

PYTHON.ID SHARING SESSION

BUILDING RAG ON LARGE DOCUMENT USING OPEN SOURCE LLM

INTRODUCTION

▶ M. ALIF RAMADHAN

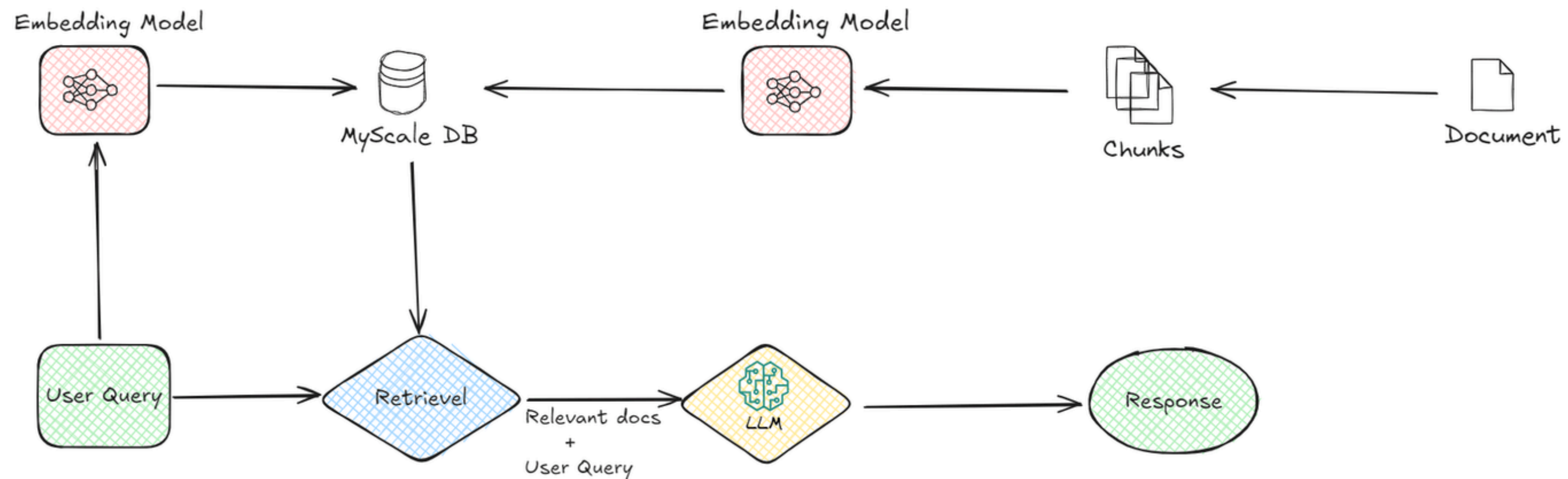
AI Engineer | Pecinta AI

▶ LATEST ACTIVITY

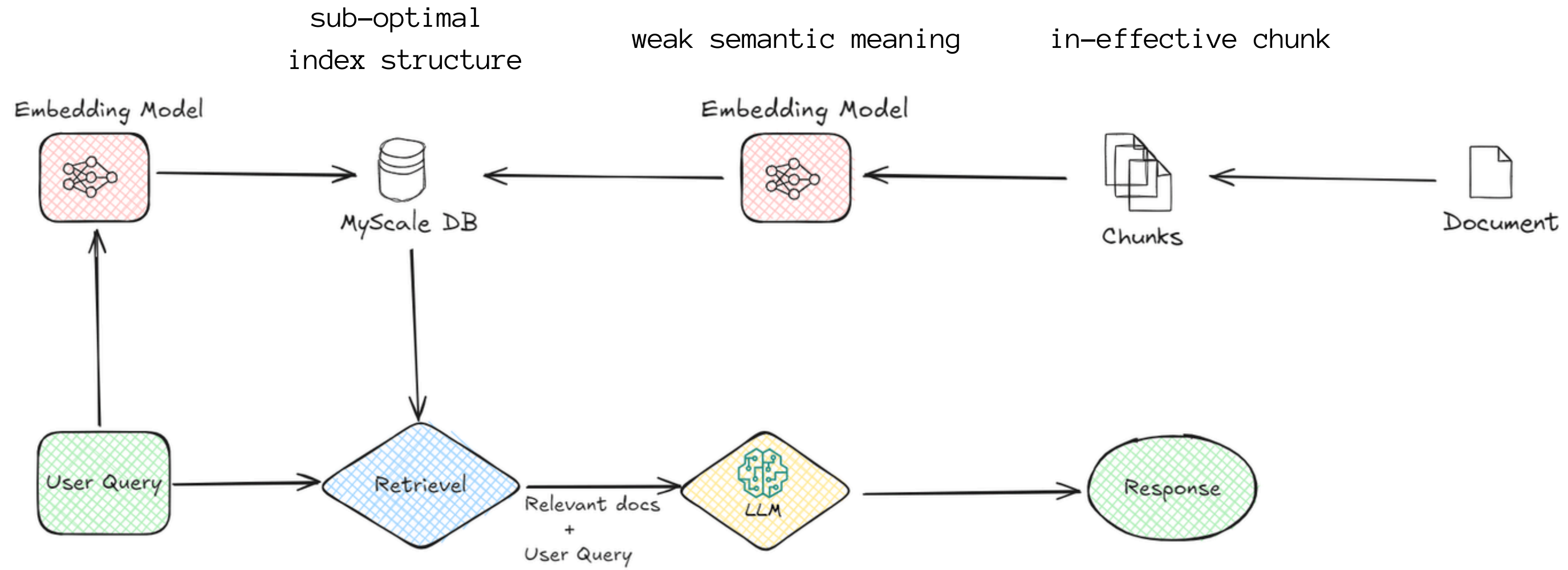
- Mahasiswa Teknik Elektro
- Built community on facebook: IMPHNEN
- Make other people's tasks easier with Gen. AI



