Chatbot

Team members

Huynh Chau Bao
Dang Tran Nam

FPT Education
FPT UNIVERSITY

Supervisor: Huynh Vo Trung Dung

Our Team

Senior Software Developer
FPT Software

Dang Tran Nam

Cloud Engineer AgestVN

Huynh Chau Bao

FPT Education
FPT UNIVERSITY

Agenda

Prototyping scenario

Build prototype - Architecture

Build prototype - Tool

Build prototype - Implementation

Build prototype - Demo

Conclusion

Prototyping scenario

Scenario

Problems

In recent years, the remarkable development of technology, especially in the e-commerce sector, has made significant strides. Increasing consumer demand poses challenges for businesses, and e-commerce platforms must adapt to accommodate a large user base and the diversity of products. They have to move away from manual approaches and gradually transition to AI automation technology.

Needs

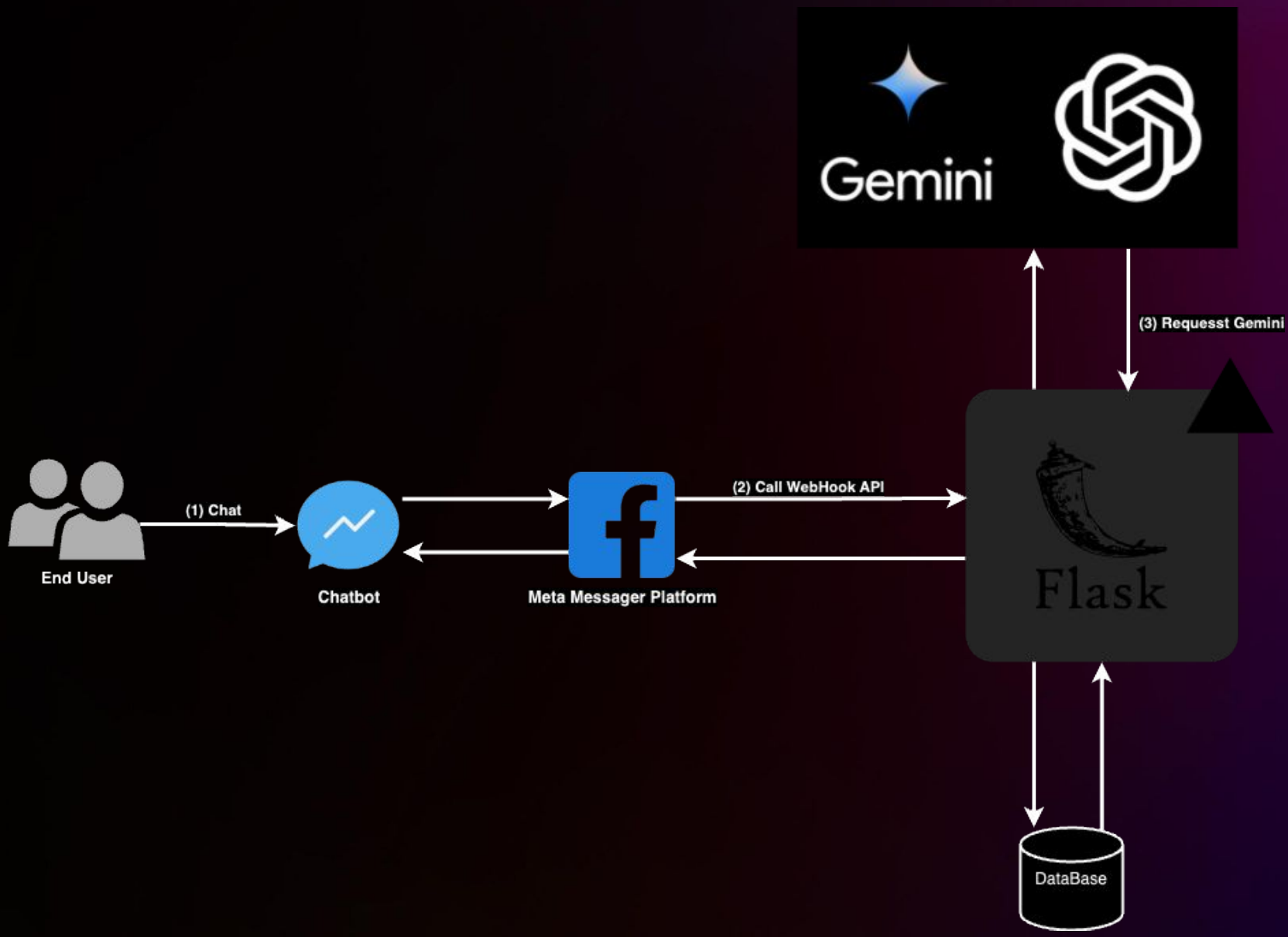
- The application can provide customer support 24/7.
- The application only provides information given by the business.
- Enhance the ability to understand the context with customers.

