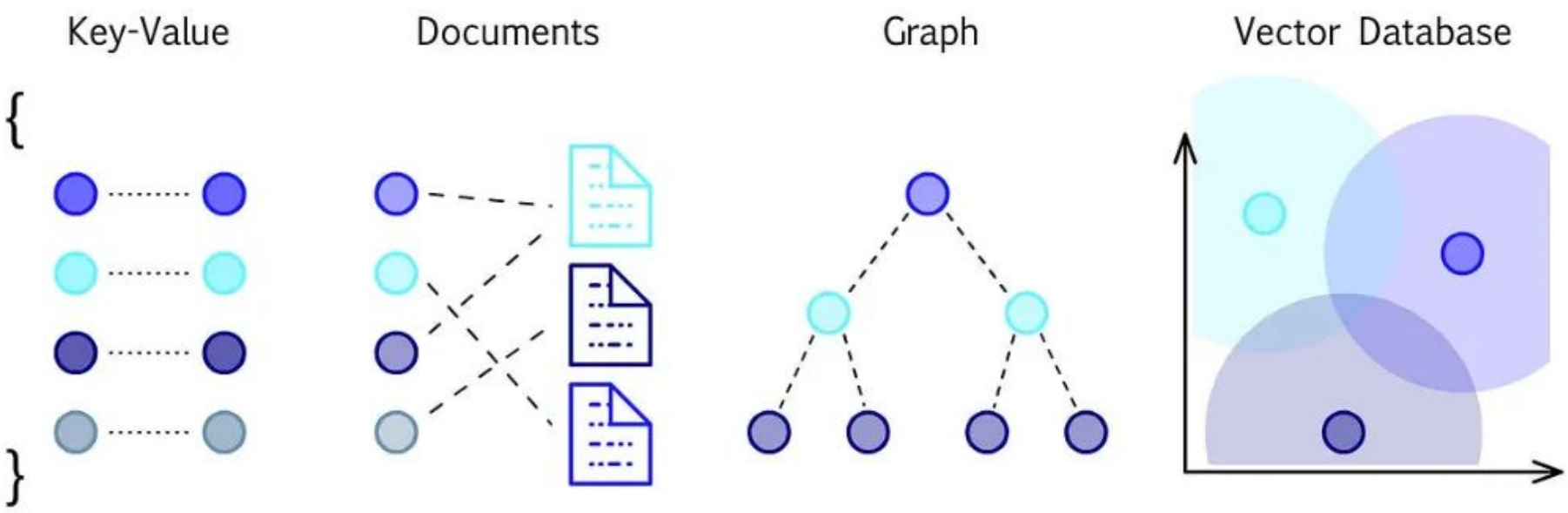


Image from https://www.learndatasci.com/glossary/cosine-similarity/

https://bit.ly/genai_langchain

Database Structure and Use Cases



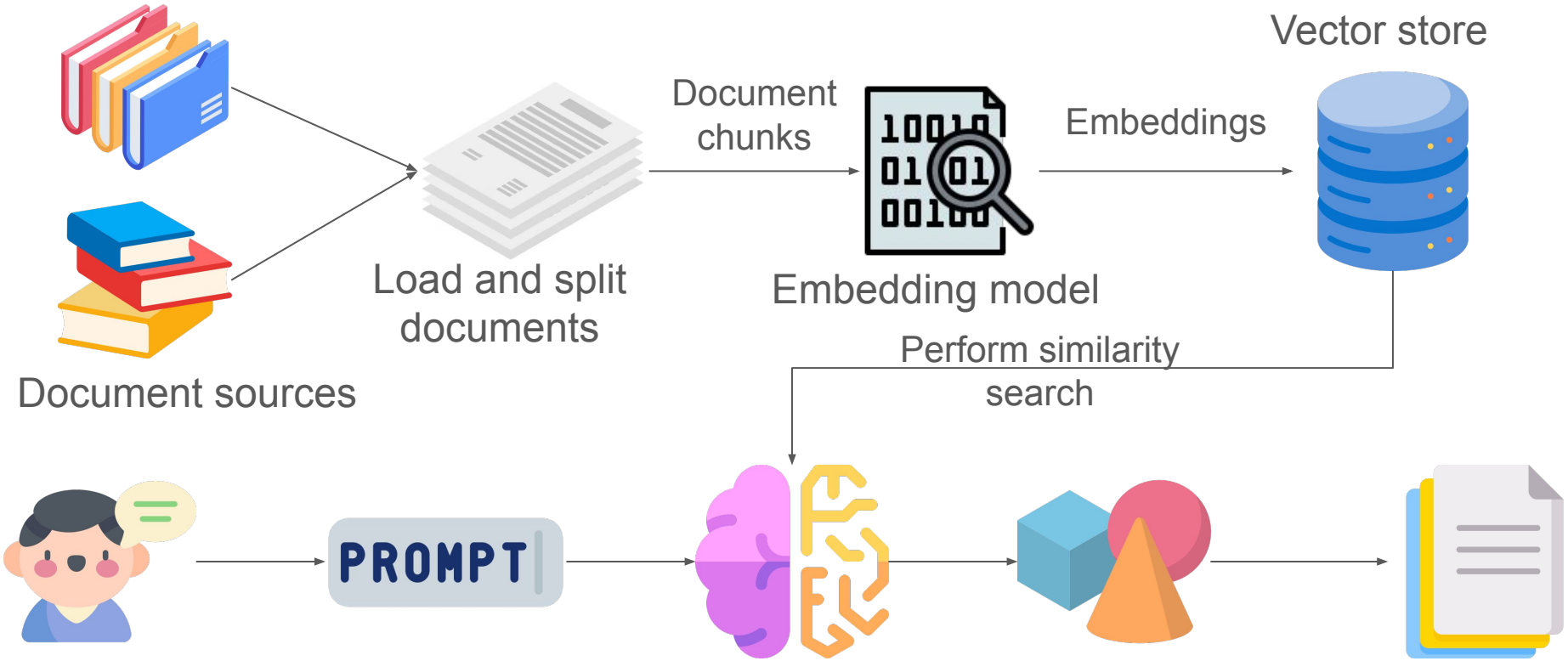
https://bit.ly/genai_langchain

Some Vector Databases

	Dedicated	Supports
Memory		
Disk	Milvus	

https://bit.ly/genai_langchain

Querying from Vector Store



