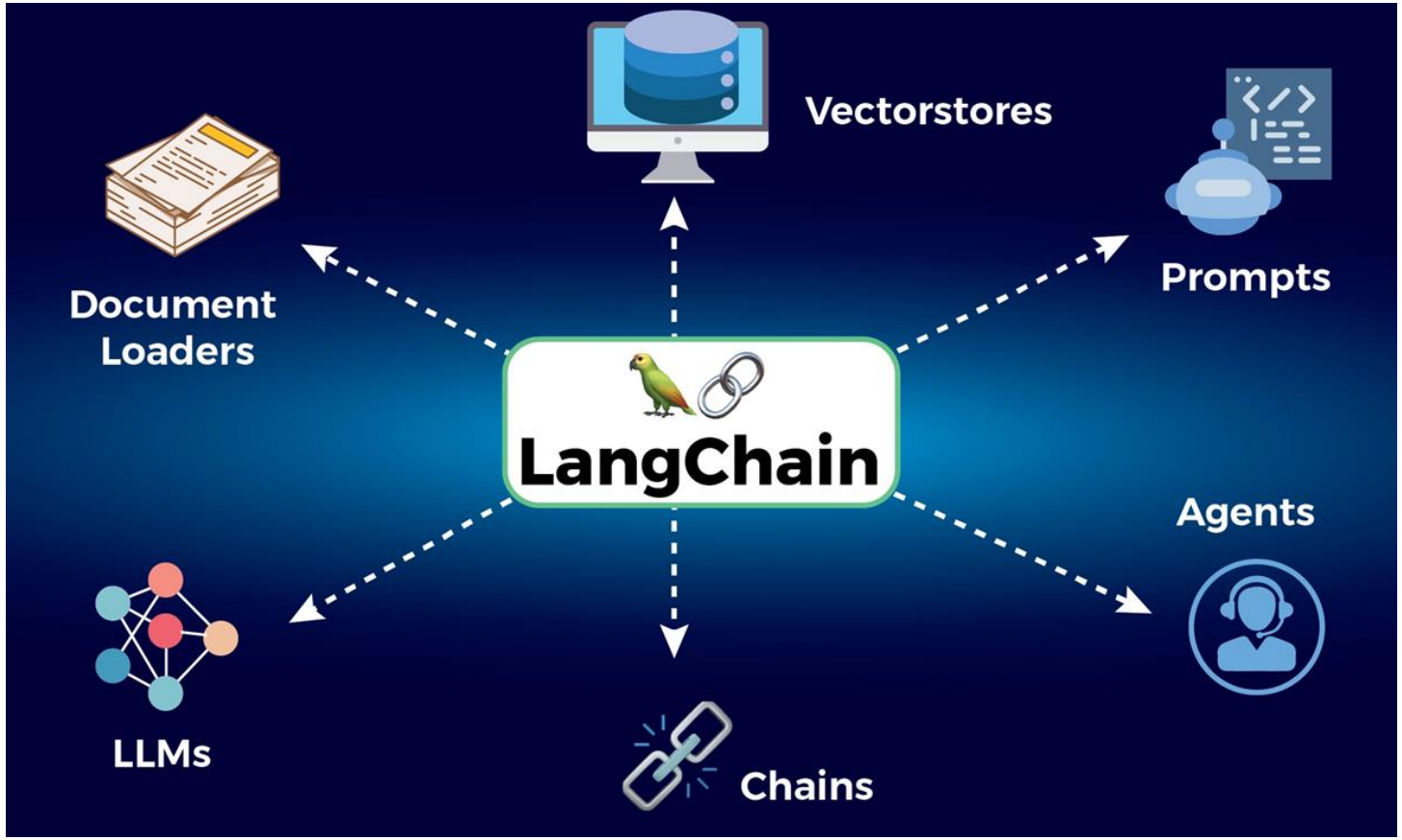
LangChain 퀵스타트



박경규

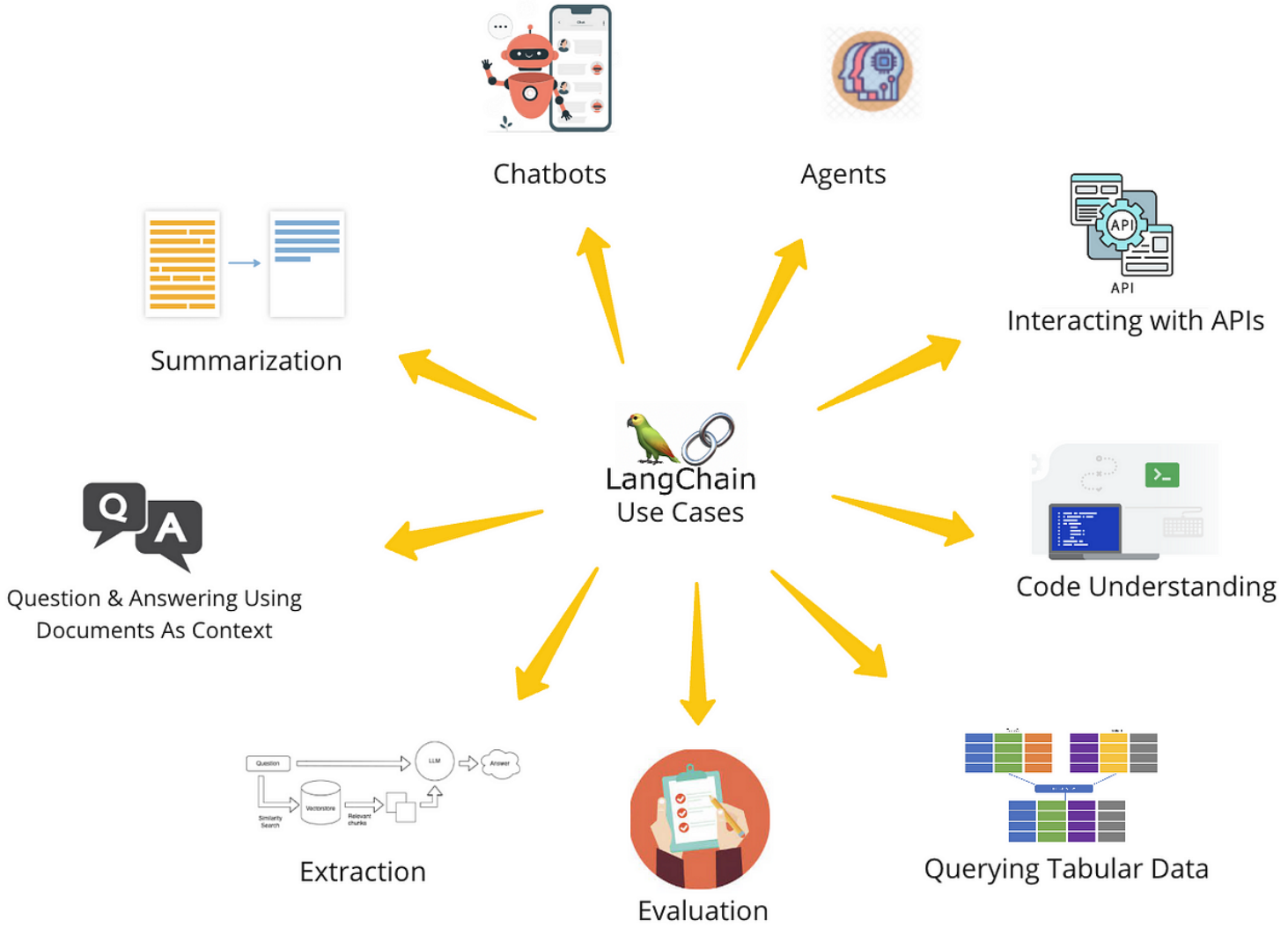
LangChain

https://github.com/langchain-ai/langchain

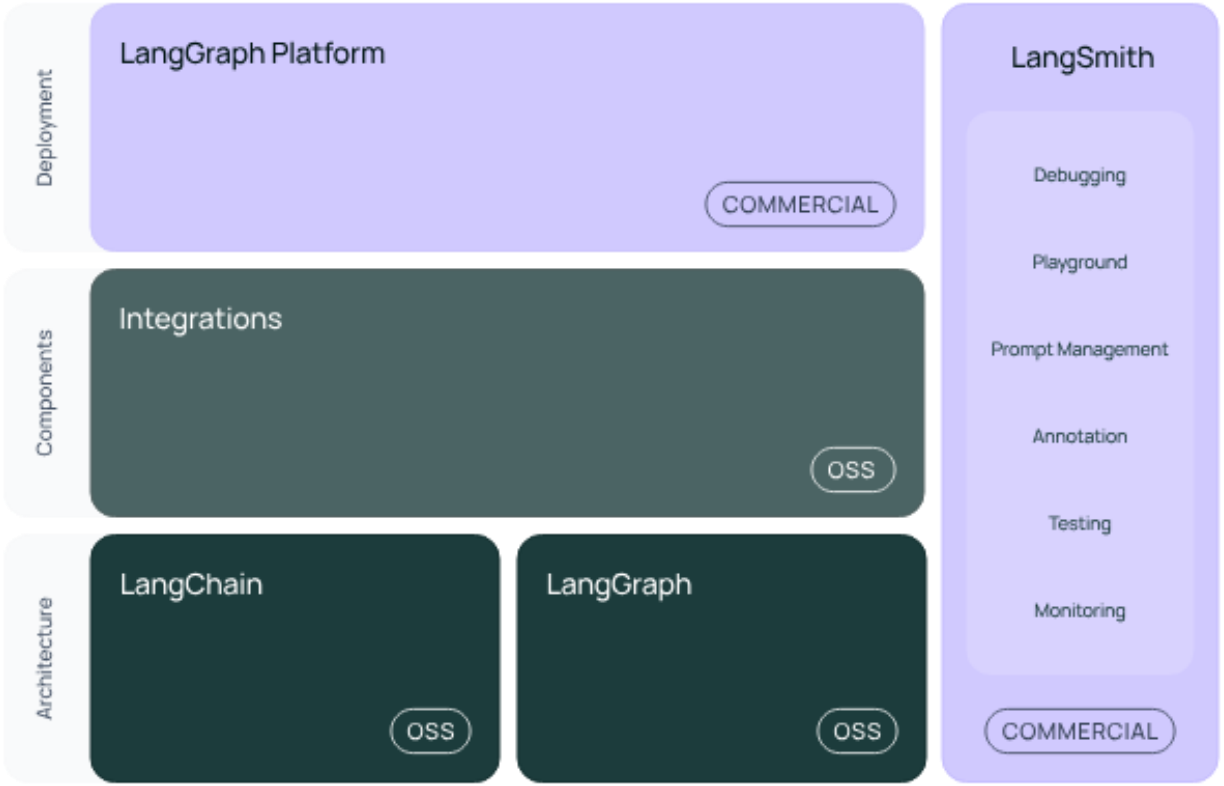
LangChain은 언어 모델로 구동되는 애플리케이션을 개발하기 위한 프레임워크입니다.

다음과 같은 애플리케이션을 지원합니다.

- **문맥 인식**(Are context-aware):
  언어 모델을 문맥 소스(프롬프트 지침, 몇 가지 예시, 응답의 근거가 되는 콘텐츠 등)에 연결합니다.
- **추론**(Reason): 언어 모델에 의존하여 추론(제공된 컨텍스트에 따라 답변하는 방법, 취해야 할 조치 등)



LangChain 아키텍처



- **Langchain Core**
  - 다양한 컴포넌트에 대한 기본 추상화와 이를 함께 구성하는 방법 포함
  - Chat Model, Vector Store, Tool 등과 같은 핵심 구성 요소 인터페이스 정의

- **LangChain**
  - 애플리케이션 Cognitive 아키텍처를 구성하는 Chain과 Retrieval Strategy 포함

- **Integration Packages**
  - langchain-openai, langchain-anthropic 등
  - https://python.langchain.com/docs/integrations/providers/

- **Langchain Community**
  - LangChain 커뮤니티에서 유지 관리하는 third-party 통합 포함

- **LangGraph**
  - LLM 기반 다중 액터 애플리케이션을 구축하는 것을 목표로 하는 LangChain extension

- **LangServe**
  - LangChain chain을 REST API로 배포하는 패키지

- **LangSmith**
  - LLM 애플리케이션을 디버깅, 테스트, 평가, 모니터링하는 개발자 플랫폼

LangChain API

langchain-core

- agents
- beta
- caches
- callbacks
- chat_history
- chat_loaders
- chat_sessions
- document_loaders
- documents
- embeddings
- example_selectors
- exceptions
- globals
- indexing
- language_models
- load
- messages
- output_parsers
- outputs
- prompt_values
- prompts
- rate_limiters
- retrievers
- runnables
- stores
- structured_query
- sys_info
- tools
- tracers
- utils
- **vectorstores**

langchain

- agents
- callbacks
- chains
- chat_models
- embeddings
- evaluation
- globals
- hub
- indexes
- memory
- model_laboratory
- output_parsers
- retrievers
- runnables
- smith
- storage

langchain-community

- adapters
- agent_toolkits
- agents
- cache
- callbacks
- chains
- chat_loaders
- chat_message_histories
- chat_models
- cross_encoders
- docstore
- document_compressors
- document_loaders
- document_transformers
- embeddings
- example_selectors
- graph_vectorstores
- graphs
- indexes
- llms
- memory
- output_parsers
- query_constructors
- retrievers
- storage
- tools
- utilities
- utils
- vectorstores

