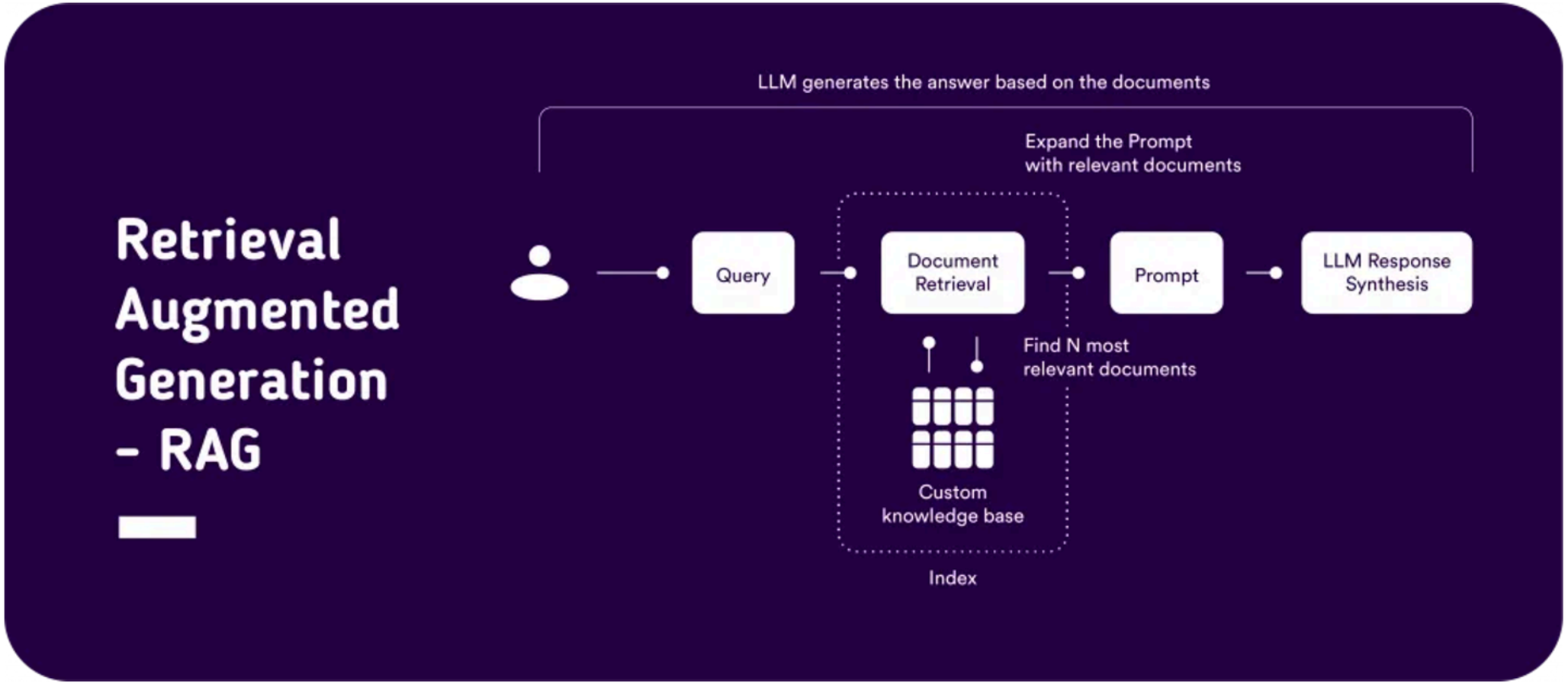
HOW TO BUILD A CHATBOT

Session 4 - RAG Chains in LangChain

GENERAL RAG PROCESS

Goal: Implement the process.



SESSION 4 AGENDA

1. Retrieval Chains
2. Q/A Retrieval Chains

RETRIEVAL CHAINS

1. **Define the Prompt:** Create a prompt template to summarize the main themes from retrieved documents.
2. **Format Documents:** Use format_docs(docs) to concatenate document content and prepare it for the model.
3. **Set Up the Chain:** Combine the document formatter, prompt, model, and output parser into a processing chain.
4. **Retrieve and Invoke:** Retrieve relevant documents using similarity_search, then pass them into the chain to generate a summary.

```python
from langchain_core.output_parsers import StrOutputParser
from langchain_core.prompts import ChatPromptTemplate

prompt = ChatPromptTemplate.from_template(
    "Summarize the main themes in these retrieved docs: {docs}"
)


# Convert loaded documents into strings by concatenating their content
# and ignoring metadata
def format_docs(docs):
    return "\n\n".join(doc.page_content for doc in docs)


chain = {"docs": format_docs} | prompt | model | StrOutputParser()

question = "What are the approaches to Task Decomposition?"

docs = vectorstore.similarity_search(question)

chain.invoke(docs)
```