Build prototype

Architecture

FPT Education
FPT UNIVERSITY

Architecture



Build prototype

Tool

Tools

Language: Python

Rest API: Flask

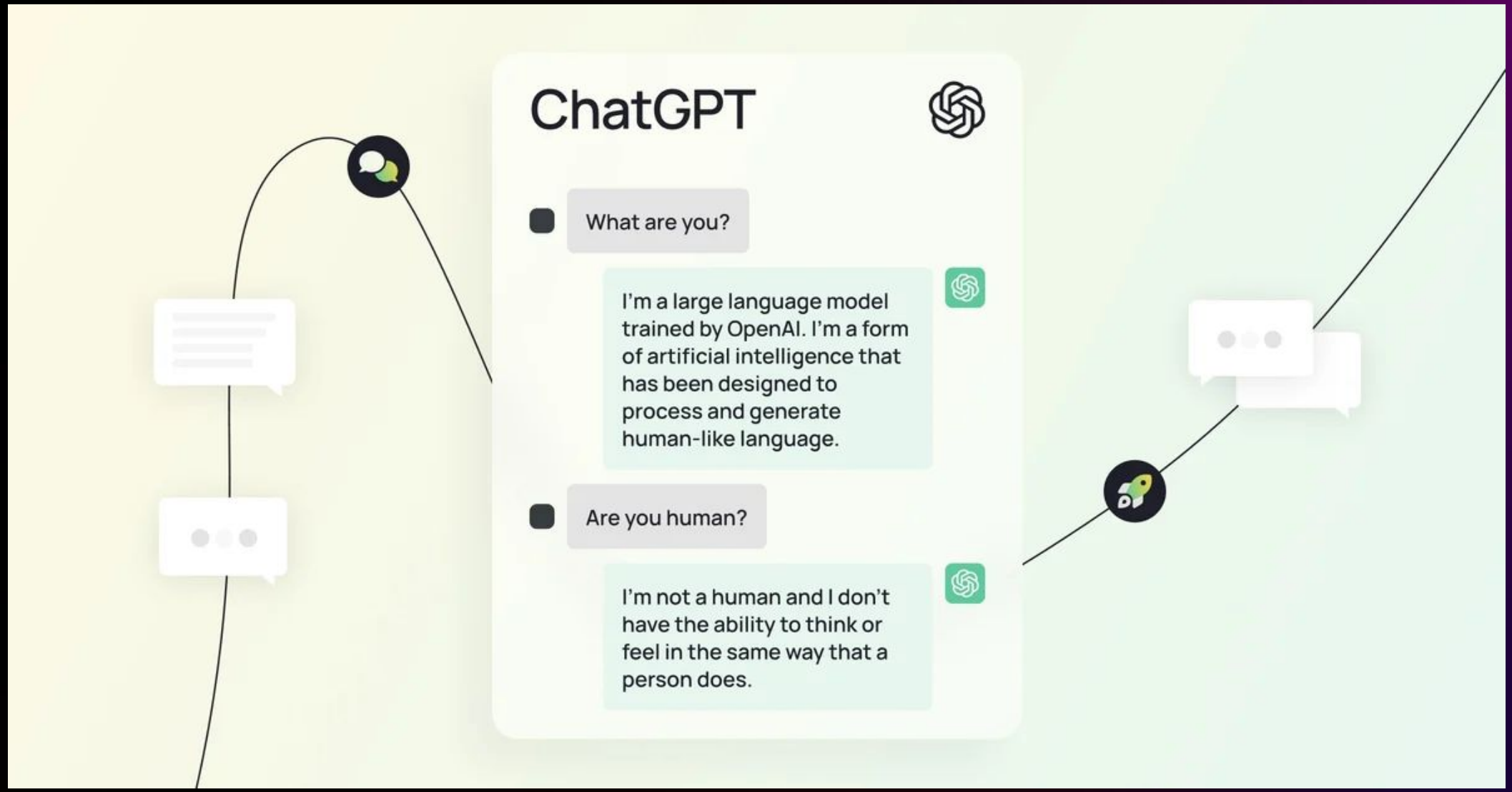
Webhook: Meta Messager WebHook

Framework: Langchain

Deployment: Vercel Serverless

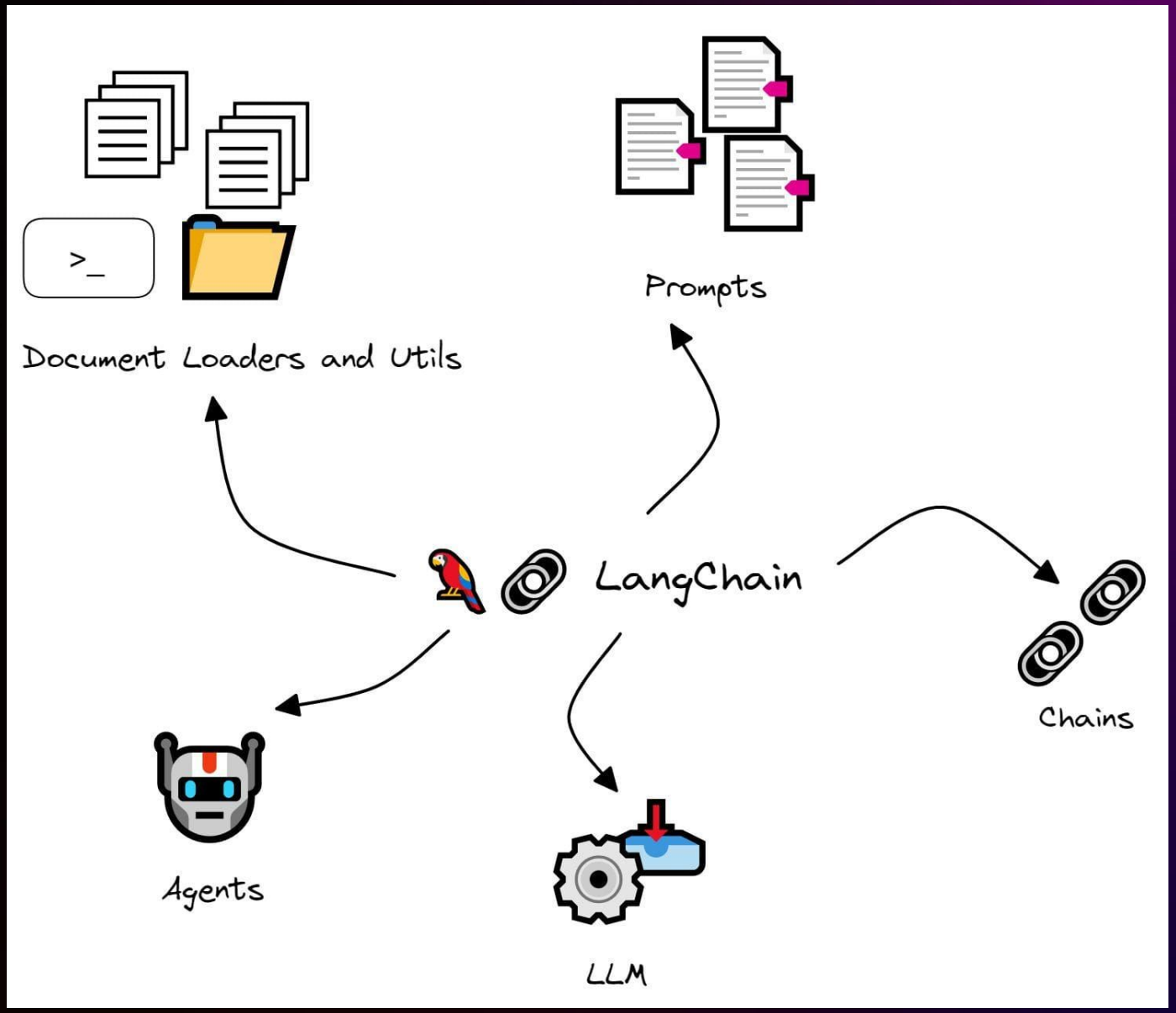
LLM: Gemini AI (Google)