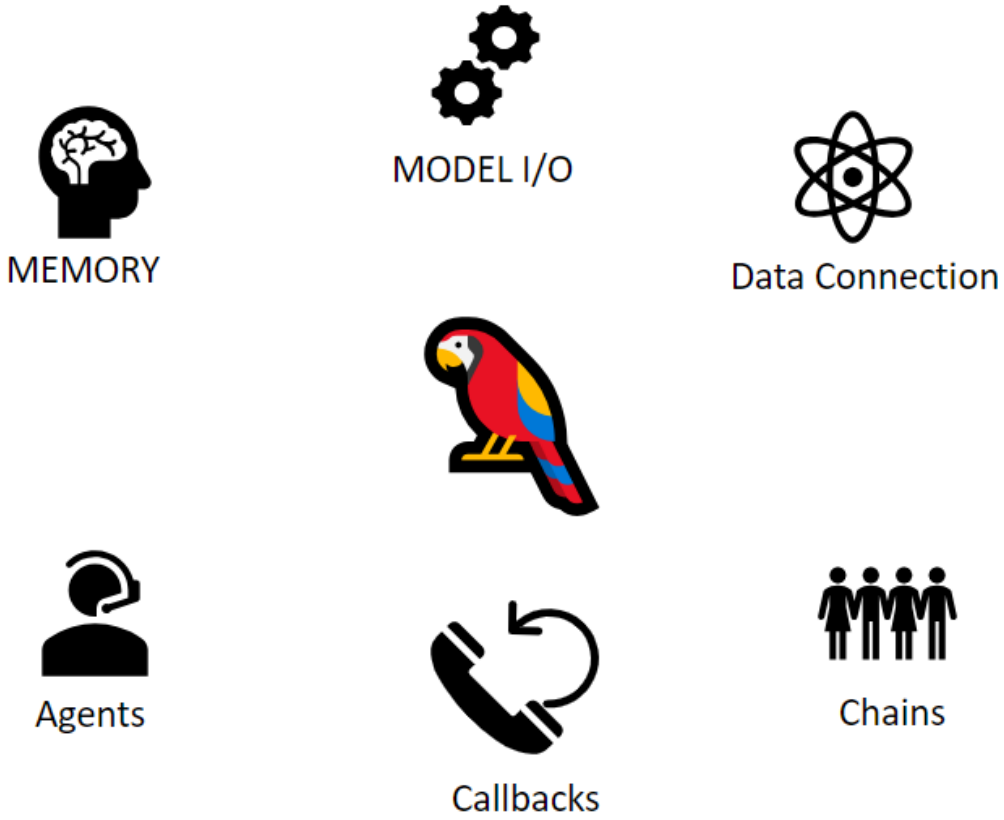
https://python.langchain.com/api_reference/core/index.html

Provider Integration

Provider	Package	Downloads	Latest	JS
OpenAI	langchain-openai	12M/month	v0.3.9	✅
Google VertexAI	langchain-google-vertexai	12M/month	v2.0.16	✅
Google Community	langchain-google-community	4.7M/month	v2.0.7	❌
AWS	langchain-aws	2.2M/month	v0.2.17	✅
Anthropic	langchain-anthropic	2.1M/month	v0.3.10	✅
Google Generative AI	langchain-google-genai	1.4M/month	v2.1.1	✅
Ollama	langchain-ollama	920k/month	v0.3.0	✅
Cohere	langchain-cohere	814k/month	v0.4.3	✅
PowerScale RAG Connector	powerscale-rag-connector	129/month	v1.0.9	❌
Modelscope	langchain-modelscope	124/month	v0.1.1	❌
Pull Md	langchain-pull-md	113/month	v0.1.1	❌
Fmp Data	langchain-fmp-data	101/month	v0.1.0	❌
Tilores	tilores-langchain	82/month	v0.2.0	❌
Oceanbase	langchain-oceanbase	83/month	v0.2.0	❌
Pipeshift	langchain-pipeshift	80/month	v0.1.1	❌
Lindorm Integration	langchain-lindorm-integration	72/month	v0.1.1	❌
Naver	langchain-naver-community	111/month	v0.1.1	❌

