

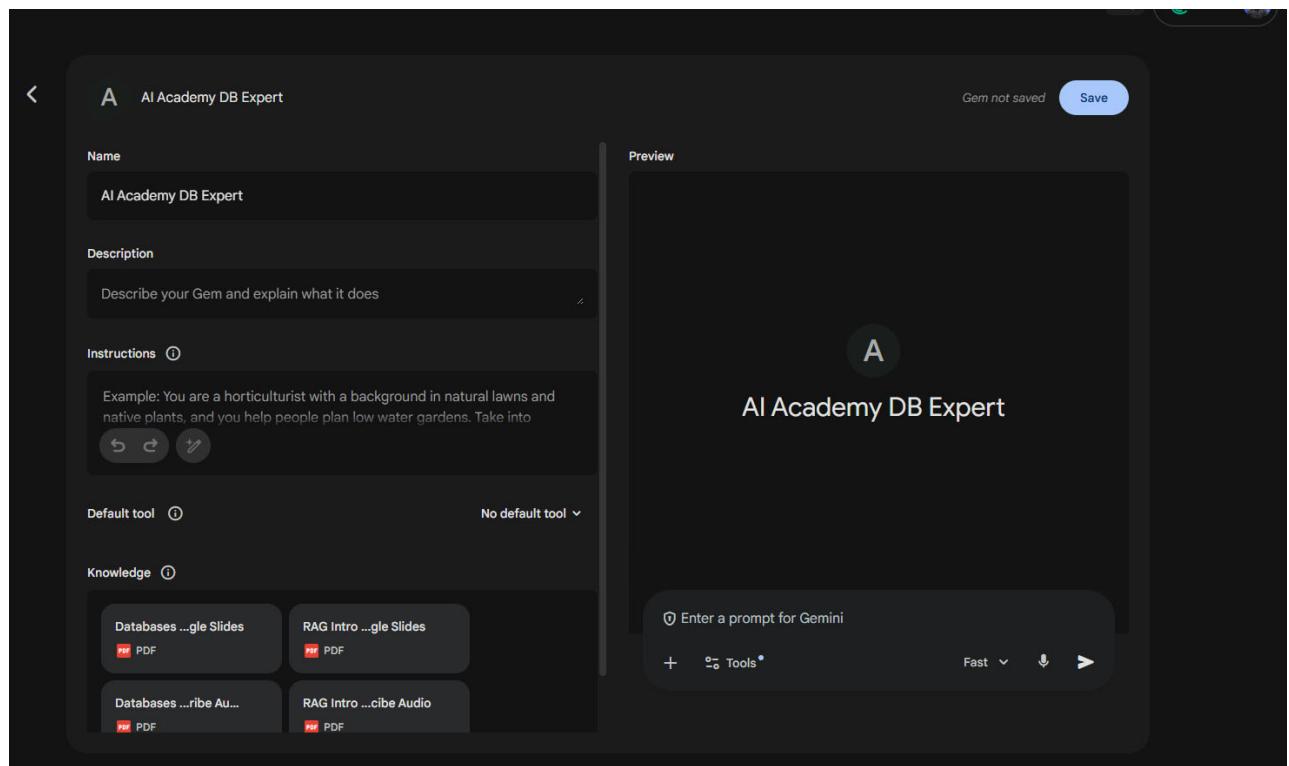
Module 4 Homework [AI Academy]

The Gem is designed to assist developers and data architects in understanding how to design, implement, and optimize Retrieval-Augmented Generation (RAG) systems and vector databases for generative AI applications.

It specializes in explaining RAG architecture, chunking strategies, and hybrid retrieval methods. Additionally, it provides guidance on embedding model selection, vector database performance optimization, and agentic RAG patterns. The Gem is instructed to reduce hallucinations by grounding every response strictly in the Databases for GenAI and RAG Intro materials.

After configuring the instructions and uploading the relevant documents, I began testing it by asking simple introductory questions to evaluate its behavior and accuracy.