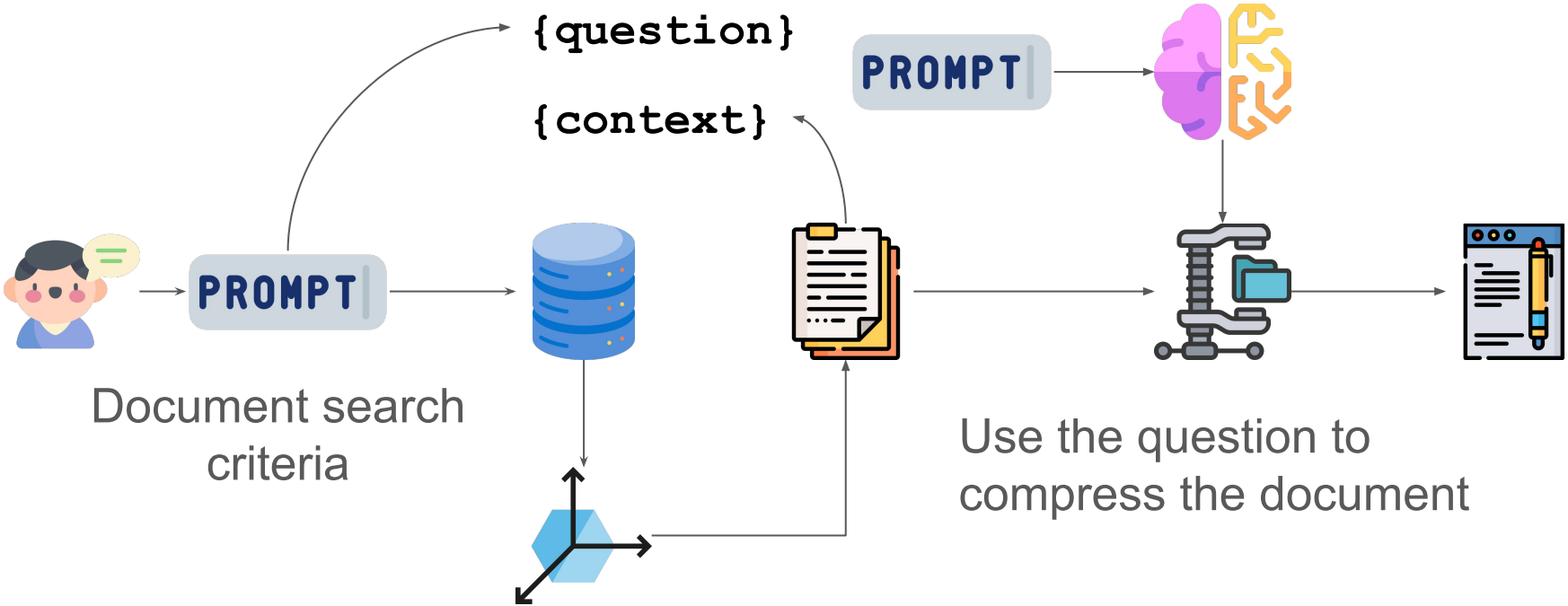
https://bit.ly/genai_langchain

Vector Store

1. Loading and querying documents
   - **`0_load_and_query.py`**
2. Contextual compression (illustration next slide)
   - **`1_retriever_with_compressor.py`**
   - **`load_vector_db.py`**
3. Retrieving from vector store (illustration next next slide)
   - **`2_llm_with_retrievers.py`**

