

Productized & Enterprise RAG

October 2025



Speaker



Lucian Gruia

Data Science Principal Lead

- Head of Engineering, Romania
- 13+ years of experience in Software Engineering
- Led cross-functional teams across 8 time zones in Telecom, Fintech, Aerospace, and MedTech
- Mentor @Google Developer Groups Romania
- Research Assistant, AI Multimedia Lab (National University of Science & Technology, Bucharest)
- Supervised 12 BSc and MSc diploma projects

Agenda

01 Existing repos & products

02 Pros, Cons, Limitations and Considerations

Quick recap



What is RAG

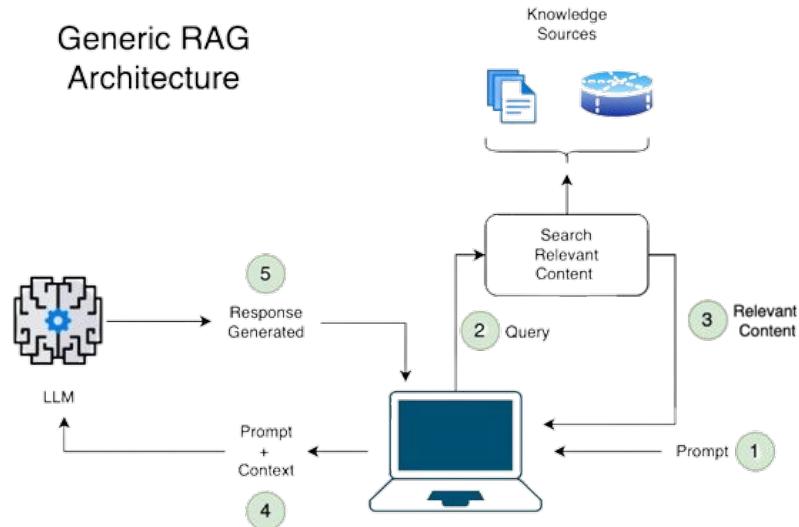


A RAG system essentially correlates a **user's prompt** with a relevant **data chunk**. It does this by identifying **the most semantically similar** chunk from the database.

This chunk then becomes **the context** for the prompt.

When **passed to the Large Language Model (LLM)**, it enables the system to provide a relevant answer within the **given context**.

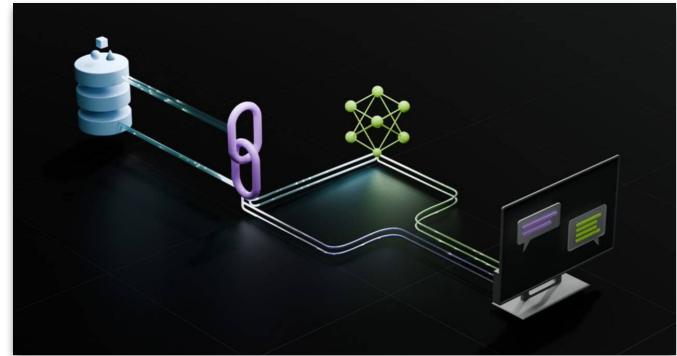
Generic RAG Architecture



Why do we need RAG?



- **Expands Knowledge Base**
RAG accesses a vast external database, enriching its knowledge beyond initial training data
- **Improves Accuracy**
Enhances response precision by integrating relevant, real-time information
- **Adaptable**
Effectively handles novel and niche queries
- **Increases Efficiency**
Streamlines information retrieval and generation process
- **Versatile Applications**
Useful across various fields, from customer support to research



Source: [What Is Retrieval-Augmented Generation, aka RAG?](#)

LLMs are limited to the knowledge they were originally trained on, and retraining or fine-tuning them is slow and expensive.

Embeddings. Similarity

- **Embeddings**

Numerical representations of concepts, in a high-dimensional space, capturing semantic meaning.