➔ Creating My Gem



→ Started with simple question

1. What are the production Do's and Don'ts of building a RAG system?

The screenshot shows the Gemini AI interface. At the top, a search bar contains the query: "What are the production Do's and Don'ts of building a RAG system?". Below the search bar, a user profile is shown: "AI Academy DB Expert · Custom Gem". A dropdown menu labeled "Analysis" is open. The main content area displays a section titled "Production Best Practices" under "The Do's", listing several bullet points about hybrid search, metadata filtering, monitoring, evaluation frameworks, data freshness, and starting simple. At the bottom of this section is a prompt input field: "Enter a prompt for Gemini". To the right of the main content is a sidebar titled "Sources" containing a list of PDF documents related to RAG systems.

What are the production Do's and Don'ts of building a RAG system?

AI Academy DB Expert · Custom Gem

Analysis

Building a production-ready Retrieval-Augmented Generation (RAG) system requires moving beyond simple vector search to ensure accuracy, security, and scalability. Based on the provided materials, here are the essential Do's and Don'ts:

Production Best Practices

The Do's

- Use Hybrid Search: Establish a baseline by combining vector search with keyword-based search (BM25). This approach is often more effective than vector-only search. ⚡+3
- Implement Metadata Filtering: Use metadata (e.g., product, department, language, or permissions) to narrow down searches and ensure security. ⚡+2
- Monitor Retrieval Quality: Regularly track metrics such as `recall@k`, `precision`, and `NDCG` to evaluate how well your system is performing. ⚡+1
- Use Evaluation Frameworks: Move away from "vibes-based" testing and use systematic frameworks like TruLens, LangSmith, or DeepEval. ⚡+2
- Keep Data Fresh: Synchronize your embeddings with document changes to prevent stale or wrong answers. ⚡+3
- Start Simple: Begin with a basic setup (e.g., Chroma + LangChain) and iterate based on metrics rather than over-engineering from the start. ⚡+2

Enter a prompt for Gemini

+ Tools

Fast

Sources

Databases for GenAI - Google Slides.pdf
1. Use Hybrid Search (vector + BM25) as...

RAG Intro - Google Slides.pdf
Combine vector + keyword search (hybrid approach)...

Databases for GenAI - Transcribe Audio.pdf
...what do don't uh hybrid hybrid search is the best. Uh...

RAG Intro - Transcribe Audio.pdf
...exact numbers. Uh hybrid search is quite often better than like only vector only...

Databases for GenAI - Google Slides.pdf
2. Implement metadata filtering...

Databases for GenAI - Google Slides.pdf
...filter = ("department": "finance", ...)

RAG Intro - Google Slides.pdf

The screenshot shows the Gemini AI interface. At the top, a search bar contains the query: "What are the production Do's and Don'ts of building a RAG system?". Below the search bar, a user profile is shown: "AI Academy DB Expert · Custom Gem". A dropdown menu labeled "Sources" is open. The main content area displays a section titled "Standard Configuration (80% of Cases)", which recommends default settings for a "golden middle": chunk size of 512 tokens, chunk overlap of 15%, top-k retrieval of 3 to 5 chunks, and embedding dimensions of 768 to 1536. Below this, a question asks if the user wants to compare specific vector databases like PostgreSQL versus Qdrant. At the bottom of this section is a prompt input field: "Enter a prompt for Gemini". To the right of the main content is a sidebar titled "Sources" containing a list of PDF documents related to RAG systems.

What are the production Do's and Don'ts of building a RAG system?

AI Academy DB Expert · Custom Gem

Sources

Rely Solely on Vector Search: Vector-only search can miss exact matches or acronyms; always aim for a hybrid approach. ⚡+2

Ignore Access Controls: Failing to implement strict security and access controls can lead to data leakage and legal issues. ⚡+2

Overload the Context Window: Keep the information sent to the LLM manageable; it is recommended to keep the context window usage below 50% to avoid performance degradation. ⚡+3

Skip Systematic Evaluation: Never rely on a few "good-looking" answers; evaluate the system using domain-specific questions and established metrics. ⚡+2

Neglect Data Freshness: Stale data leads to hallucinations and incorrect responses. ⚡+1

Standard Configuration (80% of Cases)

For those starting out, the following default settings are recommended as a "golden middle": ⚡+1

- Chunk Size: 512 tokens. ⚡+1
- Chunk Overlap: 15% (~75 tokens). ⚡+2
- Top-k Retrieval: 3 to 5 chunks. ⚡+1
- Embedding Dimensions: 768 to 1536. ⚡+1

Would you like me to compare specific vector databases, such as PostgreSQL versus Qdrant, for your production environment?