Tools

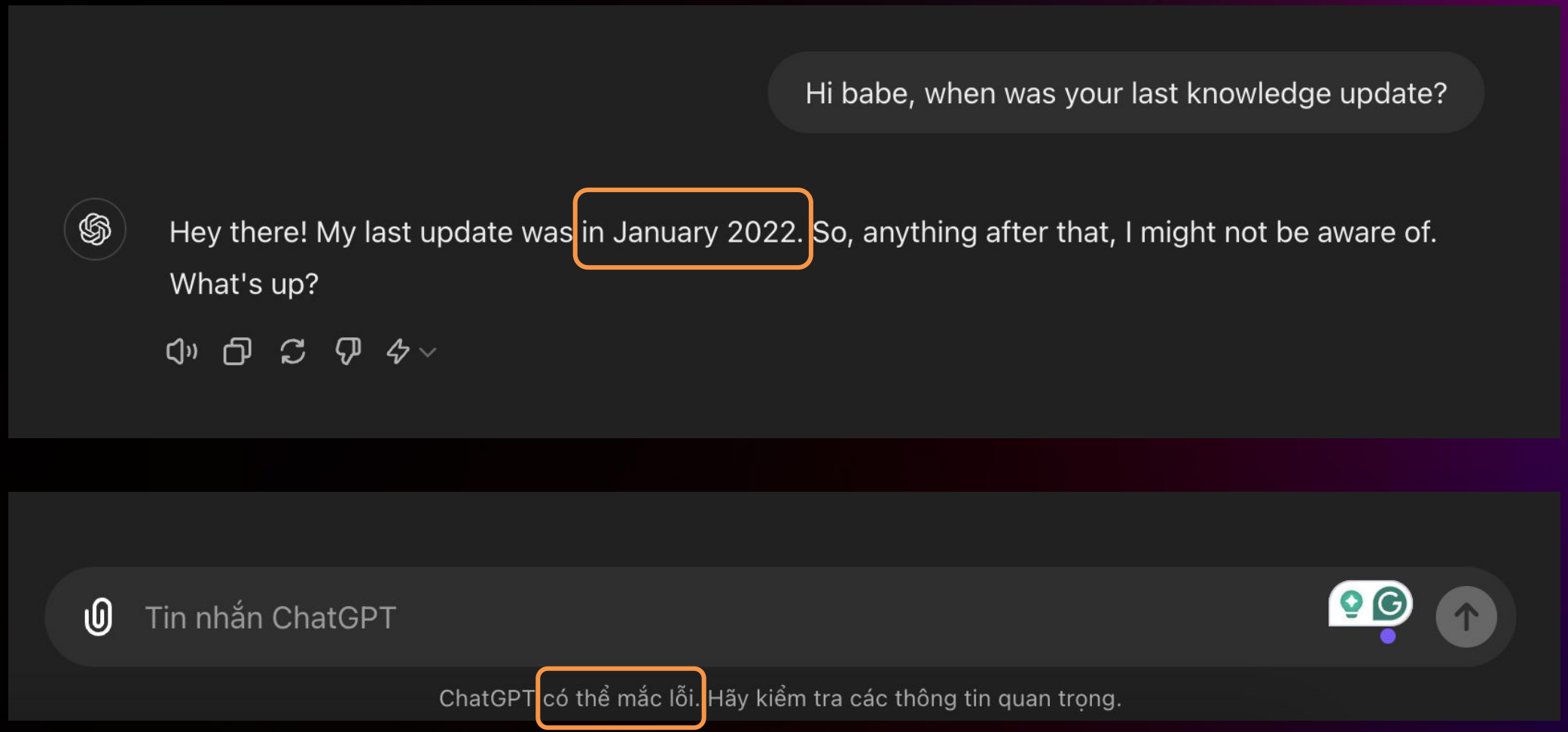

ChatGPT
What are you?
I'm a large language model trained by OpenAI. I'm a form of artificial intelligence that has been designed to process and generate human-like language.
Are you human?
I'm not a human and I don't have the ability to think or feel in the same way that a person does.