- **Similarity:**

- Lexical: entities are alike in appearance
- **Semantic:** entities are alike in meaning

- In RAG we represent entities **by describing** them.

This is a form of **knowledge representation**.

Example: Mountain, River, Canal

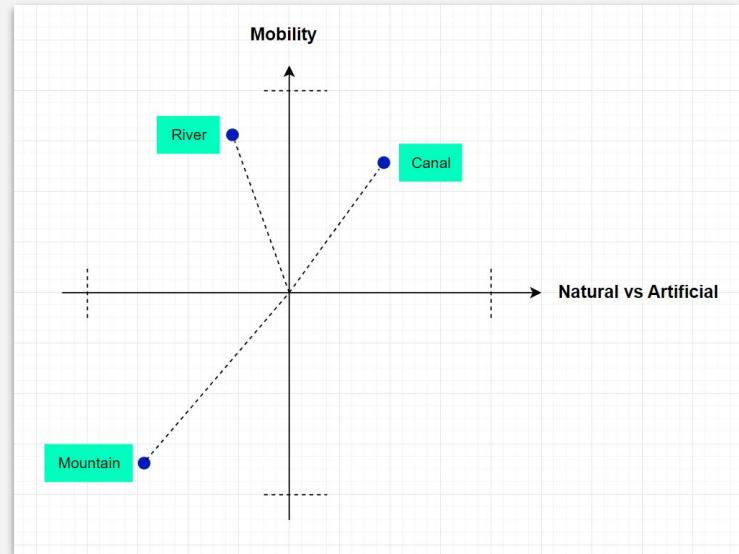
One hot encoding

Mountain:	1
River:	2
Canal:	3

2-Dimensional Space

[Natural vs Artificial, Mobility]

Mountain:	[-0.7, -0.8]
River:	[-0.3, 0.7]
Canal:	[0.4, 0.5]



Read more: [Wikipedia - Cosine Similarity](#)

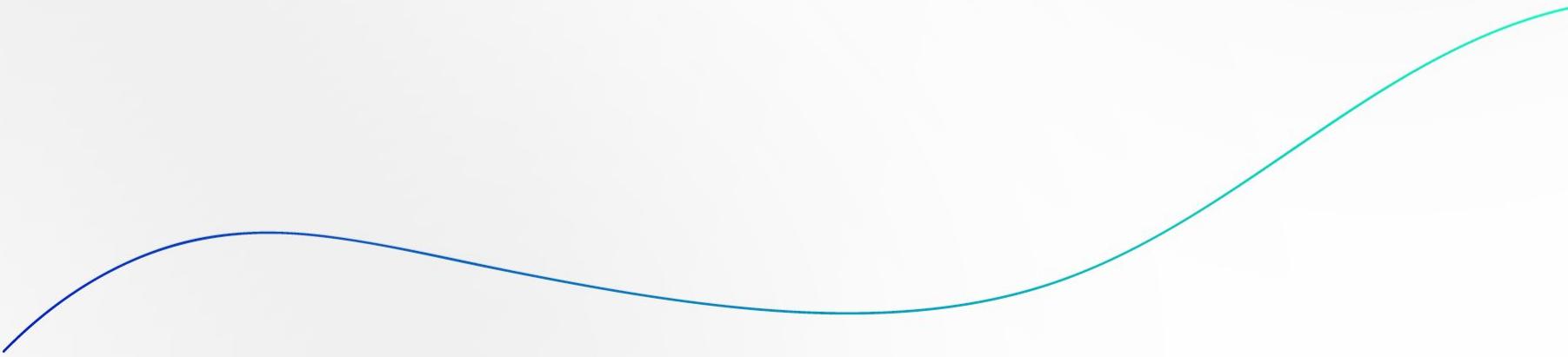
Multilinguality



Generated with DALL-E 3

- **Multilingual Capabilities**
RAG and chatbots can process and respond in multiple languages, enhancing global reach and accessibility.
- **Cross-lingual Understanding**
These systems understand queries in one language and retrieve information in another, facilitating cross-lingual interactions.
- **Language-Agnostic Embeddings**
Use of embeddings that capture semantic meanings across languages.
- **Challenges**
Handling idiomatic expressions, cultural nuances, and varying sentence structures in different languages.