Enter a prompt for Gemini

+ Tools

Fast

Sources

Databases for GenAI - Google Slides.pdf
1. Use Hybrid Search (vector + BM25) as...

RAG Intro - Google Slides.pdf
Combine vector + keyword search (hybrid approach)...

Databases for GenAI - Transcribe Audio.pdf
...what do don't uh hybrid hybrid search is the best. Uh...

RAG Intro - Transcribe Audio.pdf
...exact numbers. Uh hybrid search is quite often better than like only vector only...

Databases for GenAI - Google Slides.pdf
2. Implement metadata filtering...

Databases for GenAI - Google Slides.pdf
...filter = ("department": "finance", ...)

RAG Intro - Google Slides.pdf

2. Explain the different chunking strategies and their trade-offs.

Explain the different chunking strategies and their trade-offs.

A AI Academy DB Expert • Custom Gem

When building a RAG system, chunking is the critical process of dividing large documents into smaller pieces to fit within an LLM's limited context window. Choosing the right strategy involves balancing specificity and retrieval accuracy with the preservation of context. ⚡+1

The three main chunking strategies and their trade-offs are:

1. Fixed-Size Chunking

This is the simplest approach, where text is split at a predetermined number of tokens (e.g., 512) with a small overlap to preserve some context between blocks. ⚡+1

- **Pros:** It is fast, computationally inexpensive, and highly predictable. ⚡+1
- **Cons:** It often breaks text mid-sentence or mid-paragraph, which can cause the system to lose the main context and degrade retrieval quality. ⚡+2
- **When to Use:** Best as a general-purpose starting point or for simple documents where high precision isn't the primary concern. ⚡

2. Recursive Character Chunking

This strategy uses a set of separators (like double newlines, single newlines, or spaces) to split text repeatedly until the desired chunk size is reached. ⚡+1

- **Pros:** It respects the natural structure of the document, keeping paragraphs and sentences together better than fixed splitting. ⚡+1
- **Cons:** While more structured, it is more complex to configure than fixed-size and may still fail to capture the full semantic meaning if a topic spans multiple paragraphs. ⚡+1

⌚ Enter a prompt for Gemini

+ Tools ⚡ Fast ▾ 🔍 ➡

Sources

- PDF Page 8
RAG Intro - Google Slides.pdf
LLMs have limited context windows (32k-200k tokens). Large documents must be divided into smaller pieces.
- PDF Page 8
RAG Intro - Google Slides.pdf
Too large → loses specificity, poor retrieval Too small loses context, fragmented information...
- PDF Page 9
RAG Intro - Google Slides.pdf
Strategy, How It Works , Pros , ... Fixed-Size , Split at 512 tokens, 15% overlap , Simple, fast, predictable , ... Semantic , Use ML to Preserves identify coherent meaning, high units accuracy , ... Recursive , Split...
- PDF Page 7
RAG Intro - Transcribe Audio.pdf
...a fixed size like for example 512 uh tokens and we uh we having like 16:50...
- PDF Page 8
RAG Intro - Transcribe Audio.pdf
...potential quality of the chunks uh that we will have. It is the simp simplest uh...
- PDF Page 8
RAG Intro - Transcribe Audio.pdf

⌚ Enter a prompt for Gemini

+ Tools ⚡ Fast ▾ 🔍 ➡

3. Semantic Chunking

This advanced approach uses machine learning or embedding models to identify coherent units of meaning, splitting text only when a change in topic is detected. ⚡+1

- **Pros:** It preserves the semantic context of the information, leading to much higher accuracy during the retrieval stage. ⚡
- **Cons:** It is significantly slower and more expensive because it requires additional compute power for the ML models to analyze the text. ⚡+1
- **When to Use:** Recommended for technical documentation or complex content where understanding the relationship between ideas is vital. ⚡