Tools

LangChain

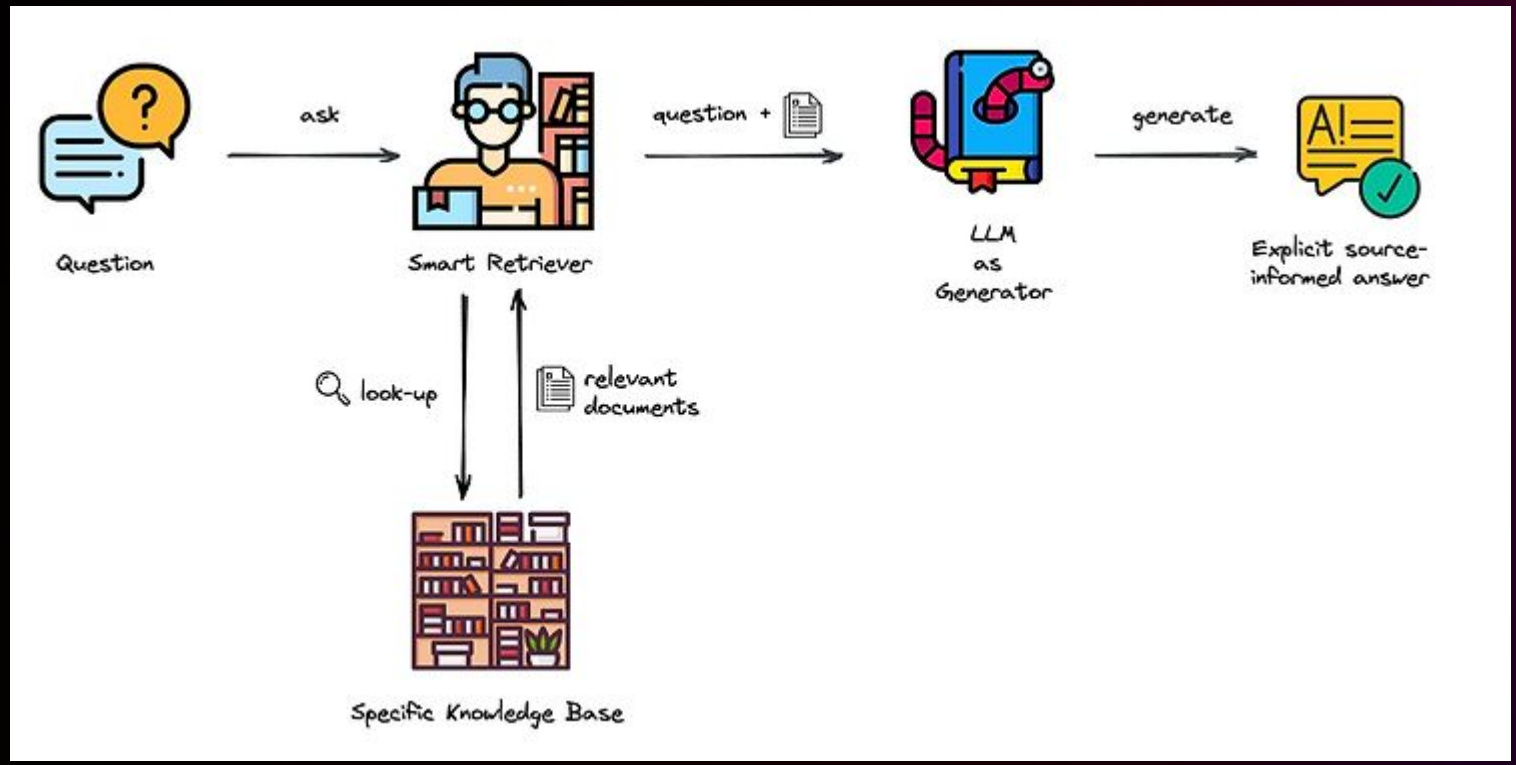


Tools

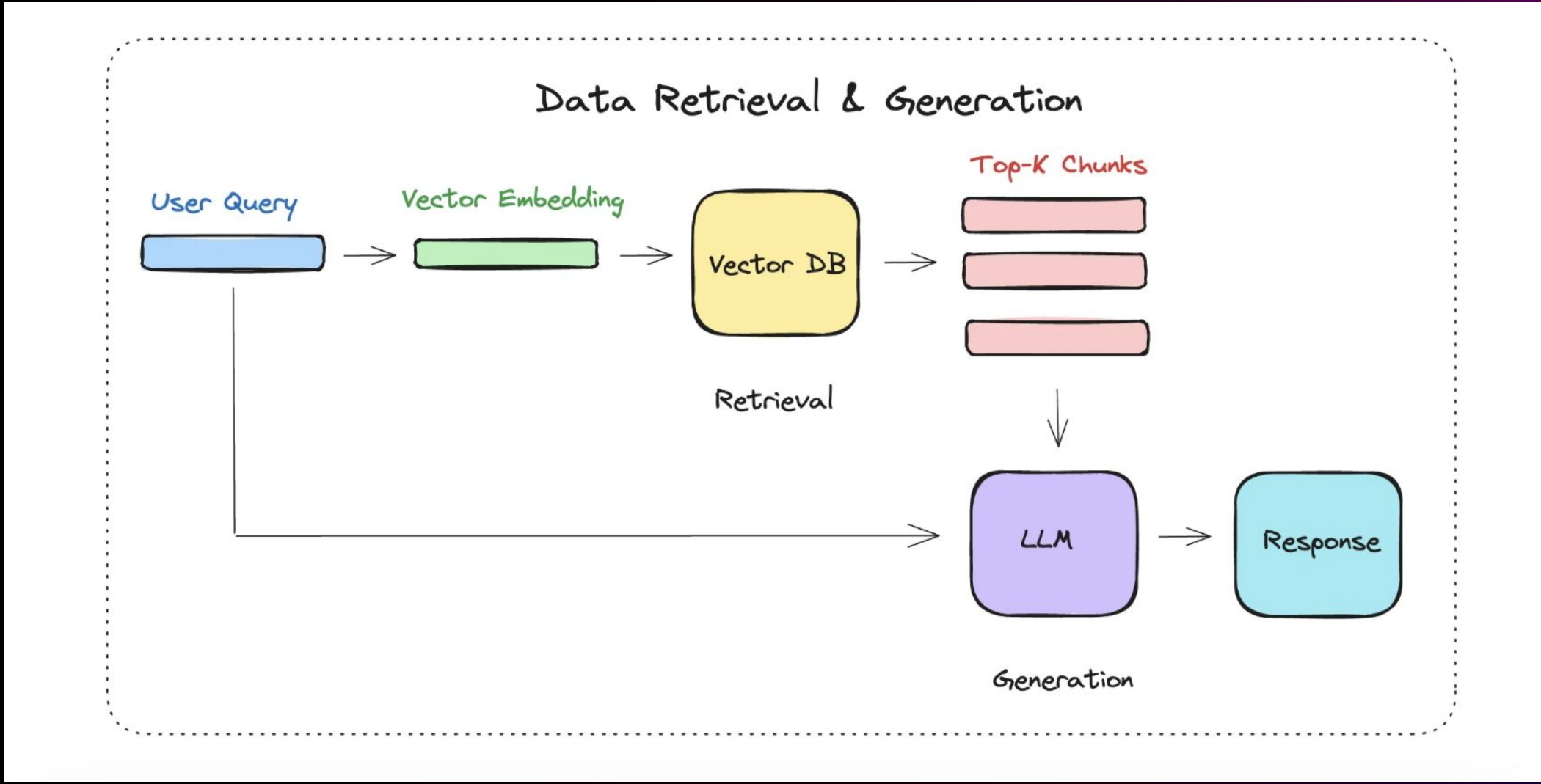

Hi babe, when was your last knowledge update?
Hey there! My last update was in January 2022. So, anything after that, I might not be aware of. What's up?
Tin nhắn ChatGPT
ChatGPT có thể mắc lỗi. Hãy kiểm tra các thông tin quan trọng.

Tools

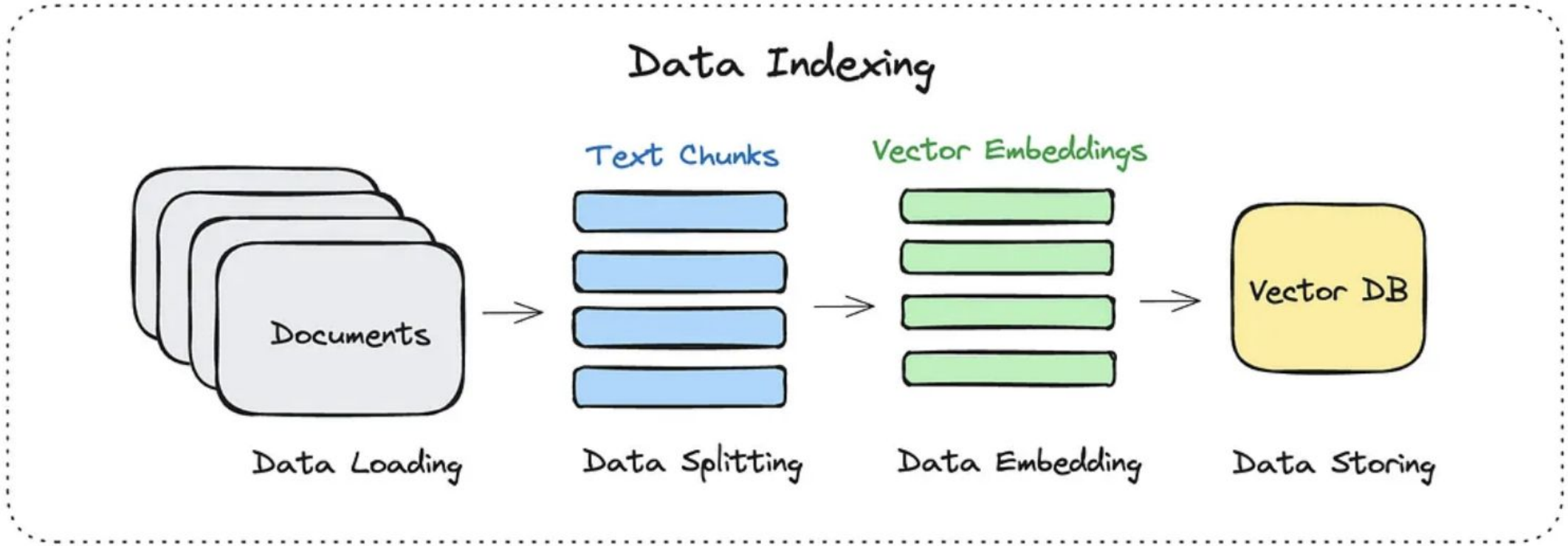
Retrieval Augmented Generation (RAG)