Existing repos & products



NotebookLM

Understand Anything

Your research and thinking partner, grounded in the information you trust, built with the latest Gemini models.

[Try NotebookLM](#)

Your AI-Powered Research Partner

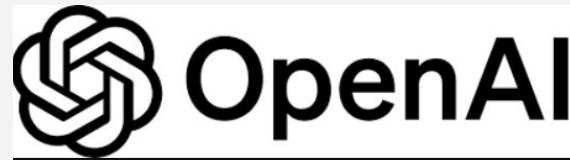


Build custom experts with Gems

Gems are your custom AI experts for help on any topic. Gems can be anything from a career coach or brainstorm partner to a coding helper. Get started with our suite of premade Gems, or build your own custom Gems tailored to your unique needs.



Custom GPTs



November 6, 2023 · Product

Introducing GPTs

You can now create custom versions of ChatGPT that combine instructions, extra knowledge, and any combination of skills.



LM Studio



The screenshot shows the LM Studio interface. At the top center is a white box containing the LM Studio logo (purple square with white horizontal lines) and the text "LM Studio". Below this is the main application window. On the left is a sidebar with "Chats" (Secret project, Ascii, Investigate EOS, Intelligence test), "File System Project" (783 tokens), and a search bar. The main area has a purple header bar with "openai/gpt-oss-20b" and "Eject" buttons. A central text input field says "Please write an example self contained toy filesystem in C++". Below it is a code editor with the following C++ code:

```
// toy_fs.cpp
#include <iostream>
#include <memory>
#include <unordered_map>
#include <vector>
#include <string>
#include <sstream>

class File {
public:
    explicit File(std::string name) : name_(std::move(name)) {}
    const std::string& name() const { return name_; }

    void write(const std::string& data) { content_ = data; }
    const std::string& read() const { return content_; }

private:
    std::string name_;
    std::string content_;
};

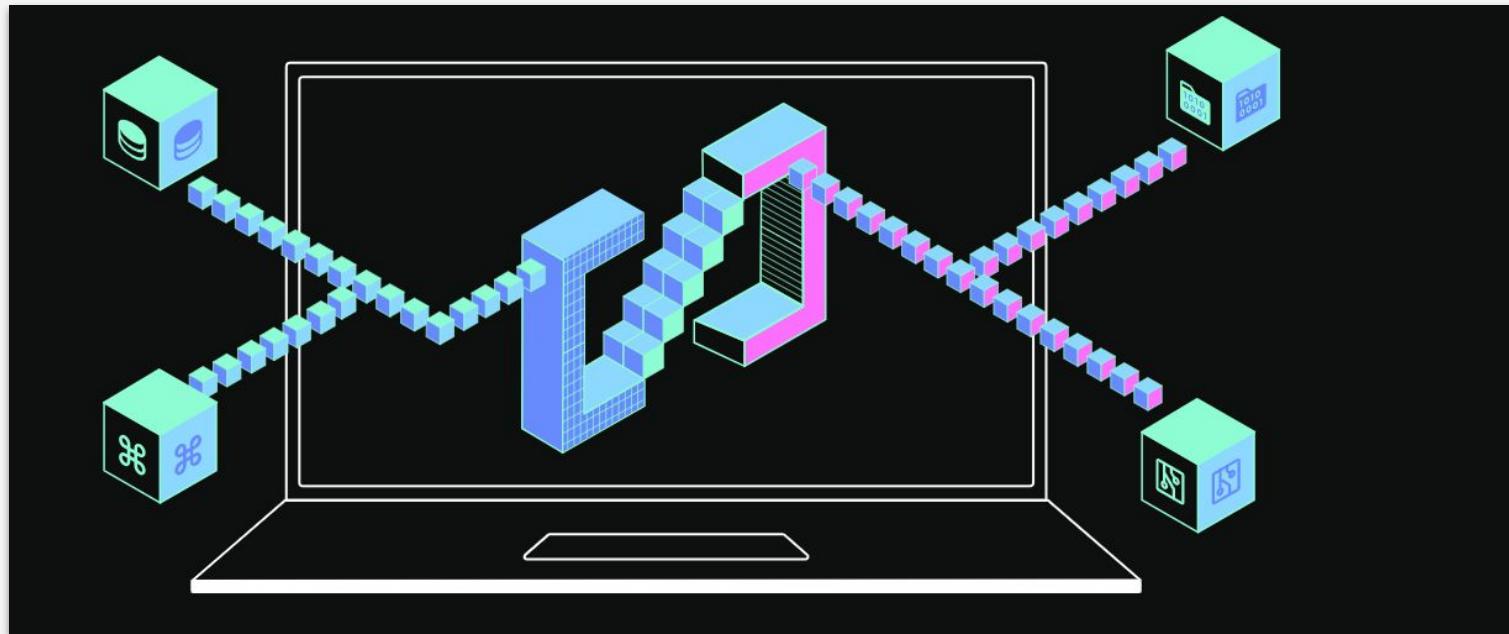
Send a message to the model...
```