1

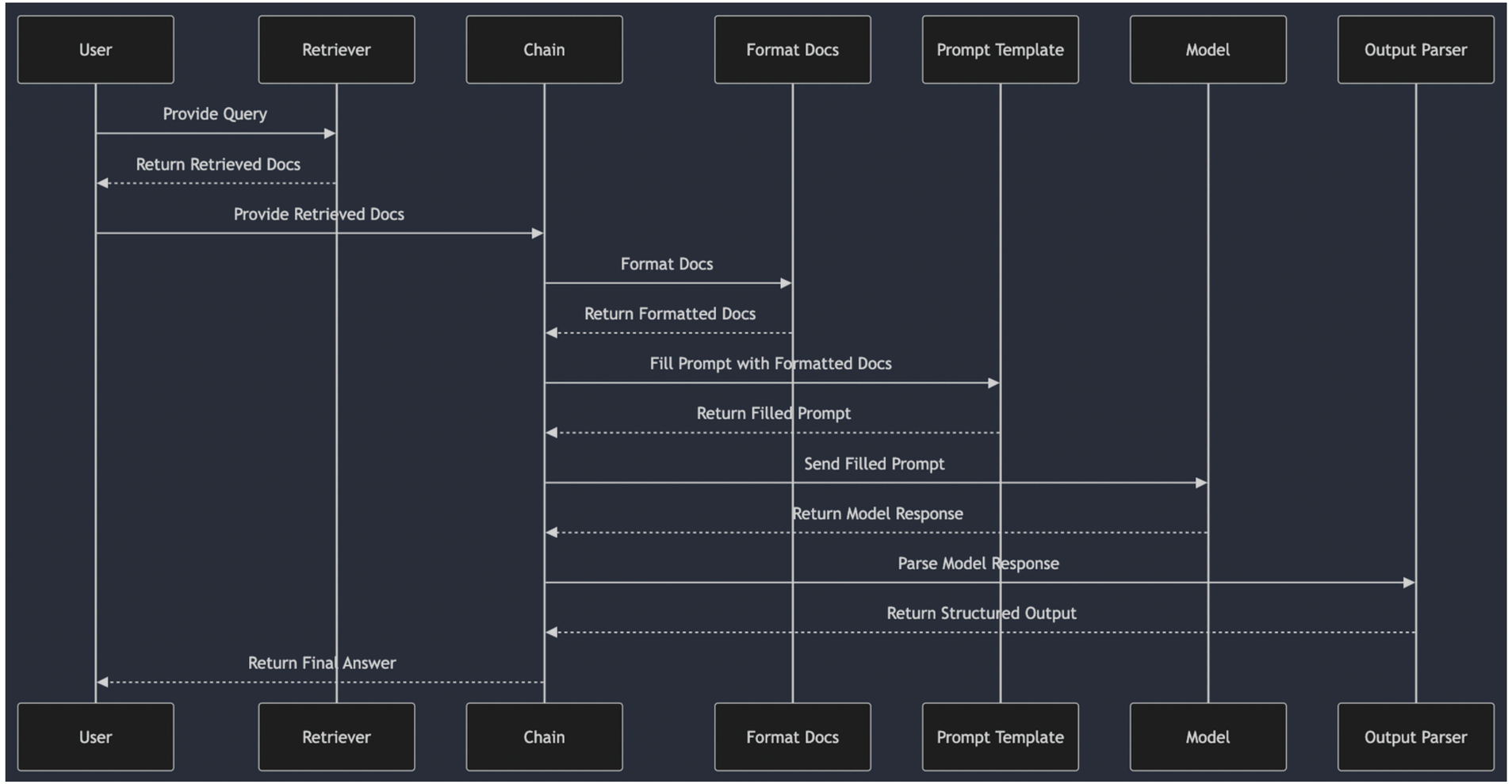
2

3

4

RETRIEVAL CHAIN

Behind the scenes of the chain.