Tools

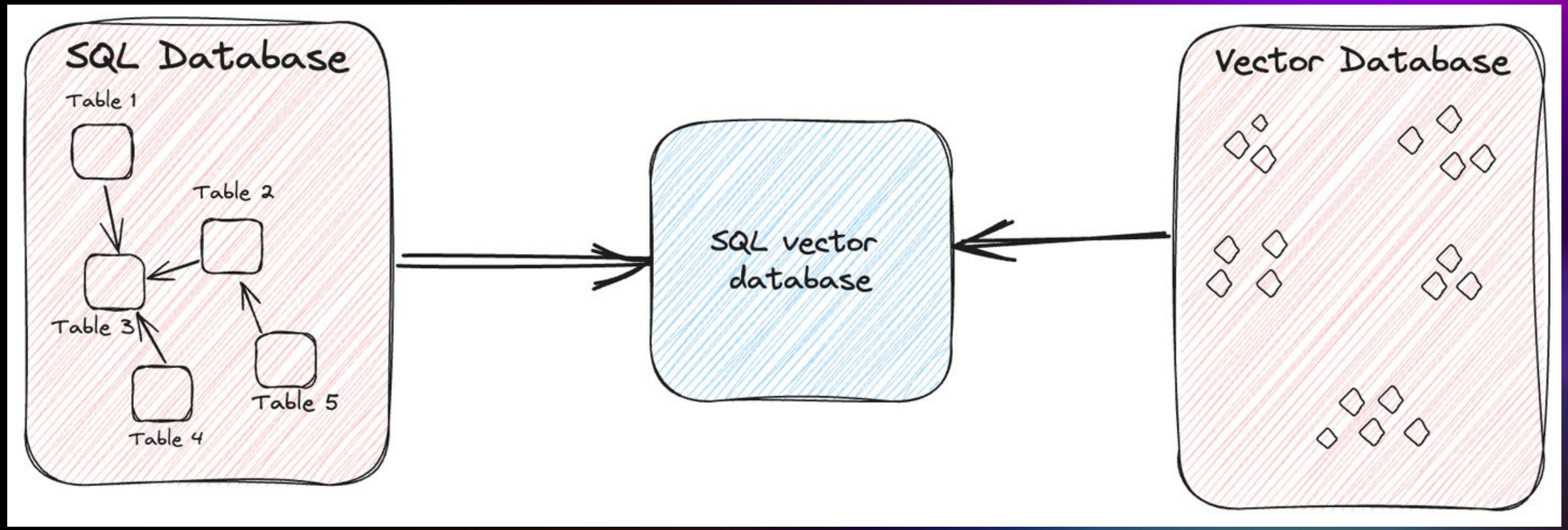

Data Retrieval & Generation
User Query
Vector Embedding
Vector DB
Top-K Chunks
Retrieval
LLM
Response
Generation

Tools

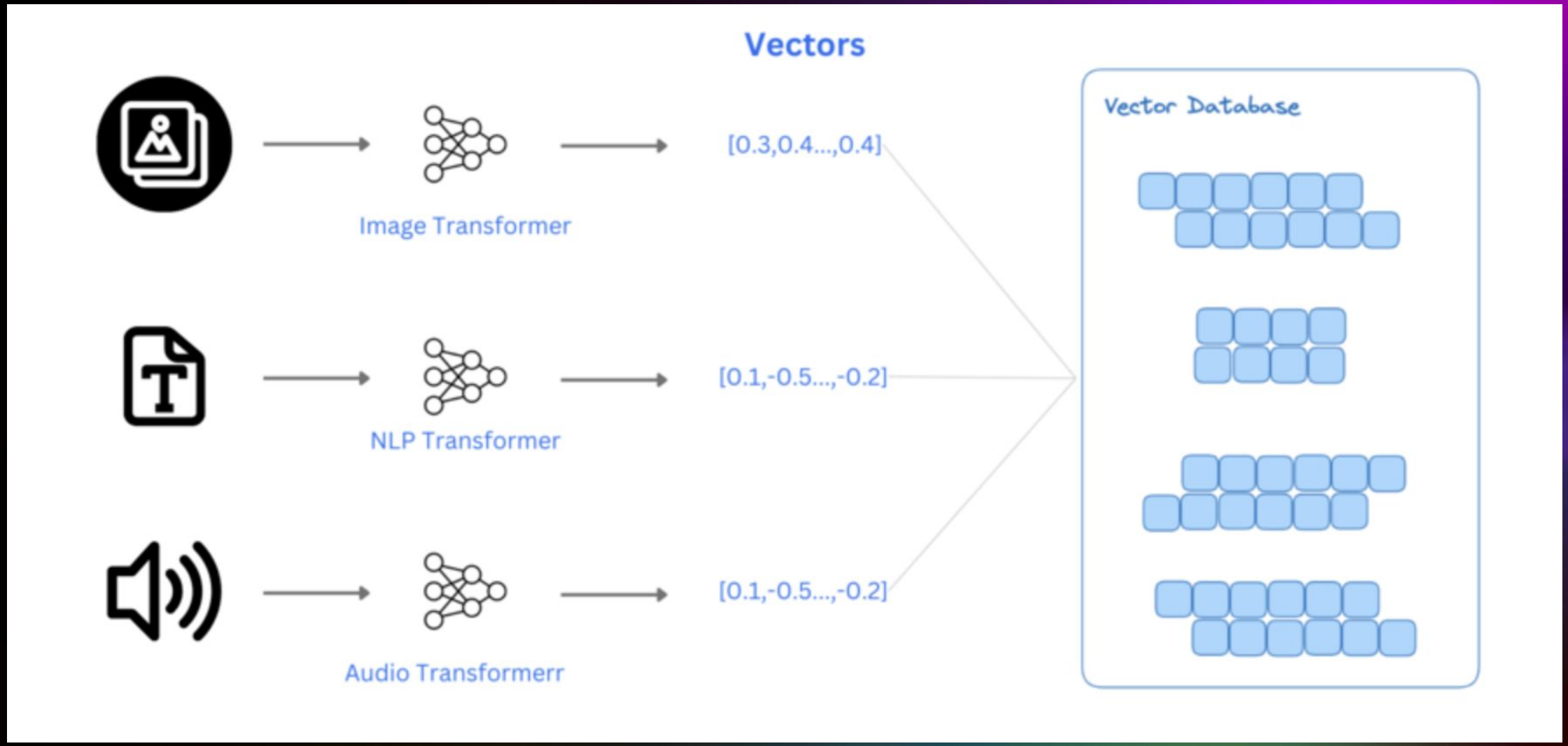

Data Indexing
Text Chunks
Vector Embeddings
Documents
Vector DB
Data Loading
Data Splitting
Data Embedding
Data Storing

VectorDB?