At the bottom of the code editor is a "Reasoning Effort" button. The footer of the application window displays "LM Studio 0.3.24 (Build 2)" and "User: Power User Developer". It also shows "Input token count: 0" and "Tokens: 783/131072". The status bar at the bottom right shows "RAM: 27.47 GB | CPU: 0.00 %".

Anything LLM



♾ Anything LLM



OpenWeb UI

A screenshot of the Open WebUI interface. At the top, there's a large header with the text "OI Open WebUI". Below the header is a sidebar with a "Prompts" dropdown and a search bar labeled "Search Prompts". A plus sign (+) button is also in the sidebar. The main content area features a section titled "Sponsored by Join the Open WebUI Team" with a "OI" logo and a hiring message: "We are hiring! Join us to build the future of AI applications.". Below this is a "New Functions" section with two items: "#1 FILTER v1.0.0" (Memory System) and "#2 PIPE v0.6.1" (Nano Banana). Further down are "#3 FILTER v0.2.0" (Qwen Thinking Filter) and "#4 FILTER v1.0" (April-1.5-15B-Thinker - Final+Think). Each item has a brief description, the author's handle, and a "See All" link.

Prompts ▾ Search Prompts +

Sponsored by Join the Open WebUI Team

OI

We are hiring! Join us to build the future of AI applications.

New Functions

#1 **FILTER** v1.0.0

Memory System
Memory system with intelligent skip detection, hybrid retrieval, and background consolidation.

@itayfur

#2 **PIPE** v0.6.1

Nano Banana
This code implements Google Flash 2.5 Image with API usage for image generation and editing. You can...