RAG: SOLUTION FOR AI HALLUCINATION? OR **ANOTHER ISSUE?**



HIGHLIGHT THE ISSUE



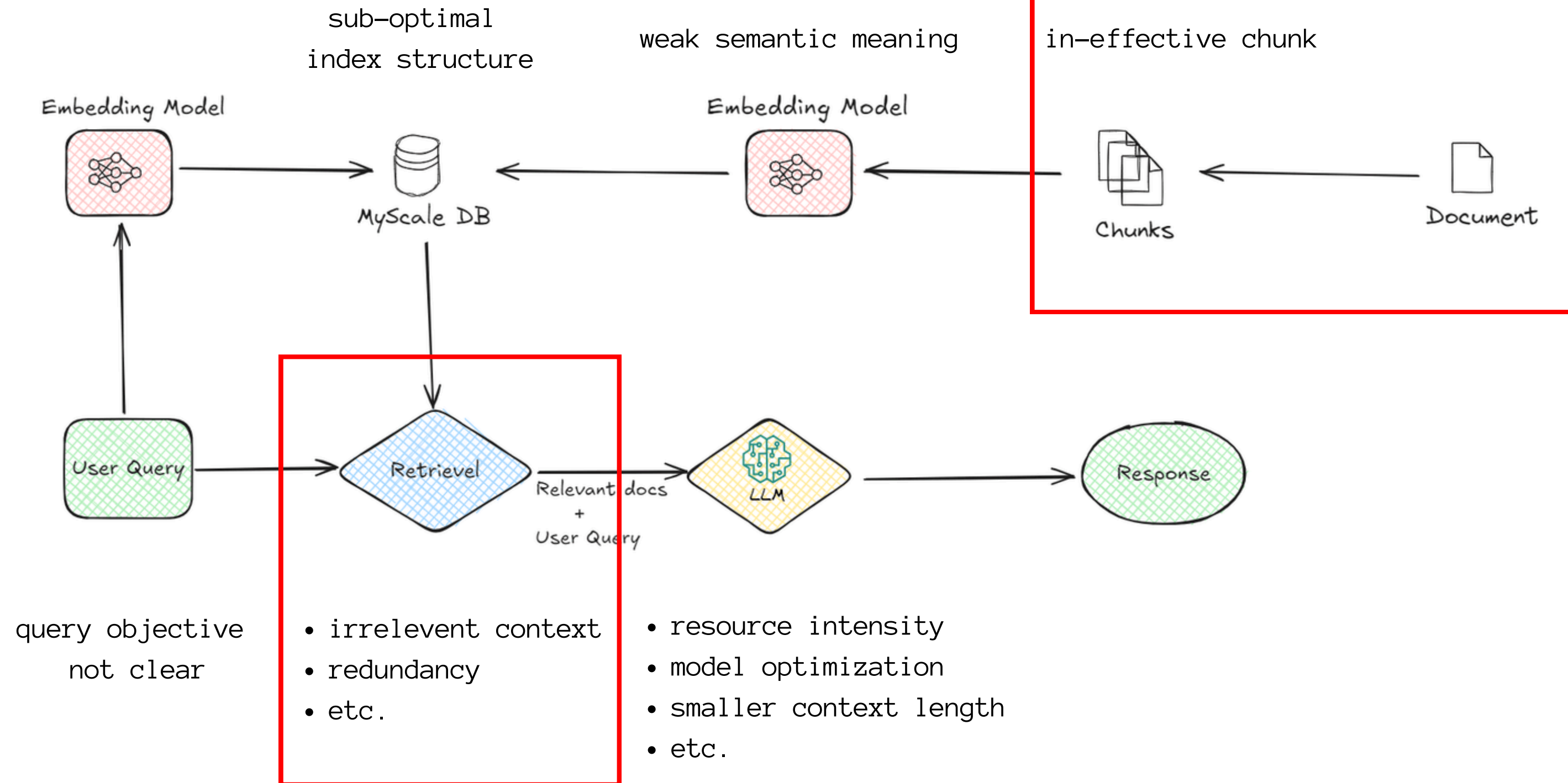
query objective not clear

- irrelevent context
- redundancy
- etc.

- resource intensity
- model optimization
- smaller context length
- etc.

FOCUS TOPIC

INGEST



RETRIEVAL

INGEST STRATEGY

WHAT ARE THE MOST COMMON VULNERABILITIES? TOP 5 VULNERABILITY CATEGORIES

What's different: Where we saw a notable difference is the rise of the Broken Access Control category, and we're not alone in our observations. OWASP noticed the same uptick. In the 2017 version of their list, Broken Access Control was at 5th place, but it shot up to the top position in their 2021 update with this stat attached: "94% of applications were tested for some form of Broken Access Control."

One possible reason for this growth in operations. We like to think of it like this: If you had a house with one door, all you have to worry about is keeping that one door locked. But if you expand that house and add a garage door and 5 windows—or in our case, adopt more technology, expand systems, widen networks, and manage more users—you're more likely to forget to lock everything. Creating or updating your user/access matrix can help.