https://python.langchain.com/docs/integrations/providers/

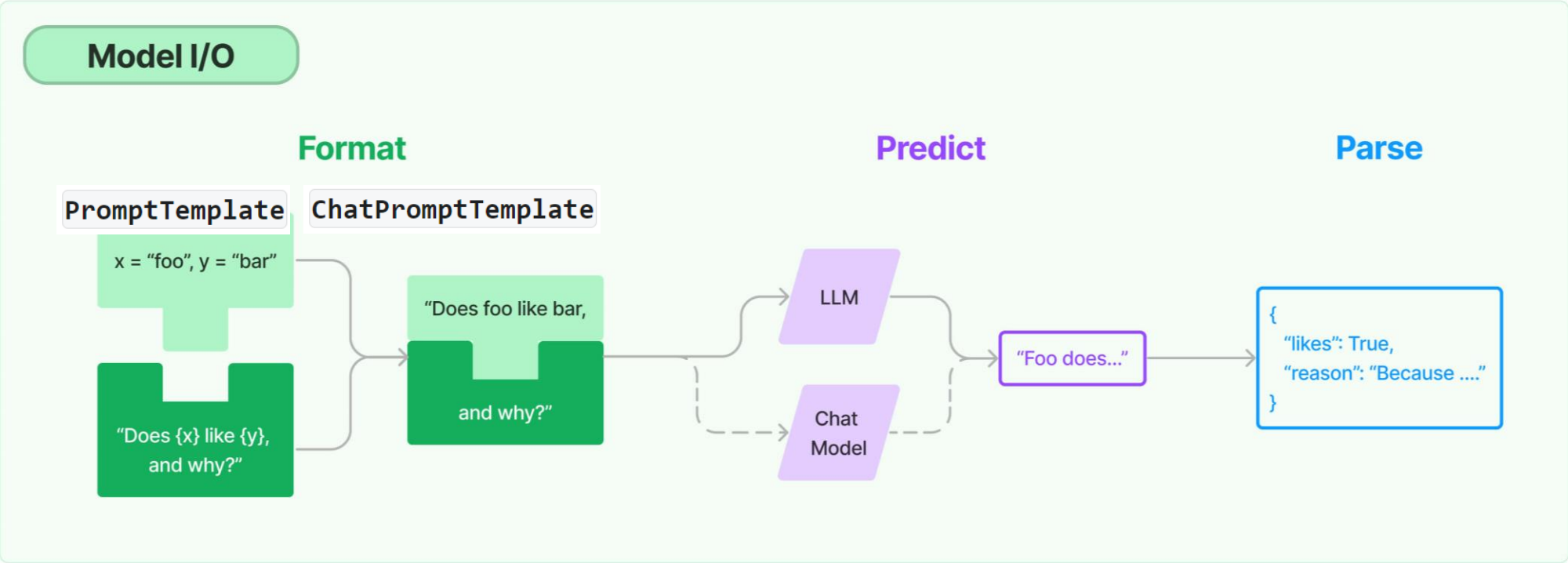
LangChain 모듈



- Model I/O : 언어 모델과의 인터페이스
- Data Connection : 데이터와의 인터페이스
- Chains : 호출 시퀀스 구축
- Callbacks : 체인의 중간 단계를 기록 및 스트리밍
- Agents : 상위 지시문이 주어지면 체인이 사용할 도구(Tool)을 선택할 수 있도록 함
- Memory : 체인 실행 간에 애플리케이션 상태 유지

Model I/O

모든 언어모델 애플리케이션의 핵심 요소는 Model입니다.
Model I/O는 LangChain이 모든 언어모델과 인터페이스 할 수 있는 빌딩 블록을 제공합니다.



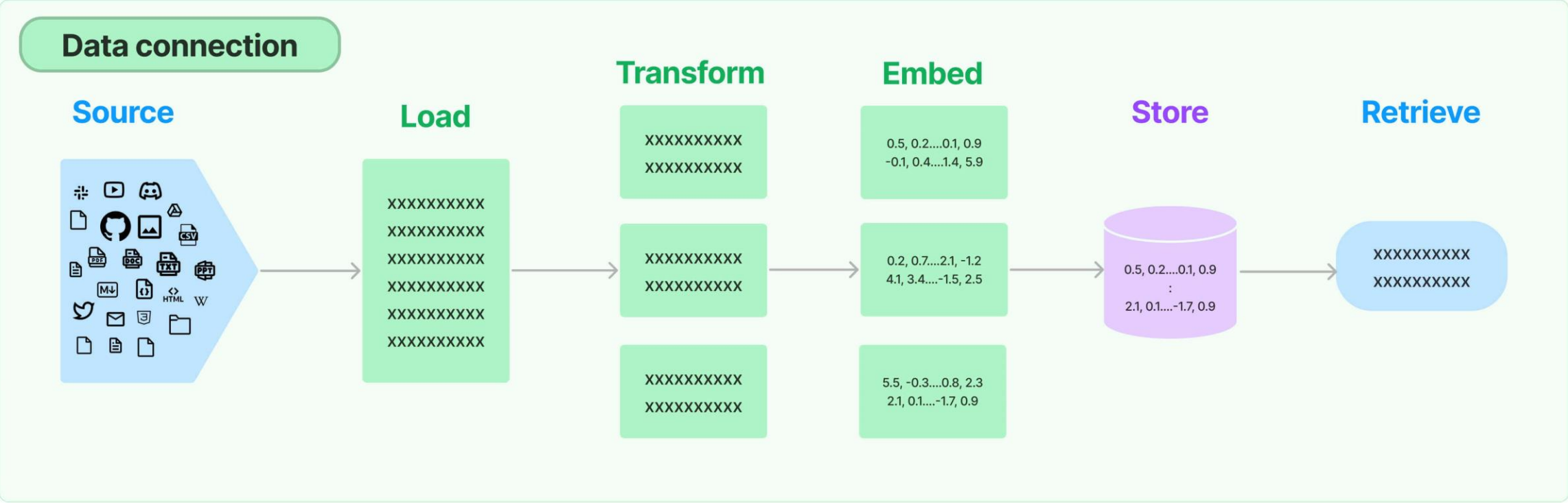
- CSV parser
- Datetime parser
- Enum parser
- JSON parser
- OpenAI Functions
- OpenAI Tools
- Output-fixing parser
- Pandas DataFrame Parser
- Pydantic parser
- Retry parser
- Structured output parser
- XML parser
- YAML parser

This is documentation for LangChain v0.1, which is no longer actively maintained. Check out the docs for the latest version here.

https://python.langchain.com/v0.1/docs/modules/model_io/

Retrieval

많은 LLM 애플리케이션에는 사용자 데이터가 필요하며, 이를 달성하는 주요 방법은 검색 증강 생성(RAG, Retrieval-Augmented Generation)을 사용하는 것입니다. LangChain은 RAG 애플리케이션을 위한 빌딩 블록을 제공합니다.



This is documentation for LangChain v0.1, which is no longer actively maintained. Check out the docs for the latest version here.

https://python.langchain.com/v0.1/docs/modules/data_connection/

Retrieval

Document loader

- LangChain은 100가지가 넘는 다양한 Document loader를 제공하며, 다양한 유형의 문서(HTML, PDF, 코드)를 로드할 수 있는 통합 기능을 제공합니다.

Text Splitting

- 검색의 핵심은 문서에서 관련 부분만 가져오는 것으로, 문서를 준비하기 위한 몇 가지 변환 단계가 포함됩니다.
- Text Splitting은 큰 문서를 작은 덩어리로 분할(청크)하는 것이며, 분할 알고리즘과 특정 유형(코드, 마크다운 등)에 최적화된 로직을 제공합니다.

Text embedding model

- 검색의 또 다른 핵심 부분은 임베딩을만드는 것입니다.
- 임베딩은 텍스트의 의미론적 의미를 포착하여 유사한 텍스트를 빠르고 효율적으로 찾을 수 있게 해줍니다.