@anaumer

#3 **FILTER** v0.2.0

Qwen Thinking Filter
filter qwens thinking into a closable element

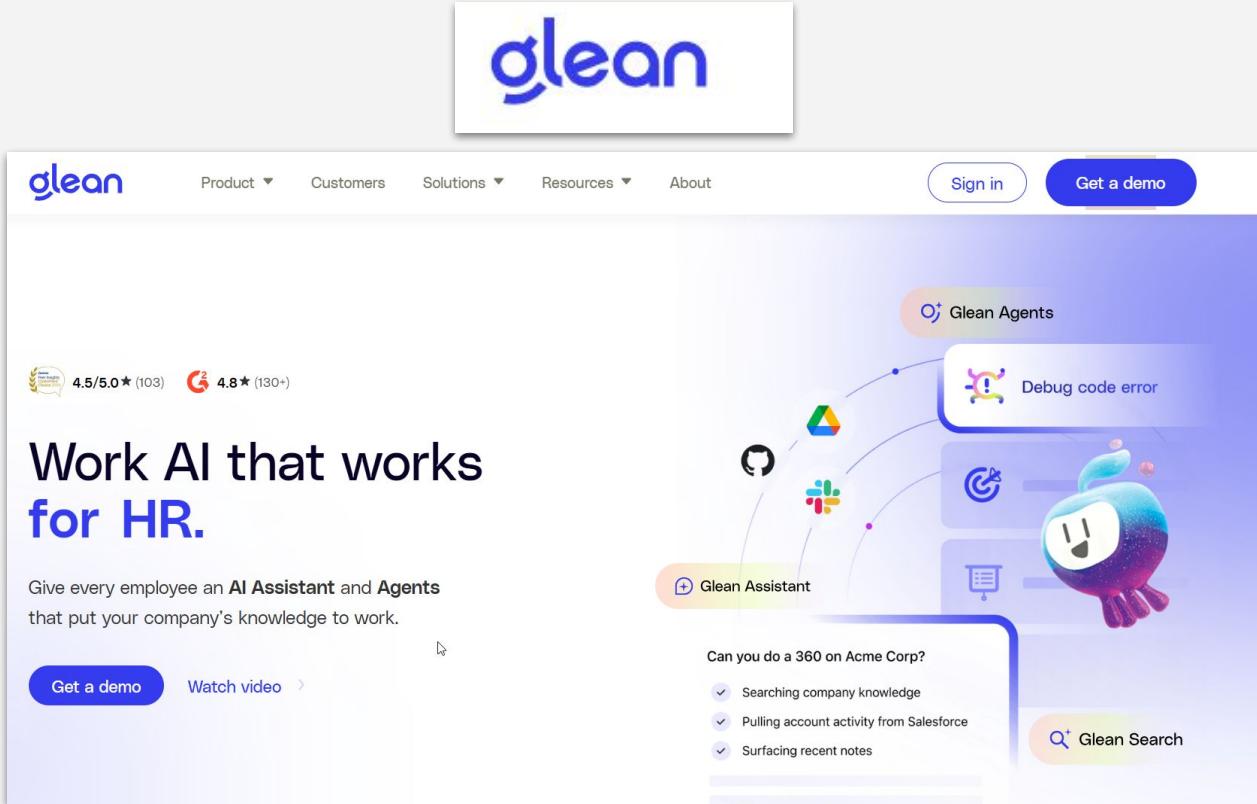
@eklsf

#4 **FILTER** v1.0

April-1.5-15B-Thinker - Final+Think
Makes April-1.5-15B-Thinker integrate cleanly with Open WebUI by rendering a native <think> panel an...

@supczinskib

See All



The image shows the Glean AI homepage. At the top, there's a large blue "glean" logo. Below it is a navigation bar with links: Product ▾, Customers, Solutions ▾, Resources ▾, and About. To the right are "Sign in" and "Get a demo" buttons. On the left, there are two rating sections: one for "Glean Agents" with a yellow badge showing 4.5/5.0 ★ (103) reviews, and another for "Glean Assistant" with a red badge showing 4.8 ★ (130+) reviews. The main headline reads "Work AI that works for HR." Below it, a sub-headline says "Give every employee an AI Assistant and Agents that put your company's knowledge to work." At the bottom left are "Get a demo" and "Watch video" buttons. The right side features a colorful illustration of various AI components: "Glean Agents" (represented by a brain icon), "Debug code error" (represented by a code editor icon), "Glean Assistant" (represented by a speech bubble icon), "Glean Search" (represented by a magnifying glass icon), and a large, friendly-looking AI character. A sidebar on the right lists tasks: "Can you do a 360 on Acme Corp?" followed by three items: "Searching company knowledge", "Pulling account activity from Salesforce", and "Surfacing recent notes".

glean

Product ▾ Customers Solutions ▾ Resources ▾ About

Sign in Get a demo

4.5/5.0 ★ (103) 4.8 ★ (130+)

Work AI that works for HR.