Top 5 Findings

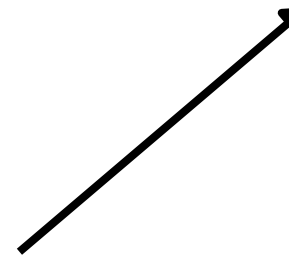
For a detailed breakdown of how to fix and prevent each of the findings listed here, check the Appendix in The State of Pentesting 2021.

Diving deeper into our data, we uncovered a similar pattern when it came to the top 5 findings: most flaws from 2020 stayed at the top, but a couple of new issues made their way onto the list.

	2020	2021
#01	Broken Access Control: Insecure Direct Object References (IDOR)	Cross-Site Scripting (XSS): Stored
#02	Cross-Site Scripting (XSS): Stored	Broken Access Control: Insecure Direct Object References (IDOR)
#03	Components with Known Vulnerabilities: Outdated Software Versions	Components with Known Vulnerabilities: Outdated Software Versions
#04	Broken Access Control: Username/Email Enumeration	Server Security Misconfiguration: Insecure SSL or TLS protocols
#05	Cross-Site Scripting (XSS): Reflected	Server Security Misconfiguration: Lack of Security Headers

Figure 2: The top 5 most common findings in our database for 2020 and 2021

For a detailed breakdown of the top vulnerabilities according to asset type, check APPENDIX A.