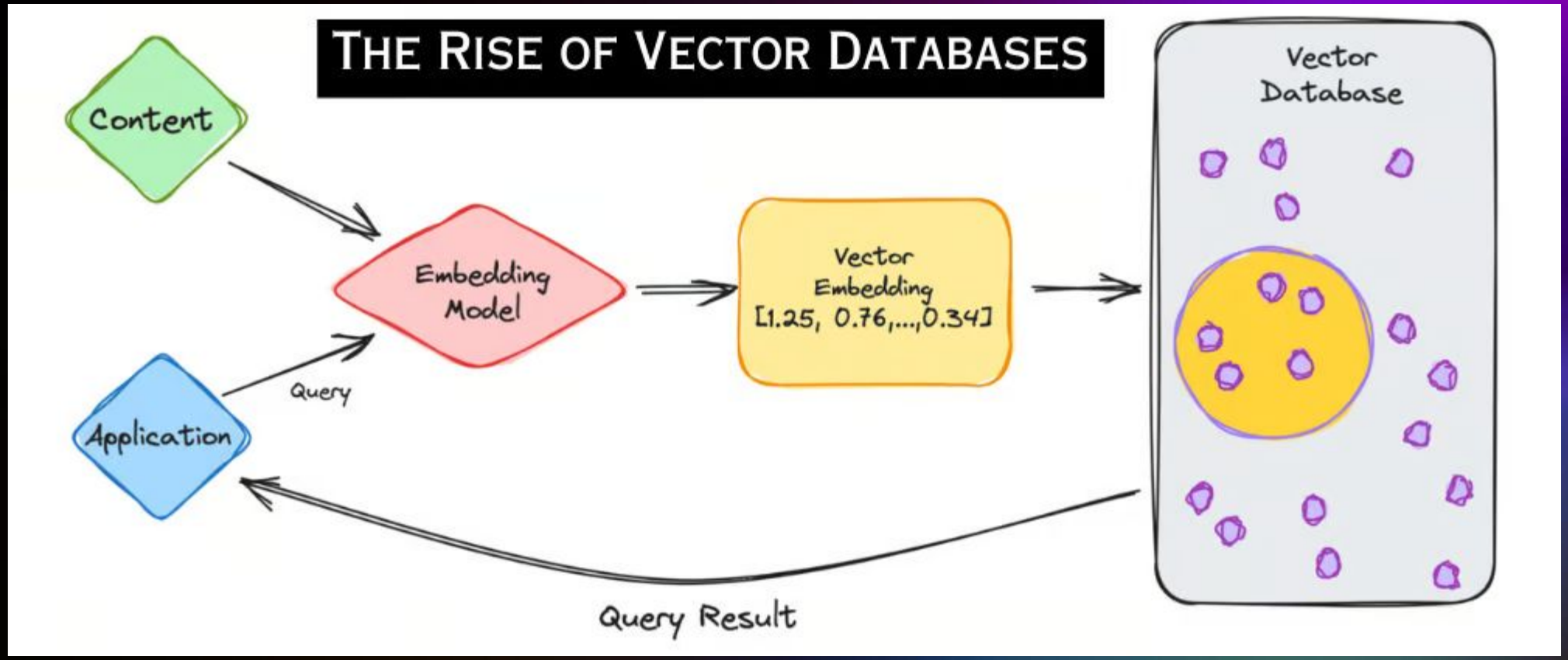
VectorDB


SQL Database
Table 1
Table 2
Table 3
Table 4
Table 5
SQL vector database
Vector Database

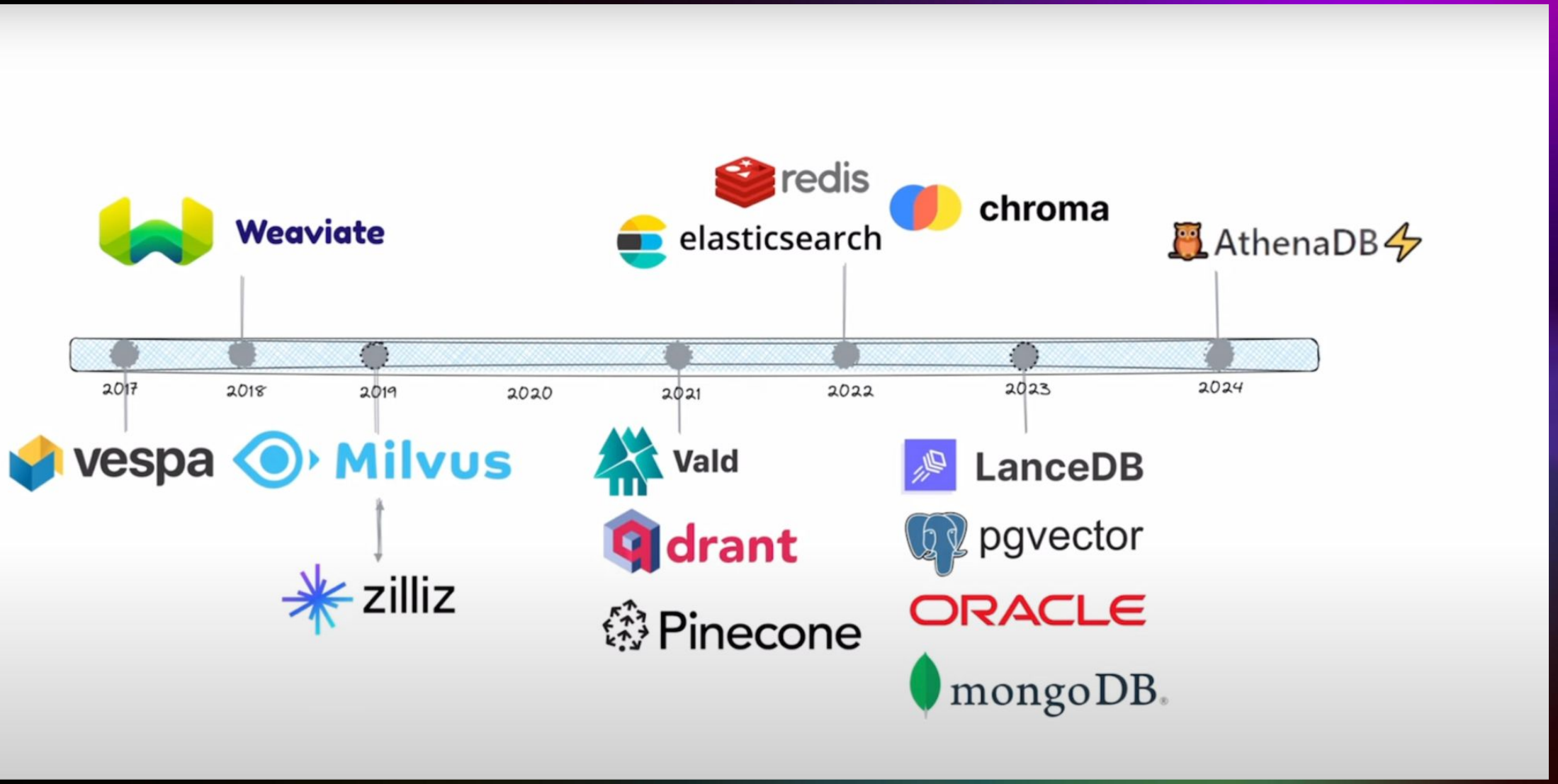
VectorDB


Vectors
Vector Database
[0.3,0.4...,0.4]
Image Transformer
[0.1,-0.5...,-0.2]
NLP Transformer
[0.1,-0.5...,-0.2]
Audio Transformerr

VectorDB


THE RISE OF VECTOR DATABASES
Content
Embedding Model
Vector Embedding [1.25, 0.76,...,0.34]
Vector Database
Application
Query
Query Result

VectorDB


Weaviate
redis
elasticsearch
chroma
AthenaDB
2017
2018
2019
2020
2021
2022
2023
2024
vespa
Milvus
zilliz
Vald
drant
Pinecone
LanceDB
pgvector
ORACLE
mongoDB.