Give every employee an **AI Assistant** and **Agents** that put your company's knowledge to work.

Get a demo Watch video >

Glean Agents

Debug code error

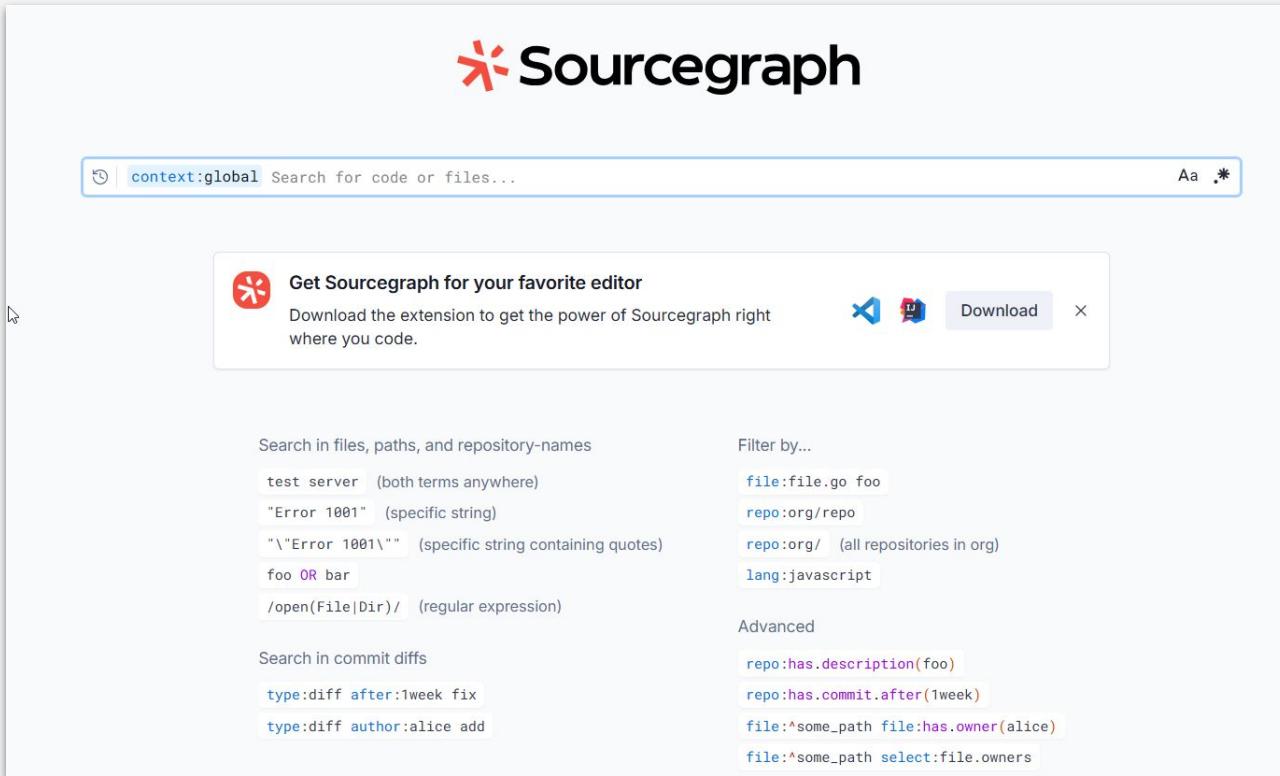
Glean Assistant

Glean Search

Can you do a 360 on Acme Corp?