- Do Some Qualitative Survey
- Gather User Expectation
- Define Scope per Milestone



- ingest strategy
- retrieval strategy
- benchmark evaluation

INGESTION - 101

FIRST ITER.

- Text Parser + Recursive Chunking + RegEx Pattern

SECOND ITER.

- Layout Parser + Text Parser + Parent-Child + Recursive Chunking

THIRD ITER.

- Multimodal Parser + Parent-Child + Semantic Chunking + Metadata Extraction

N ITER.

- DYOR



Layout Parser

DocLayNet: A Large Human-Annotated Dataset for Document-Layout Analysis

Text
Sigrig Pfitzmann
IBM Research
Rueschlikon, Switzerland
bpf@zurich.ibm.com

Text
Christoph Auer
IBM Research
Rueschlikon, Switzerland
cau@zurich.ibm.com

Text
Michele Dolh
IBM Research
Rueschlikon, Switzerland
dol@zurich.ibm.com

Text
Ahmed S. Nassar
IBM Research
Rueschlikon, Switzerland
ahn@zurich.ibm.com

Text
Peter Staar
IBM Research
Rueschlikon, Switzerland
taa@zurich.ibm.com

Section-Header
Section-Header

Text
Accurate document layout analysis is a key requirement for high quality PDF document conversion. With the recent availability of public, large ground-truth datasets such as PubLayNet and DocBank deep-learning models have proven to be very effective at layout detection and segmentation. While these datasets are of adequate size to train such models, they severely lack in layout variability since they are sourced from scientific article repositories such as PubMed and arXiv only. Consequently, the accuracy of the layout segmentation drops significantly when these models are applied on more challenging and diverse layouts. In this paper, we present DocLayNet, a new, publicly available, document-layout annotation dataset in COCO format. It contains 80863 manually annotated pages from diverse data sources to represent a wide variability in layouts. For each PDF page, the layout annotations provide labelled bounding-boxes with a choice of 11 distinct classes. DocLayNet also provides a subset of double- and triple-annotated pages to determine the inter-annotator agreement. In multiple experiments we provide baseline accuracy scores (in mAP) for a set of popular object detection models. We also demonstrate that these models all approximately 10% behind the inter-annotator agreement. Furthermore, we provide evidence that DocLayNet is of sufficient size. Lastly, we compare models trained on PubLayNet, DocBank and DocLayNet, showing that layout predictions of the DocLayNet trained models are more robust and thus the preferred choice for general-purpose document-layout analysis.

Section-Header
Section-Header

Text
Information systems → Document structure; Applied computing → Document analysis; Computing methodologies → Machine learning; Computer vision; Object detection.

Footnote
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation name for third-party components of this work must be honored.

Footnote
Sigrig Pfitzmann, Christoph Auer, Michele Dolh, Ahmed S. Nassar, and Peter Staar. 2022. DocLayNet: A Large Human-Annotated Dataset for Document-Layout Analysis. In *Proceedings of the 20th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22)*, August 14–18, 2022, Washington, DC, USA. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3534674.3539043>

Figure 1: Four examples of complex page layouts across different document categories.



Multimodal Parser

Fig. 3. OTSL description of table structure: A - table example; B - graphical representation of table structure; C - mapping structure on a grid; D - OTSL structure encoding; E - explanation on cell encoding.

Table 2. TSR and cell detection results compared between OTSL and HTML on the PubTabNet [22], FinTabNet [21] and PubTables-1M [14] data sets using TableFormer [9] (with enc=6, dec=6, heads=8).

Data set	Language	TEDs		mAP(0.75)	Inference time (secs)	
		simple	complex			
PubTabNet	OTSL	0.965	0.934	0.955	0.88	2.73
	HTML	0.969	0.927	0.955	0.857	5.39
FinTabNet	OTSL	0.955	0.961	0.959	0.862	1.85
	HTML	0.917	0.922	0.92	0.722	3.26
PubTables-1M	OTSL	0.987	0.964	0.977	0.896	1.79
	HTML	0.983	0.944	0.966	0.889	3.26

5.3 Qualitative Results

The OTSL vocabulary is comprised of the following tokens:

- "C" cell - a new table cell that either has or does not have cell content
- "L" cell - left-looking cell, merging with the left neighbor cell to create a span
- "U" cell - up-looking cell, merging with the upper neighbor cell to create a span
- "X" cell - cross cell, to merge with both left and upper neighbor cells
- "NL" - new-line, switch to the next row.