Q/A RETRIEVAL CHAIN

1. **Define the Prompt:** Set up a template for answering questions based on retrieved documents.
2. **Create the Retriever:** Use `vectorstore` to retrieve relevant documents.
3. **Set Up the Chain:** Combine retriever, document formatter, query placeholder, prompt, model, and output parser.
4. **Invoke the Chain:** Pass the query into the chain to retrieve documents and generate a concise answer.

```python
from langchain_core.runnables import RunnablePassthrough

RAG_TEMPLATE = """
You are an assistant for question-answering tasks. Use the following pieces of retrieved

<context>
{context}
</context>

Answer the following question:

{question}"""

rag_prompt = ChatPromptTemplate.from_template(RAG_TEMPLATE)
```

1

```python
retriever = vectorstore.as_retriever()

qa_chain = (
    {"context": retriever | format_docs, "question": RunnablePassthrough()}
    | rag_prompt
    | model
    | StrOutputParser()
)
```

2

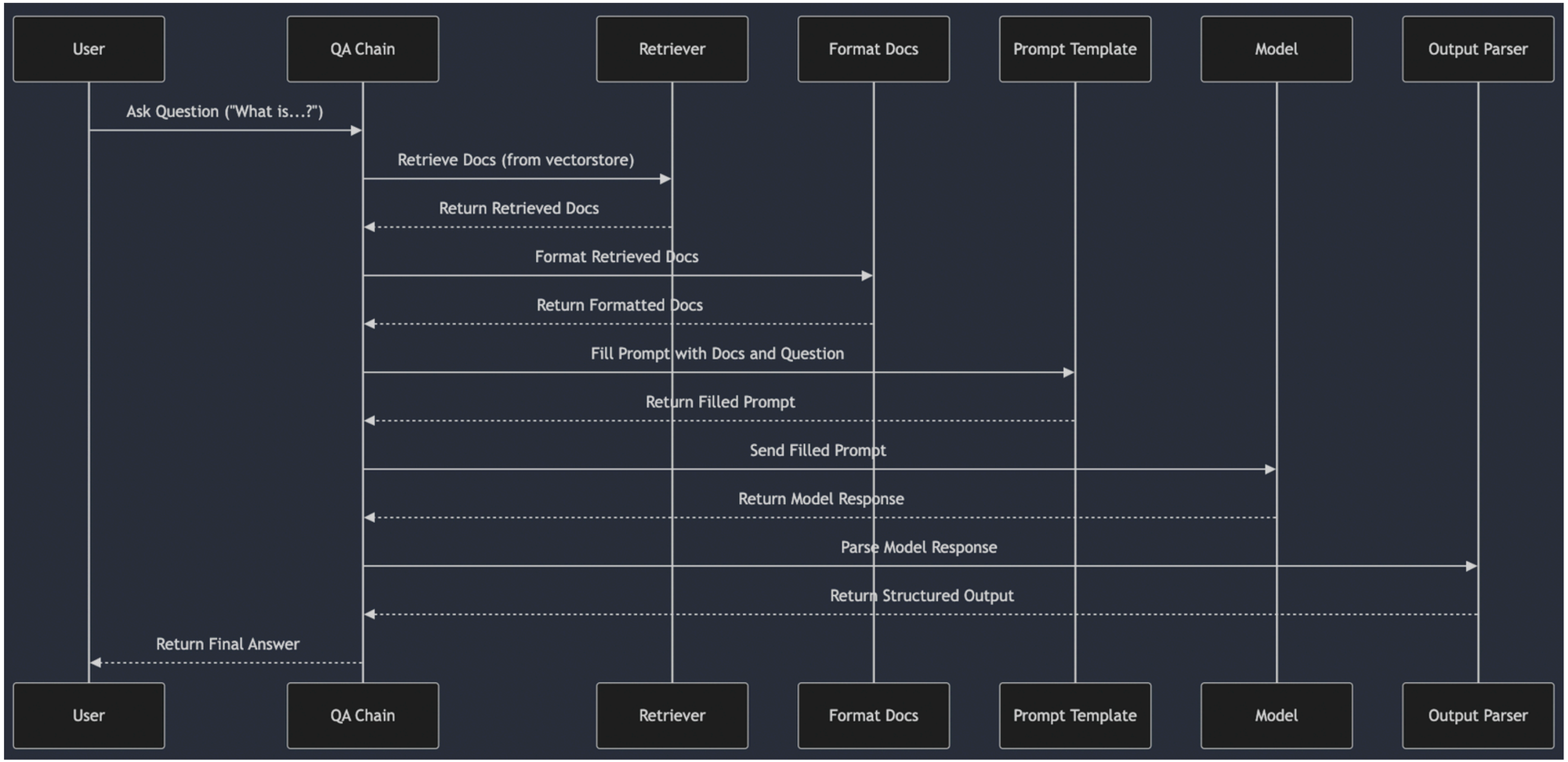
3

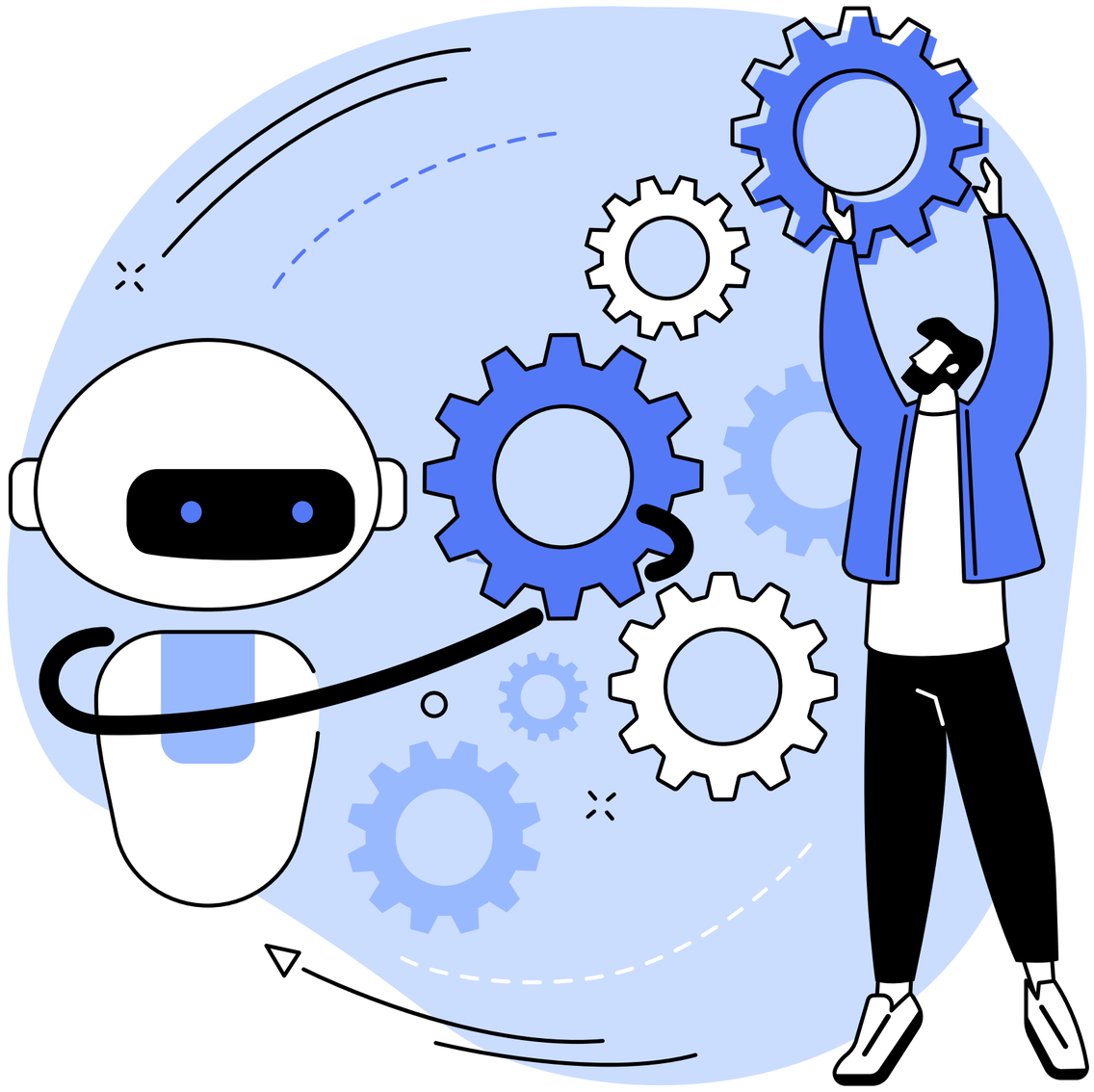
```python
question = "What are the approaches to Task Decomposition?"

qa_chain.invoke(question)
```

4

Q/A RETRIEVAL CHAIN

Behind the scenes of the chain.





IT’S YOUR TURN