Summary of Trade-offs

Strategy	Speed	Context Preservation	Compute Cost
Fixed-Size	Very Fast ⚡	Low ⚡	Low ⚡
Recursive	Fast ⚡	Medium ⚡	Low ⚡
Semantic	Slower ⚡	High ⚡	Higher ⚡

Export to Sheets

For most production cases, starting with a **fixed-size of 512 tokens and a 15% overlap** is the recommended baseline before iterating based on evaluation metrics. ⚡+1

Would you like me to show you how to implement one of these strategies using a library like [LangChain](#) or [LlamaIndex](#)? ⚡

Sources

- Page 8
RAG Intro - Google Slides.pdf
LLMs have limited context windows (32k-200k tokens). Large documents must be divided into smaller pieces.
- PDF Page 8
RAG Intro - Google Slides.pdf
Too large → loses specificity, poor retrieval Too small loses context, fragmented information...
- PDF Page 9
RAG Intro - Google Slides.pdf
Strategy, How It Works , Pros , ... Fixed-Size , Split at 512 tokens, 15% overlap , Simple, fast, predictable , ... Semantic , Use ML to Preserves identify coherent meaning, high units accuracy , ... Recursive , Split...
- PDF Page 7
RAG Intro - Transcribe Audio.pdf
...a fixed size like for example 512 uh tokens and we uh we having like 16:50...
- PDF Page 8
RAG Intro - Transcribe Audio.pdf
...potential quality of the chunks uh that we will have. It is the simp simplest uh...
- PDF Page 8
RAG Intro - Transcribe Audio.pdf
- PDF Page 8
RAG Intro - Transcribe Audio.pdf
...uh cons. Um and for for the for this approach this is the uh the problem that 512...

3. Why is hybrid search better than vector-only search in RAG systems?

The screenshot shows a dark-themed web application interface. At the top, a search bar contains the query "Why is hybrid search better than vector-only search in RAG systems?". Below the search bar, a section titled "AI Academy DB Expert - Custom Gem" displays the following text:

Hybrid search is generally superior to vector-only search because it combines the semantic depth of vector embeddings with the precision of keyword matching. By merging these two approaches, systems can overcome the specific weaknesses inherent in using either method in isolation. 🔗

Key Reasons for Hybrid Superiority

- **Complementary Strengths:** Vector search excels at understanding semantic meaning and intent, while keyword search (like BM25) is better at finding exact matches, specific technical terms, and unique acronyms. 🔗
- **Significant Accuracy Gains:** Benchmarks from Microsoft Azure AI Search show that hybrid search increases accuracy by approximately 10.5% over vector-only search (improving from 43.8 to 48.4 NDCG@3). 🔗
- **Reduction in Failures:** Research from Anthropic indicates that combining hybrid search with a reranker can reduce retrieval failure rates by 67% compared to baseline methods. 🔗
- **Statistical Reliability:** Keyword search uses statistical calculations to determine document relevance based on word frequency, providing a reliable fallback when semantic similarity is too broad or vague. 🔗

The Hybrid + Reranking Workflow

Enter a prompt for Gemini

Fast ▾ 🔍 Tools ▾

Sources

- PDF Page 17 Databases for GenAI - Google Slides.pdf Hybrid Search + Reranking Power...
- PDF Page 14 RAG Intro - Google Slides.pdf Combine vector + keyword search (hybrid approach)...
- PDF Page 17 RAG Intro - Google Slides.pdf RAG = Open-book exam for AI-Retrieves external knowledge at query time Chunking is critical - Start with 512 tokens, 15% overlap, then iterate Hybrid search > vector-only-Combine vector and keyword...
- PDF Page 17 Databases for GenAI - Google Slides.pdf Vector-only: Misses exact matches, acronyms BM25-only: No semantic understanding...
- PDF Page 13 Databases for GenAI - Transcribe Audio.pdf ...addition uh you can utilize like best match search this is what BM25...
- PDF Page 13 Databases for GenAI - Transcribe Audio.pdf ...uh it doesn't have semantic understanding but it's uh calculate the...