- ✓ Searching company knowledge
- ✓ Pulling account activity from Salesforce
- ✓ Surfacing recent notes

Sourcegraph



The screenshot shows the Sourcegraph search interface. At the top is the Sourcegraph logo (a red asterisk-like icon followed by the word "Sourcegraph"). Below it is a search bar with placeholder text "Search for code or files...". To the right of the search bar are icons for "Aa" (font size) and a magnifying glass. A modal window titled "Get Sourcegraph for your favorite editor" is open, showing instructions to download the extension for VS Code and Atom, with "Download" and "X" buttons. The main search area has two sections: "Search in files, paths, and repository-names" and "Filter by...". The "Search in files, paths, and repository-names" section contains examples like "test server" (both terms anywhere), "'Error 1001'" (specific string), "\"Error 1001\"\"", "foo OR bar", and "/open(File|Dir)/" (regular expression). The "Filter by..." section contains examples like "file:file.go foo", "repo:org/repo", "repo:org/ (all repositories in org)", and "lang:javascript". Below these are sections for "Advanced" search queries, such as "repo:has.description(foo)" and "file:^some_path file:has.owner(alice)".

Search for code or files...

Get Sourcegraph for your favorite editor

Download X

Search in files, paths, and repository-names

- test server (both terms anywhere)
- "Error 1001" (specific string)
- "\Error 1001\""" (specific string containing quotes)
- foo OR bar
- /open(File|Dir)/ (regular expression)

Filter by...

- file:file.go foo
- repo:org/repo
- repo:org/ (all repositories in org)
- lang:javascript

Advanced

- repo:has.description(foo)
- repo:has.commit.after(1week)
- file:^some_path file:has.owner(alice)
- file:^some_path select:file.owners

Pros, Cons, Limitations and Considerations

Pros & Cons



✓ Pros

- Enhances factual accuracy via external knowledge
- Reduces hallucinations vs. pure LLMs
- Enables dynamic updates without retraining
- Modular, domain-adaptable architecture
- Scalable across multimodal or multi-source data

⚠️ Cons

- Retrieval quality depends on data prep & embeddings
- Latency increases with retrieval pipeline
- Requires infra (vector DBs, indexing, chunking, etc.)
- Context window still limits answer depth

Limitations & Considerations



Limitations

- Not reasoning-first, still retrieval-driven
- Poor handling of implicit or fuzzy queries
- No built-in memory or learning from feedback
- Struggles with conflicting or outdated sources

Considerations

- Balance retrieval vs. generation cost & latency
- Continuous evaluation of relevance & drift
- Governance for data freshness and accuracy
- Potential upgrade path: **Agentic / Tool-use RAG**

Enterprise Data Protection and Privacy



- **Safeguarding sensitive information** and maintaining customer trust.
- **Regulatory Compliance:** Adherence to laws like GDPR, which mandate data security and privacy.
- **Encryption & Anonymization:** Secure data and protect individual identities.
- **Access Control:** Implementing strict permissions and access policies to sensitive data.
- **Regular Audits:** Monitoring and auditing to identify and rectify any vulnerabilities.
- Importance of using **External Data** & RAG models.
- **Augment context** in order to get accurate completions.



Image by Jan Alexander from Pixabay

Quiz Time



What is the main goal of RAG?

- A. To train larger models
- B. To reduce hallucinations using external knowledge
- C. To replace fine-tuning
- D. To speed up token generation

What is a key limitation of RAG systems?

- A. They cannot retrieve multimodal data
- B. They require GPU inference for retrieval
- C. They depend heavily on data quality and embeddings
- D. They cannot use APIs

What's the next evolution beyond basic RAG?

- A. Fine-tuned GPT models
- B. Static FAQ chatbots
- C. Agentic or tool-using RAG architectures
- D. Prompt templates with zero retrieval

Key Takeaways from the session



- Boosts **factual accuracy** by grounding LLMs in **real data**
- Reduces **hallucinations**, but depends on retrieval quality
- **No retraining needed**, knowledge updated via data refresh
- Latency-complexity **trade-off**: retrieval adds overhead
- Still **limited** reasoning and context awareness
- Next **evolution** → Agentic, monitored, tool-using RAGs



Thank you!

