

Prompt Engineering for RAG



Prompt engineering in RAG involves crafting structured templates that combine user queries with retrieved context. Well-designed prompts help LLMs generate accurate, grounded, and context-aware responses based on relevant documents.

01

Prompt Engineering for RAG

Prompt Engineering in RAG refers to crafting effective prompts that guide the LLM to generate accurate, relevant, and grounded answers using retrieved context from a vector database.

In RAG, you're not just giving the model a question — you're giving it:

- A **question**
- Some **retrieved context** •
- And an **instruction** on how to use that context



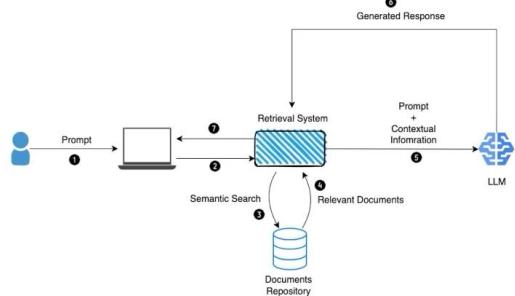
02

RAG Flow Recap

User Query → Retriever → Context → 🎨 Prompt → LLM → Answer

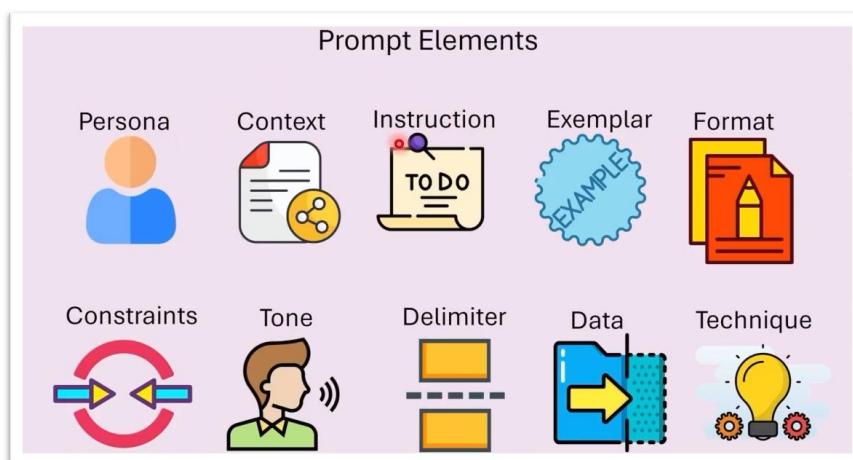
Prompt Engineering influences:

- ❑ How the model interprets the query
- ❑ How it uses the retrieved chunks
- ❑ How confident, concise, and factual the response is



03

Prompt Template Components



04

Example Prompt Template

You are an AI assistant. Use the following context to answer the question
accurately and concisely.

Context:
{retrieved_context}

Question:
{user_query}

Answer in a helpful tone. If the answer is not found in
the context, say "The information is not available."

Prompt & Prompt Template

Example code (e.g. Python)

```
prompt_template = """Act as support staff.  
Help the owners of the HCR3000 operate their cleaning  
robot by giving answers to questions on features and step-  
by-step instructions when they ask for help.
```

```
User: {query} Assistant:"""
```

```
# for each conversation turn  
prompt = prompt_template.format(query=actual_user_query)
```

1 prompt_template instead of prompt

2 Variable in the template.

3 Variable in the template is replaced by
current user query to get the prompt

05

Techniques in Prompt Engineering for RAG

Technique

Context Injection

Chain-of-Thought Prompting

Answer Formatting

Fallback Handling

Tone/Style Control

Goal

Insert relevant info before question

Ask the model to explain its reasoning

Force structured replies (bullets, JSON, tables)

Handle cases where context is insufficient

Adjust tone (e.g., formal, friendly, technical)

06

Best Practices

- Keep prompts short and consistent
- Include system role or persona if needed
- Limit context size to stay within token limits
- Escape hallucination by telling the model: "If unsure, say you don't know."
- Use few-shot prompting only if needed (more costly)