The screenshot shows a dark-themed web application interface, similar to the one above. At the top, a search bar contains the query "Why is hybrid search better than vector-only search in RAG systems?". Below the search bar, a section titled "RAG Production Do's and Don'ts" displays the following text:

Significant Accuracy Gains: Benchmarks from Microsoft Azure AI Search show that hybrid search increases accuracy by approximately 10.5% over vector-only search (improving from 43.8 to 48.4 NDCG@3). 🔗

Reduction in Failures: Research from Anthropic indicates that combining hybrid search with a reranker can reduce retrieval failure rates by 67% compared to baseline methods. 🔗

Statistical Reliability: Keyword search uses statistical calculations to determine document relevance based on word frequency, providing a reliable fallback when semantic similarity is too broad or vague. 🔗

The Hybrid + Reranking Workflow

In production-grade systems, hybrid search is often used as the first step in a two-stage architecture: 🔗

1. **Stage 1 (Hybrid Search):** The system performs both vector and keyword searches to retrieve a broad set of candidates, typically the top 100 results. 🔗
2. **Stage 2 (Reranking):** A reranking pipeline analyzes these candidates to select the top 5-10 most relevant pieces of information for the LLM to process. 🔗

This combination ensures that the model receives the most accurate and contextually relevant evidence available. 🔗

Would you like me to explain how to implement BM25 keyword search alongside your vector database to create a hybrid pipeline? 🔗

Sources

Enter a prompt for Gemini

Fast ▾ 🔍 Tools ▾

Sources

- PDF Page 17 Databases for GenAI - Google Slides.pdf Hybrid Search + Reranking Power...
- PDF Page 14 RAG Intro - Google Slides.pdf Combine vector + keyword search (hybrid approach)...
- PDF Page 17 RAG Intro - Google Slides.pdf RAG = Open-book exam for AI-Retrieves external knowledge at query time Chunking is critical - Start with 512 tokens, 15% overlap, then iterate Hybrid search > vector-only-Combine vector and keyword...
- PDF Page 13 Databases for GenAI - Google Slides.pdf Vector-only: Misses exact matches, acronyms BM25-only: No semantic understanding...
- PDF Page 13 Databases for GenAI - Transcribe Audio.pdf ...addition uh you can utilize like best match search this is what BM25...
- PDF Page 13 Databases for GenAI - Transcribe Audio.pdf ...uh it doesn't have semantic understanding but it's uh calculate the...

4. What are the main challenges in building a reliable RAG pipeline?

What are the main challenges in building a reliable RAG pipeline?

A AI Academy DB Expert - Custom Gem

Building a reliable Retrieval-Augmented Generation (RAG) pipeline involves navigating several technical and operational hurdles. According to the provided materials, the main challenges include:

1. Data Preparation and Chunking

- **Contextual Loss:** If chunks are too small, the system loses the core meaning and context of the information. ⚡+3
- **Information Noise:** Chunks that are too large introduce "water" or noise, which can distract the LLM and lead to less specific or inaccurate answers. ⚡+2
- **Complex Document Parsing:** Traditional pipelines often struggle with PDFs containing tables, formulas, and images, leading to a 40–60% information loss in visual documents. ⚡+2

2. Retrieval Quality and Accuracy

- **Semantic Limitations:** Relying solely on vector search can cause the system to miss exact matches, technical acronyms, or specific keywords. ⚡+1
- **Recall Issues:** If the vector database or retrieval algorithm fails to find the most relevant documents, the LLM is likely to hallucinate an answer. ⚡
- **Multi-Hop Reasoning:** Traditional RAG often fails to "connect the dots" across isolated chunks of information when a query requires reasoning across multiple related topics.

Enter a prompt for Gemini

Fast ▾ 🔍 ➤

Sources

PDF Page 8
RAG Intro - Google Slides.pdf
Too large → loses specificity, poor retrieval Too small loses context, fragmented information...

PDF Page 5
RAG Intro - Transcribe Audio.pdf
...the big problem uh that together with the data we are providing like water or 10:51...