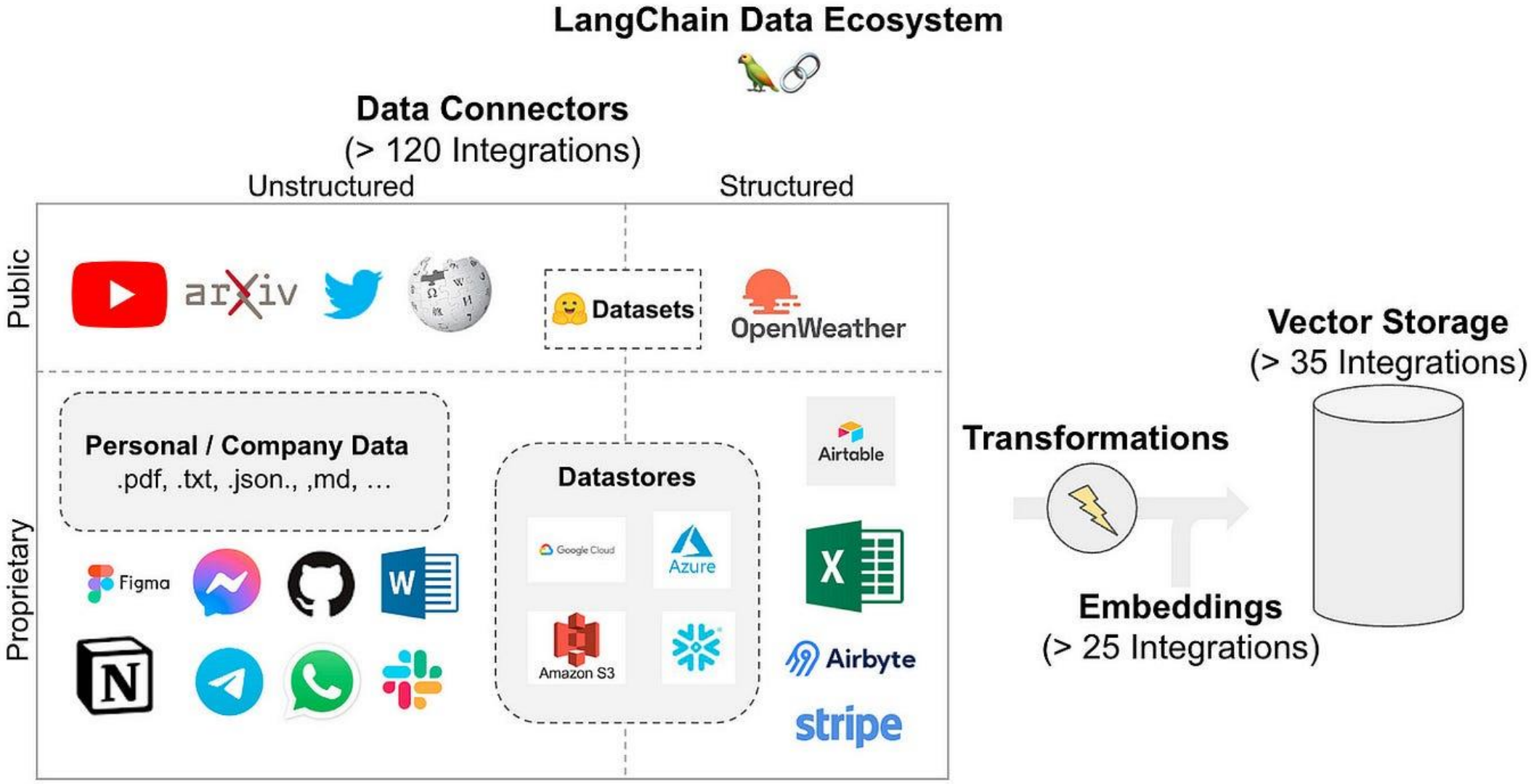
Vector store

- 벡터 스토어는 임베딩벡터 데이터의 효율적인 저장과 검색을 지원합니다.
- LangChain은 오픈소스 로컬 스토어부터 클라우드 호스팅 벡터스토어까지 다양한 벡터 스토어와의 통합을 제공합니다.

Retriever

- LangChain은 벡터 스토어에 저장된 데이터를 검색하는 다양한 알고리즘을 지원합니다.
- LangChain은 시맨틱 검색을 지원하며, 성능 향상 알고리즘이 있습니다.

Retrieval

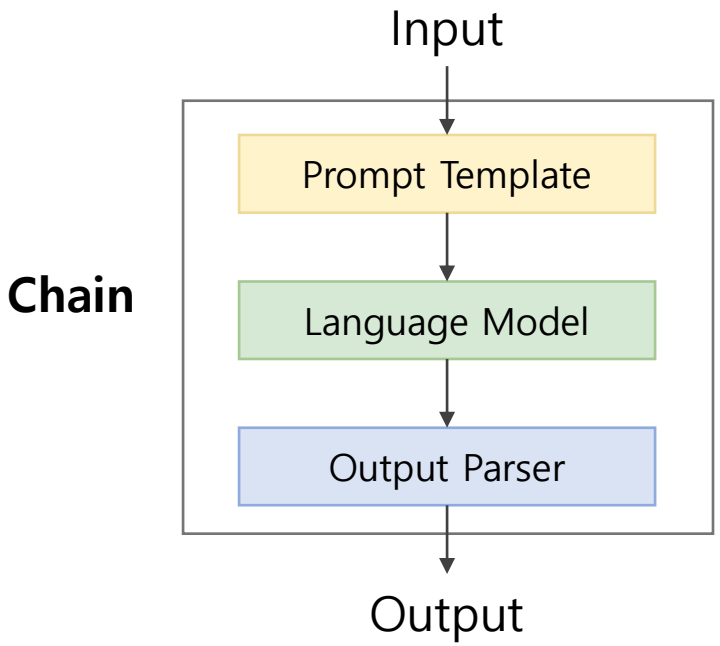


https://nikhilpentapalli.medium.com/langchain-in-detail-6cf8ecfb2464

Chain

Chain은 LLM, 도구 또는 데이터 전처리 단계에 대한 호출 시퀀스입니다.
Chain 을 구성하는 기본적인 방법은 LCEL(LangChain Expression Language)입니다.
| 기호는 서로 다른 구성 요소를 연결하고, 한 구성 요소의 출력을 다음 구성 요소의 입력으로 전달합니다.

```
chain = prompt | model | output_parser
```



Agent

에이전트는 LLM을 사용하여 수행할 작업 시퀀스를 선택하는 클래스입니다.
Chain에서는 작업 시퀀스를 하드코딩 하지만, 에이전트에서는 LLM을 추론 엔진으로 사용하여 어떤 작업을 어떤 순서로 수행할지 결정합니다. 에이전트는 작업을 위한 도구와 툴킷을 선택하여 사용합니다.