A notable attribute of OTSL is that it has the capability of achieving lossless expression using a data frame called `df` in R where you want to summarize `x` by month:

```
require(plyr)
df <- data.frame(
  x = runif(120, 1, 168),
  y = runif(120, 7, 334),
  z = runif(120, 1.7, 20.7),
  month = rep(c(5,6,7,8),30),
  week = sample(1:4, 120, TRUE)
)
ddply(df, .(month, week), summarize,
  mean = round(mean(x), 2),
  sd = round(sd(x), 2))
)
```

Fig. 3. OTSL description of table structure: A - table example; B - graphical representation of table structure; C - mapping structure on a grid; D - OTSL structure encoding; E - explanation on cell encoding.

Table 2. TSR and cell detection results compared between OTSL and HTML on the PubTabNet [22], FinTabNet [21] and PubTables-1M [14] data sets using TableFormer [9] (with enc=6, dec=6, heads=8).

5.3 Quantitative Results

The OTSL vocabulary is comprised of the following tokens:

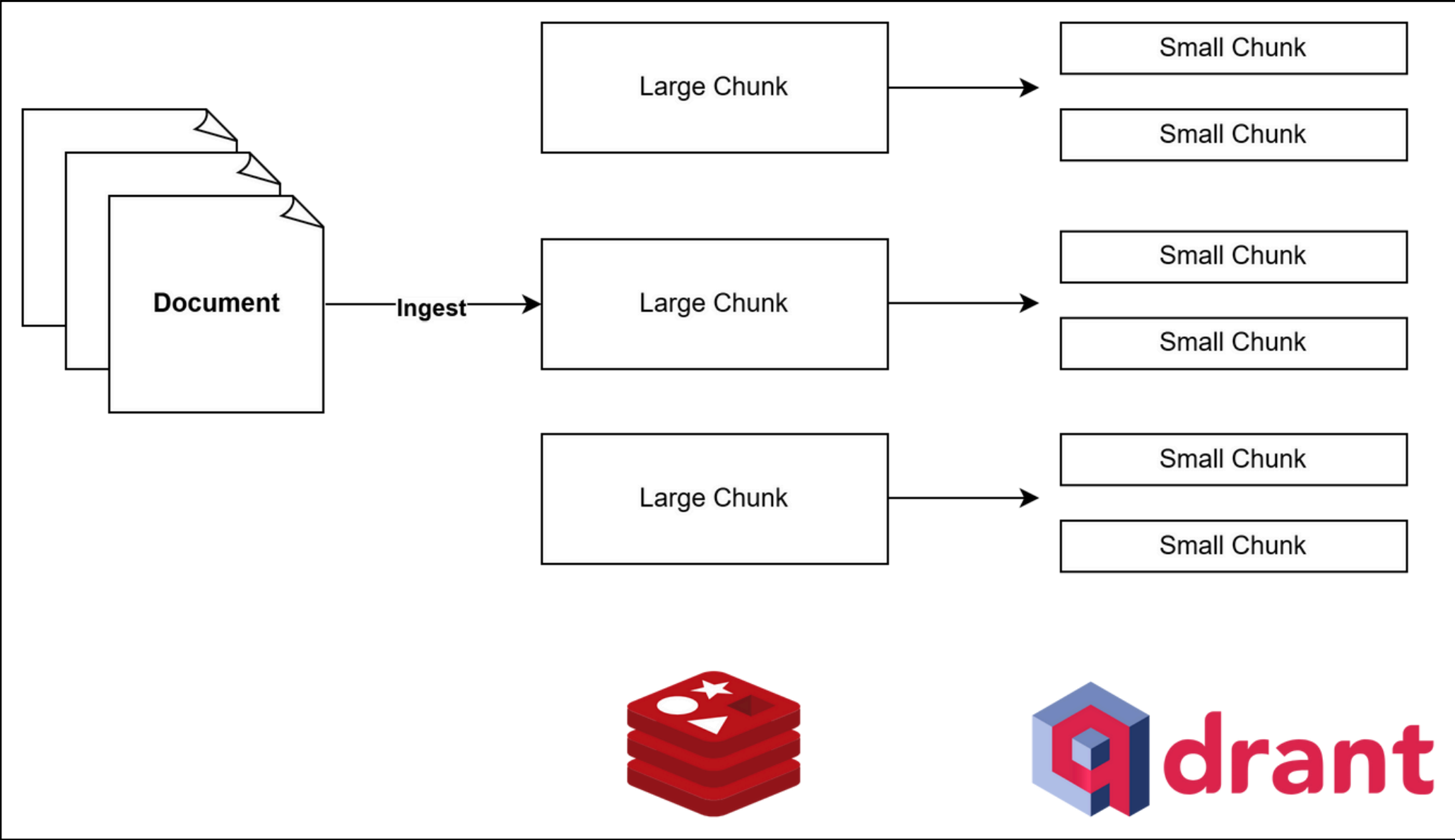
- "C" cell - a new table cell that either has or does not have cell content
- "L" cell - left-looking cell, merging with the left neighbor cell to create a span
- "U" cell - up-looking cell, merging with the upper neighbor cell to create a span
- "X" cell - cross cell, to merge with both left and upper neighbor cells
- "NL" - new-line, switch to the next row.

```
require(plyr)
df <- data.frame(
  x = runif(120, 1, 168),
  y = runif(120, 7, 334),
  z = runif(120, 1.7, 20.7),
  month = rep(c(5,6,7,8),30),
  week = sample(1:4, 120, TRUE)
)
ddply(df, .(month, week), summarize,
  mean = round(mean(x), 2),
  sd = round(sd(x), 2))
)
```