PDF Page 7
RAG Intro - Transcribe Audio.pdf
...not the best retrieval uh or a lot of the noise or water uh that you're 15:34...

PDF Page 13
RAG Intro - Google Slides.pdf
Chunking & context windows: If chunks are poorly defined, the retrieved information may miss critical context or include too much irrelevant text.

PDF Page 13
Databases for GenAI - Google Slides.pdf
Layout detection → Misses table structures...

PDF Page 13
Databases for GenAI - Google Slides.pdf
Result: 40–60% information loss on...

3. Model and System Constraints

- Context Window Degradation:** While some models support massive context windows, LLM performance and retrieval quality often start to degrade when windows exceed 32–64k tokens. ⚡+2
- The “Middle” Problem:** LLMs frequently miss critical information located in the middle of a long prompt or context window. ⚡+1
- Latency and Cost:** Implementing complex retrieval systems, agentic patterns, or high-compute semantic chunking adds significant time and expense to each query. ⚡+2

4. Operational and Security Risks

- Security and Access Control:** Managing data leakage is a major challenge; failing to implement strict document-level permissions can lead to unauthorized users accessing sensitive data. ⚡+1
- Data Freshness:** Keeping embeddings synchronized with frequently updated source documents is a significant maintenance burden. ⚡+1
- Lack of Systematic Evaluation:** Many systems rely on “vibes-based” testing rather than robust frameworks, making it difficult to measure true accuracy or performance improvements. ⚡+1

Would you like me to explain how **GraphRAG** or the **CoLPali** model can help solve some of these specific challenges, such as multi-hop reasoning or PDF parsing? ⚡+1

 Sources

Enter a prompt for Gemini

+ Tools *

Fast ▾

Microphone icon

Forward arrow icon

Sources

Too large → loses specificity, poor retrieval Too small loses context, fragmented information...

 PDF Page 5
RAG Intro - Transcribe Audio.pdf
...the big problem uh that together with the data we are providing like water or 10:51...

 PDF Page 7
RAG Intro - Transcribe Audio.pdf
...not the best retrieval uh or a lot of the noise or water uh that you're 15:34...

 PDF Page 13
RAG Intro - Google Slides.pdf
Chunking & context windows: If chunks are poorly defined, the retrieved information may miss critical context or include too much irrelevant text.

 PDF Page 13
Databases for GenAI - Google Slides.pdf
Layout detection → Misses table structures...

 PDF Page 13
Databases for GenAI - Google Slides.pdf
Result: 40–60% information loss on...

 PDF Page 9
Databases for GenAI - Transcribe Audio.pdf

Reflection on Gem Performance

1. How well did your Gem answer the questions? Was it accurate?

My Gem answered the questions accurately and successfully retrieved relevant information from the uploaded PDFs. It clearly explained topics like chunking strategies, hybrid search, and RAG architecture using correct details from the materials. However, even after instructing it to keep responses concise and include citations, it did not always strictly follow those formatting rules. Despite this, the answers remained relevant and grounded in the provided documents.

2. What were the limitations of providing knowledge by copying and pasting it into the instructions?

Copying and pasting the knowledge into the instructions limited flexibility and scalability. While the Gem could recall factual information, it sometimes struggled with maintaining concise formatting or properly citing sources. Managing large volumes of text in the instruction field can also become difficult and less structured. This approach works for smaller datasets but may not be ideal for more complex or dynamic knowledge bases.

3. What kind of “prompt engineering” did you have to do in the instructions to make the Gem behave correctly?

I clearly defined the Gem’s role as a “GenAI RAG Database Expert” and instructed it to only use the uploaded PDFs. I added rules to avoid hallucinations, provide concise answers (maximum five lines), and cite document sections when possible. I also specified the tone and response style. These structured instructions helped improve focus and reduce irrelevant or overly long responses.

4. How is this approach different from just asking the standard Gemini model the same questions?

This customized Gem provided focused, document-specific answers based strictly on the uploaded materials. In contrast, the standard Gemini model would respond using its broader general knowledge, which may be less aligned with the specific course content. The Gem approach ensures better grounding and relevance to the assignment materials.