■ Agent 주요 특징

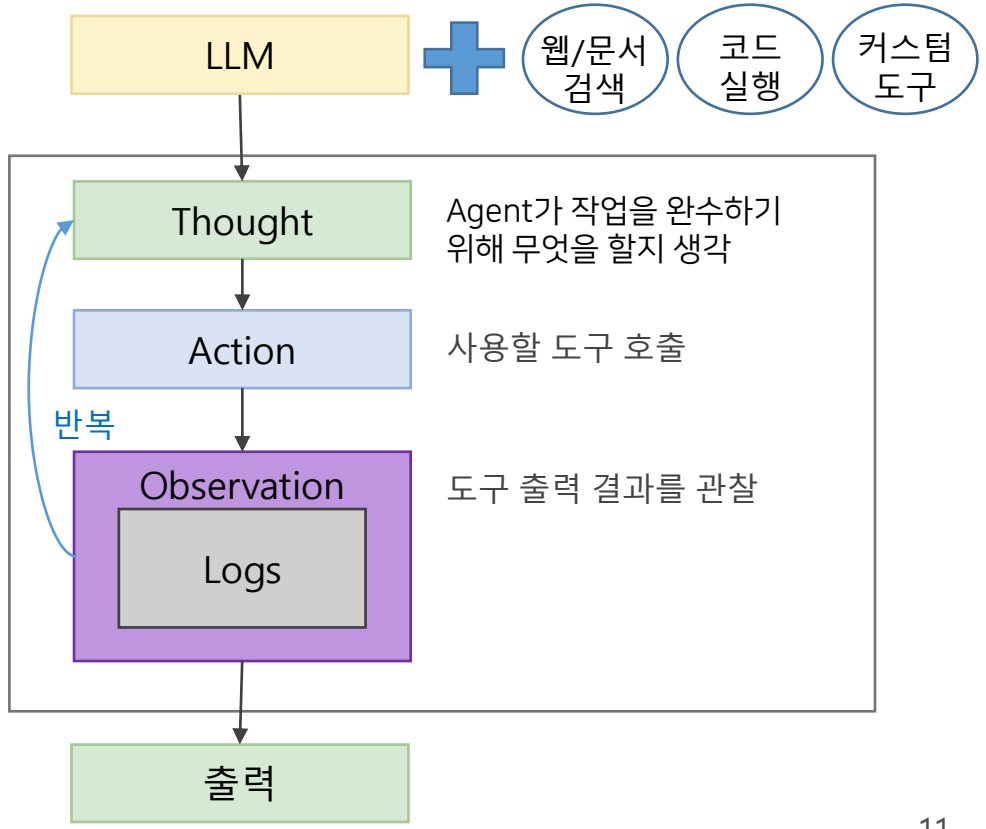
- 목표 기반 행동: 주어진 목표를 달성하기 위해 행동
- 자율성: 목표가 주어지면 자동으로 작동
- 감지: 주변 환경에서 정보를 수집

■ LangChain Agent 클래스

```
BaseSingleActionAgent --> LLMSingleActionAgent
                          OpenAIFunctionsAgent
                          XMLAgent
                          Agent --> <name>Agent  # Examples: ZeroShotAgent, ChatAgent


BaseMultiActionAgent  --> OpenAIMultiFunctionsAgent
```

■ ReAct Agent의 작업 처리 FLOW



Tools

도구는 에이전트, 체인 또는 LLM이 세상과 상호 작용하는 데 사용할 수 있는 인터페이스입니다.

Search

Bing Search
Brave Search
DuckDuckgoSearch
Exa Search
Google Search
Google Serper
Jina Search
Mojeek Search
SearchApi
SearxNG Search
SerpAPI
Tavily Search
You.com Search

Web Browsing

AgentQL Toolkit
MultiOn Toolkit
PlayWright Browser Toolkit
Requests Toolkit

Productivity

Github Toolkit
Gitlab Toolkit
Gmail Toolkit
Infobip Tool
Jira Toolkit
Office365 Toolkit
Slack Toolkit
Twilio Tool

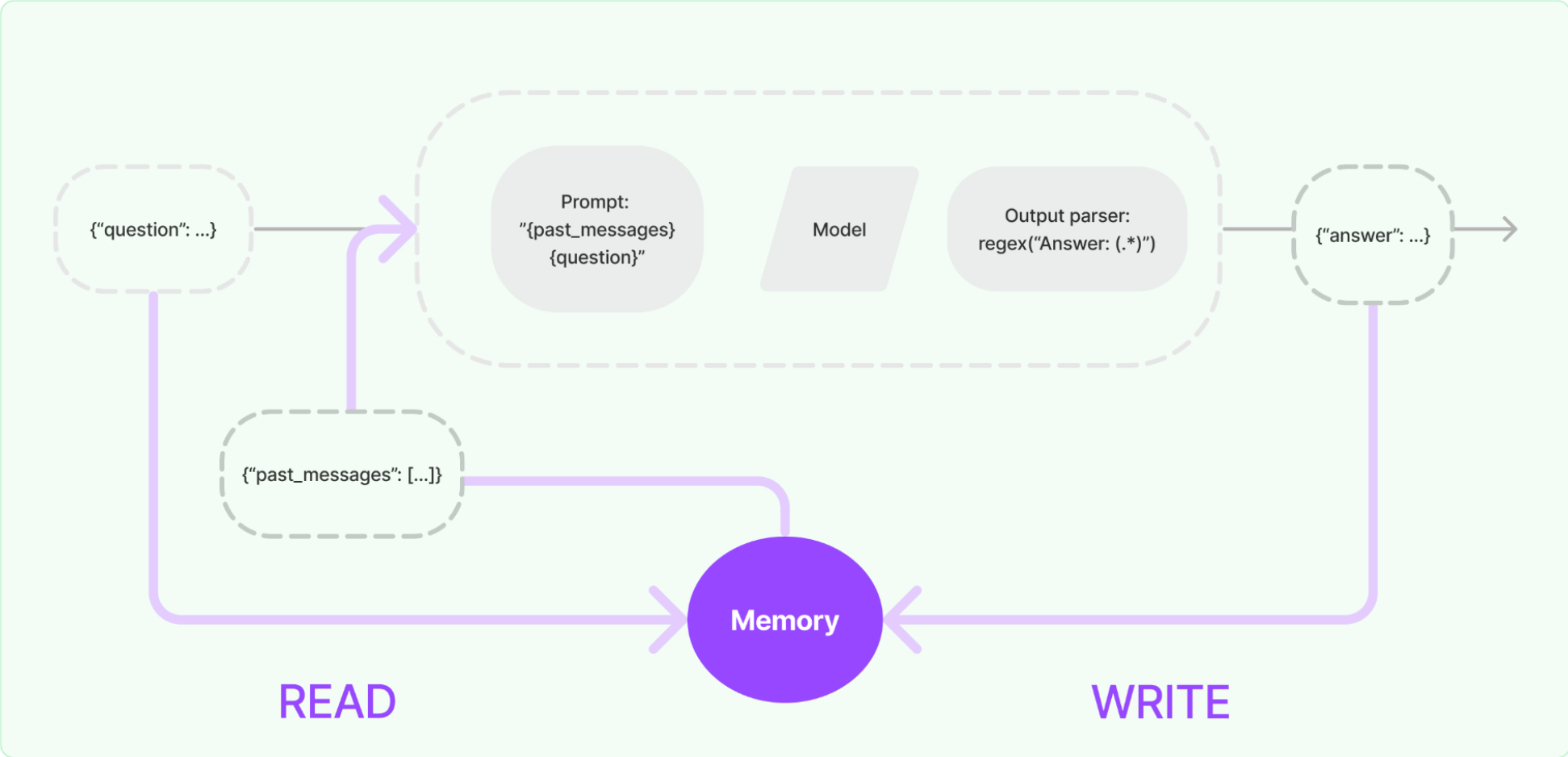
Database

Cassandra Database Toolkit
SQLDatabase Toolkit
Spark SQL Toolkit

https://python.langchain.com/docs/integrations/tools/

Memory

LangChain Memory 는 대화 과정에서 발생하는 정보를 저장하고 관리하여,
보다 자연스럽고 지속적인 대화 경험을 제공할 수 있도록 합니다.