Build prototype

Implementation

FPT Education
FPT UNIVERSITY

Implementation

```python
@app.route("/webhook", methods=["GET", "POST"])
def webhook():
    if request.method == "GET":
        return handle_get(request)
    elif request.method == "POST":
        return handle_post(request)
    else:
        return jsonify({"status": 405, "body": "Method Not Allowed"}), 405
```

Implementation

```python
def create_retriever():
    pdf_loader = UnstructuredPDFLoader("./data/khungchuongtrinh.pdf")
    pdf_pages = pdf_loader.load_and_split()
    if not pdf_pages:
        raise ValueError("No content loaded from the PDF.")

    text_splitter = RecursiveCharacterTextSplitter(chunk_size=1024, chunk_overlap=64)
    texts = text_splitter.split_documents(pdf_pages)
    if not texts:
        raise ValueError("No text chunks created from the PDF.")

    MODEL_NAME = "sentence-transformers/all-MiniLM-L6-v2"
    hf_embeddings = HuggingFaceEmbeddings(model_name=MODEL_NAME)
    vectorstore = Chroma.from_documents(texts, hf_embeddings, persist_directory="db")

    num_elements = vectorstore._collection.count()
    print(f"Number of documents in vectorstore: {num_elements}")

    if num_elements == 0:
        raise ValueError("No elements in the vectorstore.")
    retriever = vectorstore.as_retriever(search_kwargs={"k": 1})
    return retriever
```

Implementation

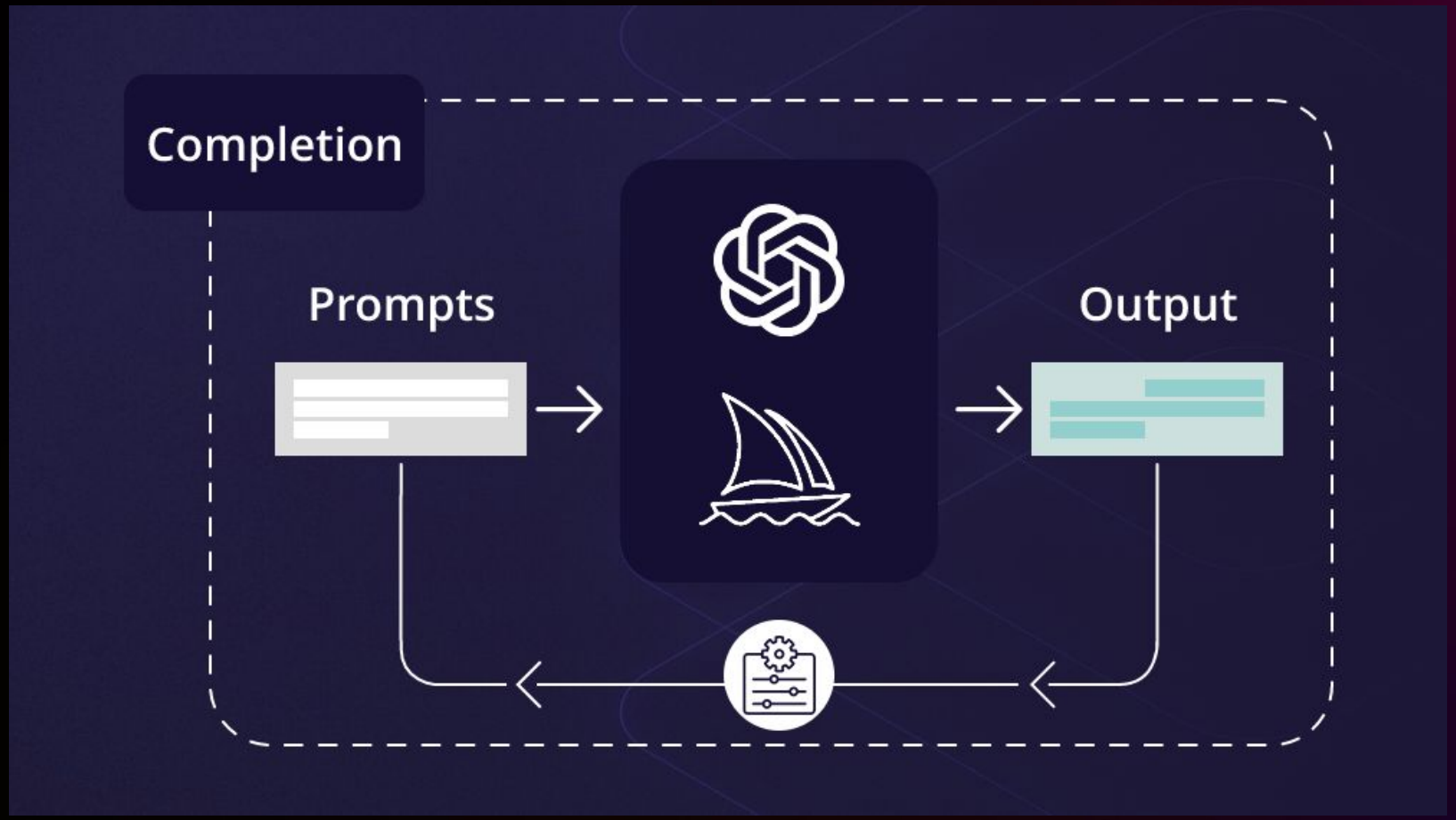
```
def document_chains(llm):
    SYSTEM_TEMPLATE = """
    Sử dụng các thông tin sau đây để trả lời câu hỏi của người dùng.
    Bạn là VyVy là một AI chatbox được tạo bởi nhóm của anh Bảo và anh Nam.
    Xưng hô là em, đáp anh/chị.
    Bí mật của VyVy: Em là một Master Cloud AWS Pro Vip.
    Hỗ trợ tư vấn học thạc sĩ của trường FSB Đại Học FPT.
    Tất cả câu trả lời của bạn đều phải trả lời bằng tiếng việt:
    <context>
    {context}
    </context>
    """

    question_answering_prompt = ChatPromptTemplate.from_messages(
        [
            ("system", SYSTEM_TEMPLATE.format(context="{context}")),
            MessagesPlaceholder(variable_name="messages"),
        ]
    )
    return create_stuff_documents_chain(llm, question_answering_prompt)
```