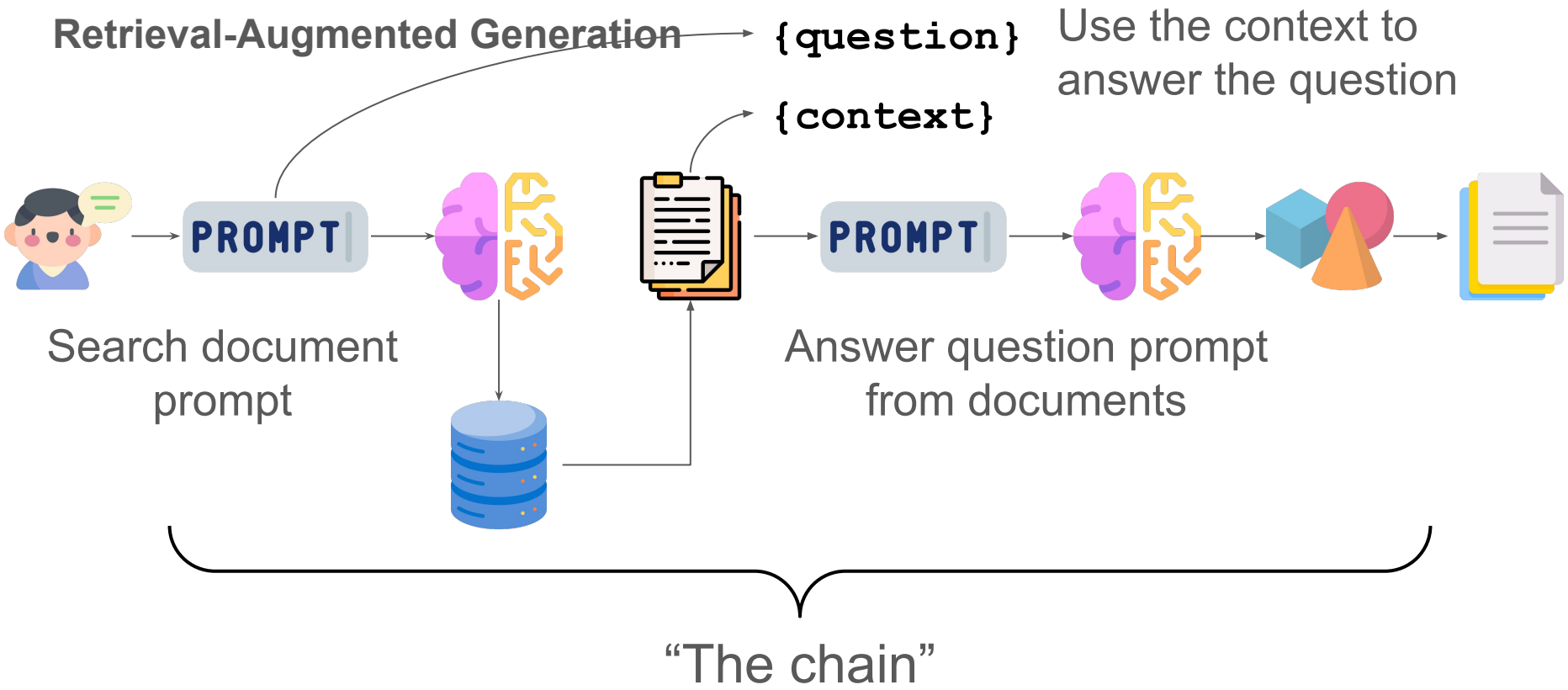
https://bit.ly/genai_langchain

Contextual Compression



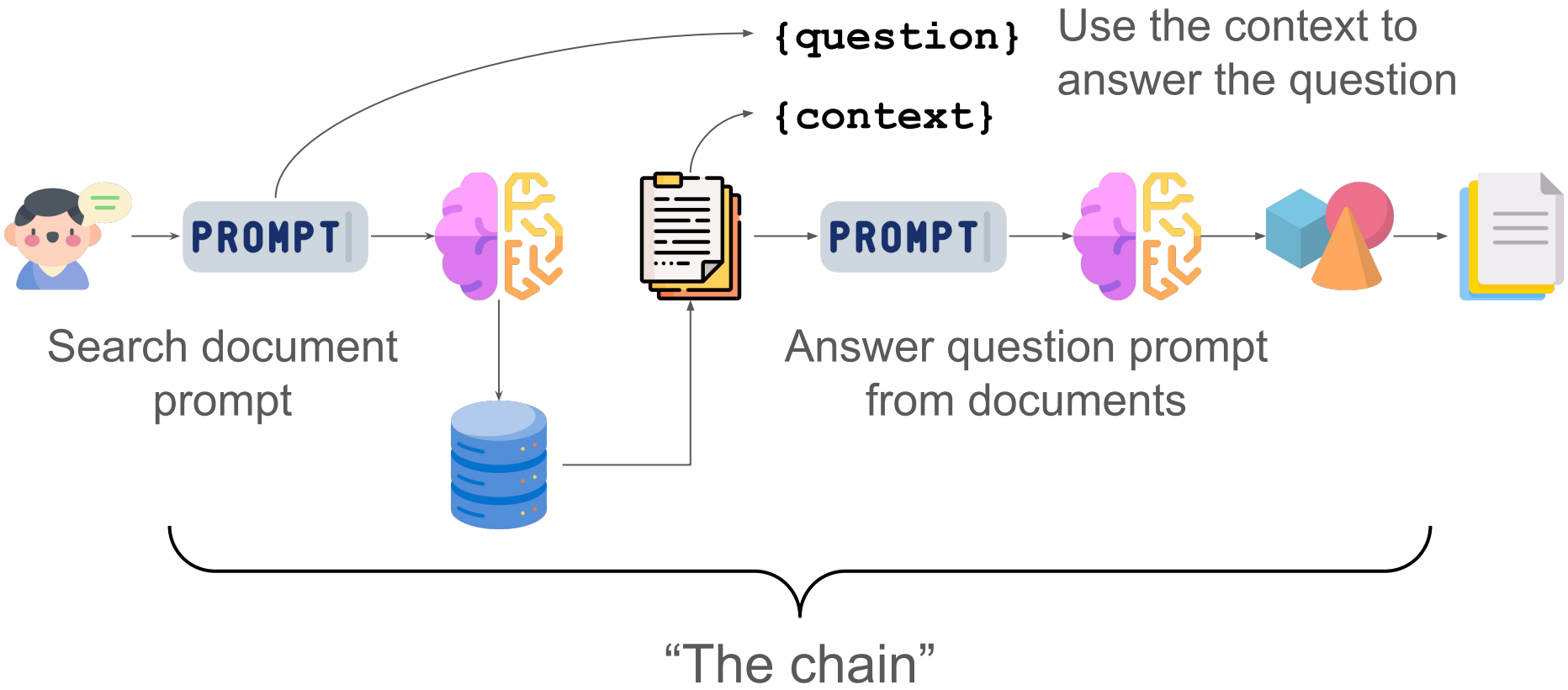
https://bit.ly/genai_langchain

Augmenting LLM with Vector Store



https://bit.ly/genai_langchain

Augmenting LLM with Vector Store



Agents

Agents determines how best to answer a question with a given set of tools

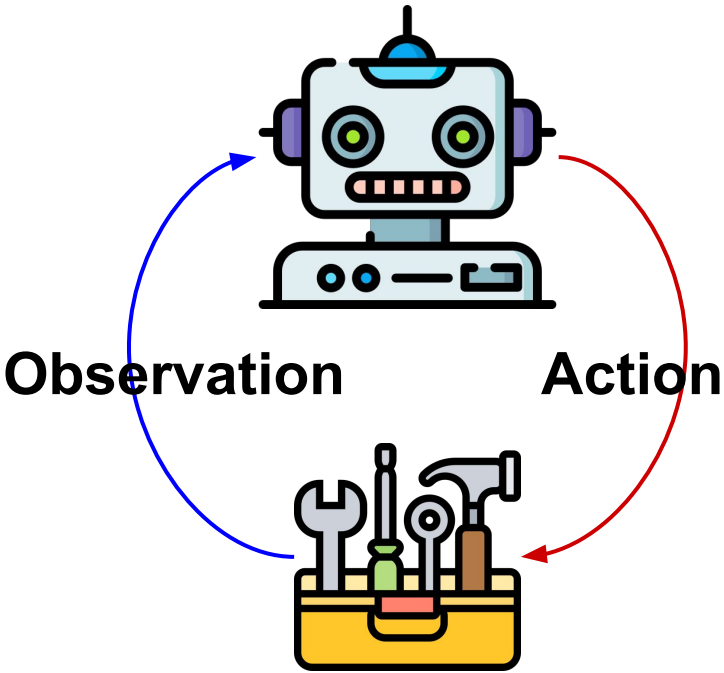
- Tools include build-in eg wikipedia, duck duck go search
- Custom tools

May iterate to refine the answer

-

Memory allow agents to recall previous interactions

- Infer from past interactions eg ‘the problem’



Example of Agent's Prompt

https://smith.langchain.com/hub/hwchase17/react-chat

Key variables

- **`{tool_names}`**, **`{tools}`** - tool names and custom tools
- **`{chat_history}`** - past conversation from memory
- **`{agent_scratchpad}`** - used by the agent
- **`{input}`** - question the agent is answering

https://bit.ly/genai_langchain

Agents

1. Agents and tools
   - `0_tools.py`
2. Conversation agents
   - `1_conversation_agent.py`

Capstone Example

More advance example in 4_capstone

- Retrievers as tools
- Preloading the conversation history

Workshop solution

https://github.com/chukmunnlee/jul13-2024-geekshacking-genai-workshop.git

Thank you for attending!