SAMIR GENAI ACADEMY

PROFESSIONAL STUDY GUIDE SERIES

Phase 02 · RAG & Knowledge Systems

07

RAG Architecture

Naive RAG -> Advanced RAG -> Modular RAG pipelines

Module 07 of 25 · **1 hour**

Instructor: **Samir Kumar** · Edition 2026

STUDY GUIDE

01 Overview

> Naive RAG -> Advanced RAG -> Modular RAG pipelines. This guide gives you the core theory, the tools that matter, a hands-on build, working code, and the pitfalls to avoid — everything you need to learn the topic to a professional standard and apply it in real projects.

1 hour	4 + 4	4	07/25
LESSON LENGTH	CONCEPTS + BUILD STEPS	FRAMEWORKS & TOOLS	PHASE 02

02 Learning Objectives

- Describe the retrieve-augment-generate loop end to end.
- Implement chunking, retrieval, and grounded generation.
- Add reranking and contextual compression to improve precision.
- Cite sources and reduce hallucination.

03 Core Concepts

Why RAG

RAG grounds a model in your private, current data without retraining. At query time you retrieve relevant chunks and inject them into the prompt, so answers stay factual, citeable, and up to date — solving the 'frozen knowledge' problem of base models.

The pipeline

Ingest -> chunk -> embed -> store. At query time: embed query -> retrieve top-k -> (rerank) -> build grounded prompt -> generate with citations. Each stage is independently tunable — this modularity is the heart of 'Modular RAG'.

Chunking strategy

Chunk size and overlap drive quality. Too large dilutes relevance; too small loses context. Prefer semantic / structure-aware chunking (by heading, paragraph) over fixed character counts, and keep source metadata for citations.

Reranking & compression

Initial vector retrieval favours recall; a cross-encoder reranker reorders for precision. Contextual compression then trims retrieved chunks to only the sentences that answer the query, saving tokens and sharpening the prompt.

04 Visual Diagram

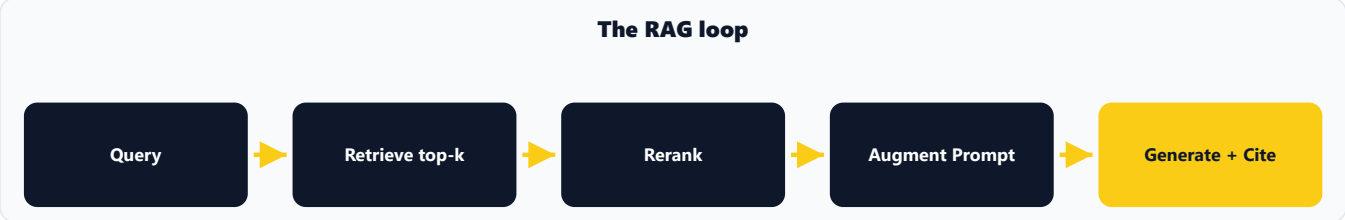

Figure 7 — Retrieve, rerank, augment, generate. Grounding the model in retrieved context keeps answers factual and citeable.

05 Key Frameworks & Tools

LangChain / LlamaIndex | Cohere / cross-encoder rerankers | Vector DB of choice | Unstructured (parsing)

06 Hands-On Build

🛠 Advanced Reranking & Contextual Compression system

1. Ingest a document set with structure-aware chunking + metadata.
2. Build the retrieve -> prompt -> generate loop with citations.
3. Add a cross-encoder reranker over the top-k results.
4. Add contextual compression and measure answer quality before/after.

07 Code Walkthrough

Minimal grounded-generation prompt — PYTHON

```python
def build_prompt(query, chunks):
    context = "\n\n".join(f"[{i+1}] {c['text']}" for i, c in enumerate(chunks))
    return (
        "Answer ONLY from the context. Cite sources as [n]. "
        "If the answer is not present, say you don't know.\n\n"
        f"Context:\n{context}\n\nQuestion: {query}"
    )
```

08 🧪 Lab Assistant

Lab 7 · Grounded Q&A with citations

Prerequisites:A vector index from Lab 6 · An LLM API key

1. **Retrieve top-k chunks for a question**

   > Expected: Relevant passages with source metadata

2. **Build the grounded prompt and generate**

   > Expected: An answer that cites sources as [n]

3. **Ask something not in the corpus**

   > Expected: The model says it doesn't know (no hallucination)

Troubleshooting

- Model invents facts → tighten the 'answer ONLY from context' instruction.
- Poor answers → improve chunking and add reranking (Module 7 build).

09 Pitfalls & Key Takeaways

⚠ COMMON PITFALLS

- Letting the model answer beyond the retrieved context (hallucination).
- Fixed-size chunking that splits tables or code mid-structure.
- Skipping reranking and trusting raw vector order.

✓ KEY TAKEAWAYS

- RAG grounds models in private, current data without retraining.
- Chunking quality and reranking dominate answer precision.
- Always cite sources and constrain answers to context.

📺 **Companion video:** Watch the module 07 lesson in your Student Dashboard, then use this guide to revise, copy the code, and complete the build. Download both for offline study.

10 Further Resources

LlamaIndex RAG guides • 'Retrieval-Augmented Generation' (Lewis et al.) • Cohere Rerank docs