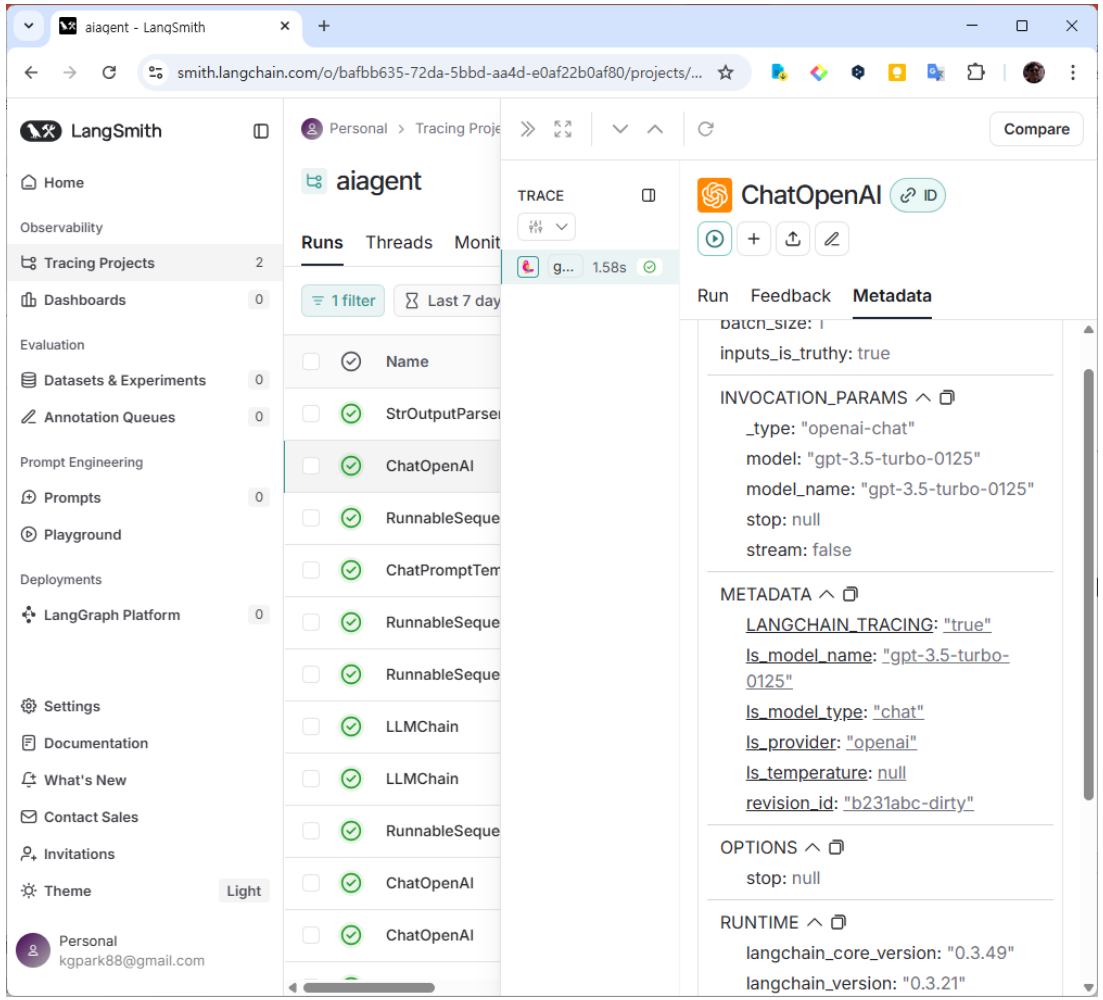


https://python.langchain.com/v0.1/docs/modules/memory/

LangSmith

https://smith.langchain.com/

LLM 애플리케이션을 개발, 디버깅, 최적화할 수 있도록 도와주는 Observability & Debugging 도구입니다.



Observability

- LangSmith에서 Trace를 분석
- 이를 기반으로 메트릭, 대시보드, 알림을 구성

Evals

- 애플리케이션 생산 트래픽을 평가
- 애플리케이션 성능을 평가하고 데이터에 대한 인적 피드백

Prompt Engineering

- LangSmith에서 Trace를 분석
- 자동 버전 제어 및 협업 기능을 사용하여 프롬프트를 반복

https://docs.smith.langchain.com/

LangSmith

■ 환경변수 설정

- export LANGSMITH_TRACING=true
  export LANGSMITH_API_KEY="<langsmith-api-key>"
  # OpenAI 사용 경우
  export OPENAI_API_KEY="<your-openai-api-key>"

Python TypeScript

```python
from openai import OpenAI
from langsmith.wrappers import wrap_openai

openai_client = wrap_openai(OpenAI())

# This is the retriever we will use in RAG
# This is mocked out, but it could be anything we want
def retriever(query: str):
    results = ["Harrison worked at Kensho"]
    return results

# This is the end-to-end RAG chain.
# It does a retrieval step then calls OpenAI
def rag(question):
    docs = retriever(question)
    system_message = """Answer the users question using only the provided information below:

    {docs}""".format(docs="\n".join(docs))

    return openai_client.chat.completions.create(
        messages=[
            {"role": "system", "content": system_message},
            {"role": "user", "content": question},
        ],
        model="gpt-4o-mini",
    )
```