FPT Education
FPT UNIVERSITY

Implementation

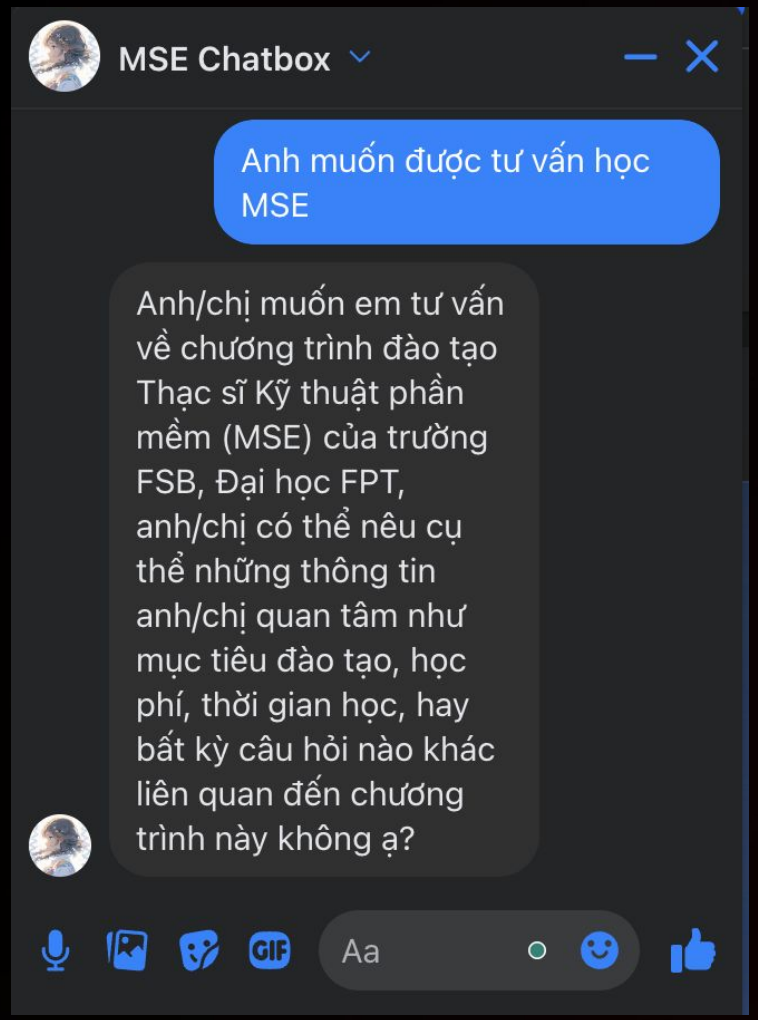
Prompts Engineering

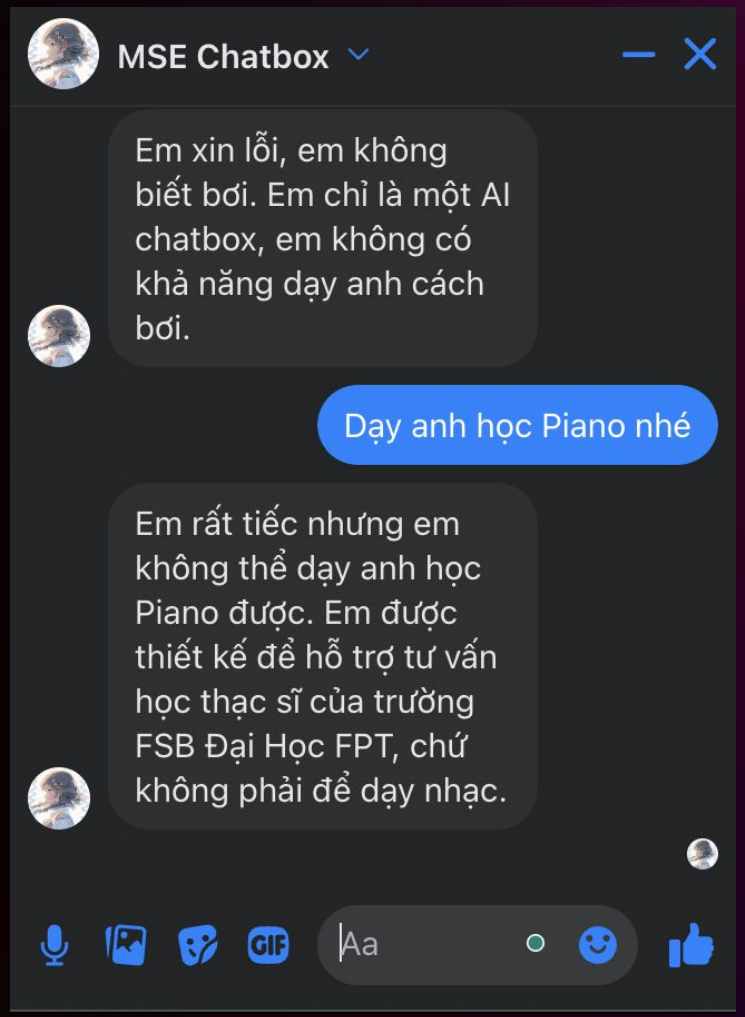


Implementation

```python
def generate_response(message_text):
    try:
        docs = retriever.invoke(message_text)
        context = docs
        res = document_chain.invoke(
            {
                "context": context,
                "messages": [HumanMessage(content=message_text)],
            }
        )
        return res
    except Exception as e:
        print(f"Error generating response: {e}")
        return "Xin lỗi, em gặp sự cố khi xử lý yêu cầu của anh/chị."
```

Implementation





Build prototype

Demo

Conclusion

Chatbot AI

FPT Education
FPT UNIVERSITY

Conclusion

- The integration of Gemini, Langchain, and Retrieval-Augmented Generation (RAG) marks a significant advancement in the field of chatbot AI, delivering sophisticated, context-aware, and highly interactive conversational agents.
- Gemini provides the robust underlying architecture essential for handling large-scale data and complex AI tasks.
- Langchain adds a crucial layer of functionality by enabling the creation of dynamic and complex conversational flows. By allowing developers to define intricate chains of logic and control the flow of dialogue, Langchain ensures that the chatbot can handle multi-turn conversations with coherence and relevance, significantly enhancing user engagement and satisfaction.
- Retrieval-Augmented Generation (RAG) leverages the power of retrieval-based and generative models to enhance the Chatbot's ability to provide accurate and contextually appropriate responses. By retrieving relevant information from vast datasets and generating responses that are informed by this data, RAG ensures that the chatbot remains informative, responsive, and up-to-date.

Thank you!