https://docs.smith.langchain.com/observability

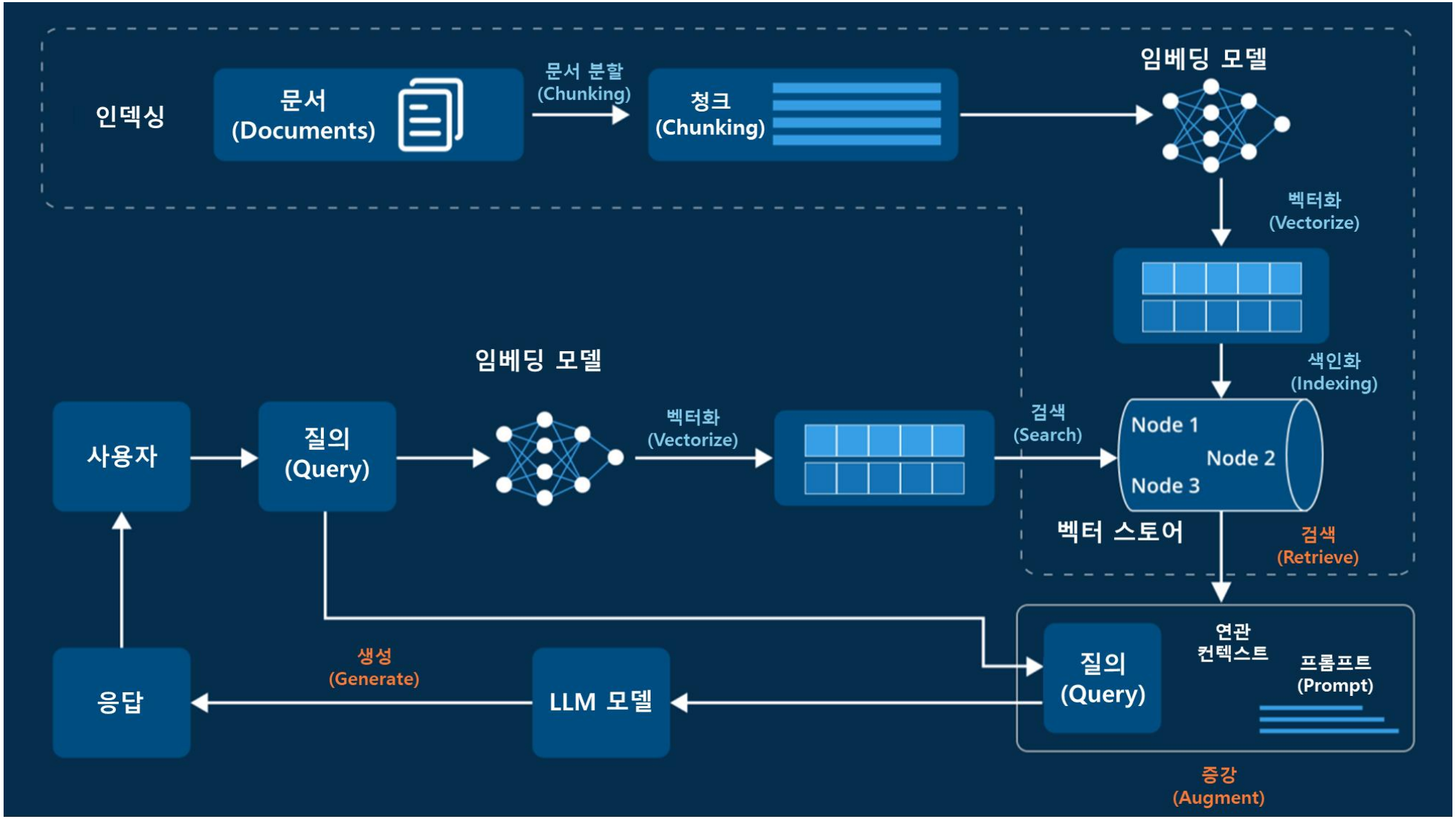
LangChain 실습

langchain_basic.ipynb

langchain_component.ipynb

langchain_rag.ipynb

Advanced RAG



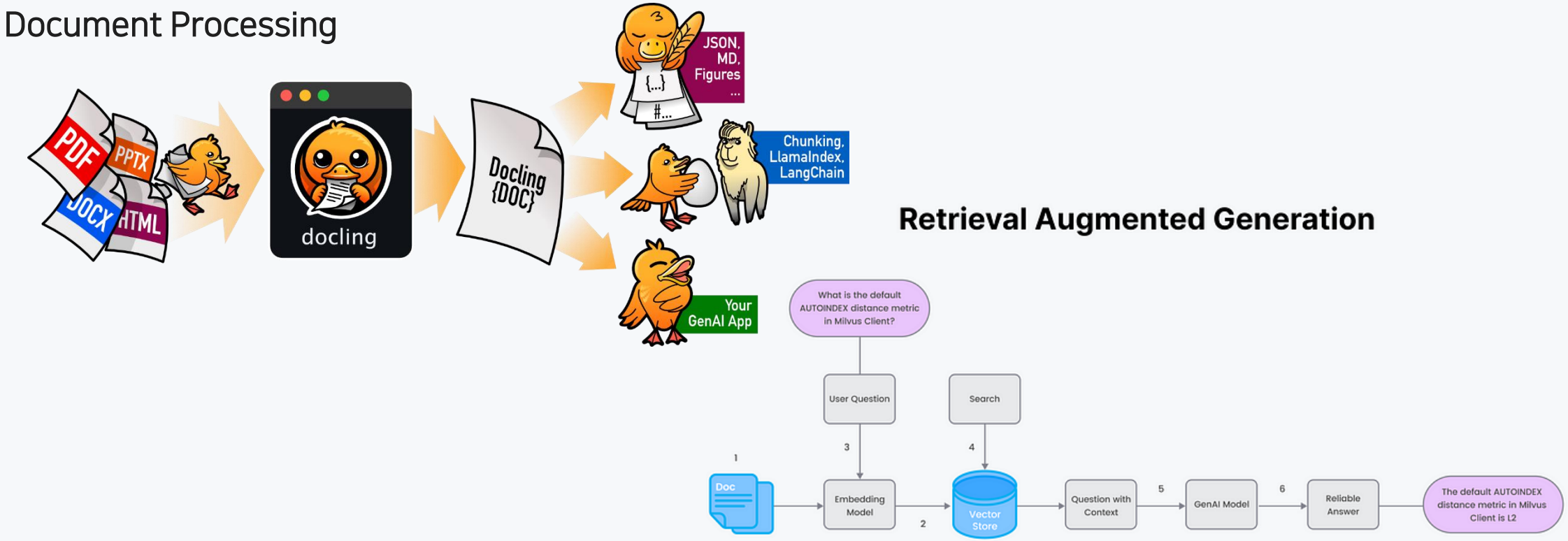
https://www.leewayhertz.com/advanced-rag/

RAG 파이프라인 구축 실습

langchain_rag_docling_milvus

오픈소스 기술스택

- Docling : https://github.com/docling-project/docling
- LangChain : https://github.com/langchain-ai/langchain
- Milvus : https://github.com/milvus-io/milvus
- HuggingFace : https://huggingface.co/
- Ollama : https://github.com/ollama/ollama
- Langfuse : https://github.com/langfuse/langfuse



Thank you