INGESTION - 101

Parent-Child



RETRIEVAL - 101



FIRST ITER.

- Go To [MTEB Benchmark](#) + Experiment (Language Type)
- Setup LLM Monitoring



LangSmith



opik



Langfuse

*note: tools above might not stable yet on "production"

Deploy
⋮

RAG Chat System

Upload Documents

Upload PDF

Drag and drop file here
Limit 200MB per file • PDF

Browse files

Conversations

New Conversation

Select Conversation

Collaboration in DevSecOps Security Framework (2025-03-19 03:06)

New Conversation (2025-03-19 03:06)

Hai, Let's Start a Conversation (2025-03-19 03:02)

Hai! Let's Start a Conversation (2025-03-18 22:20)

RAG Chat System

Conversation: Collaboration in DevSecOps Security Framework

🗨️ What role does collaboration play in achieving security in a DevSecOps framework?

📄 Based on the provided context, it is clear that collaboration plays a significant role in achieving security in a DevSecOps framework. The concept of DevSecOps is built around the idea of integrating security into the development and operational processes through modern integrated security paradigms.

Pillar 1: Collective Responsibility emphasizes the importance of changing the organization's mindset, ideas, customs, and behaviors regarding software security. It highlights the need for security to be seen as a collective responsibility, rather than someone else's responsibility. This suggests that collaboration among different stakeholders, including development, operations, and security teams, is essential for achieving security in a DevSecOps framework.

The text also mentions the importance of "collective collaboration" in the context of DevOps, which implies that collaboration is a key aspect of the DevOps practice. This collaboration can help to improve outcomes by integrating existing security practices into the development and operational processes.


Furthermore, the text emphasizes the need for a "Pragmatic" approach to security in DevOps, which involves connecting all stakeholders (development, operations, and security) in a manner that ensures security is built into applications and the software lifecycle. This approach requires collaboration among different stakeholders to ensure that security is integrated into the development and operational processes.


In summary, collaboration plays a crucial role in achieving security in a DevSecOps framework by integrating security into the development and operational processes, changing the organization's mindset regarding software security, and ensuring that security is built into applications and the software lifecycle.


👍 🗨️

Thank you for your feedback!


NnA301023/cygen




 1
Contributor

 0
Issues


 0
Stars

 0
Forks

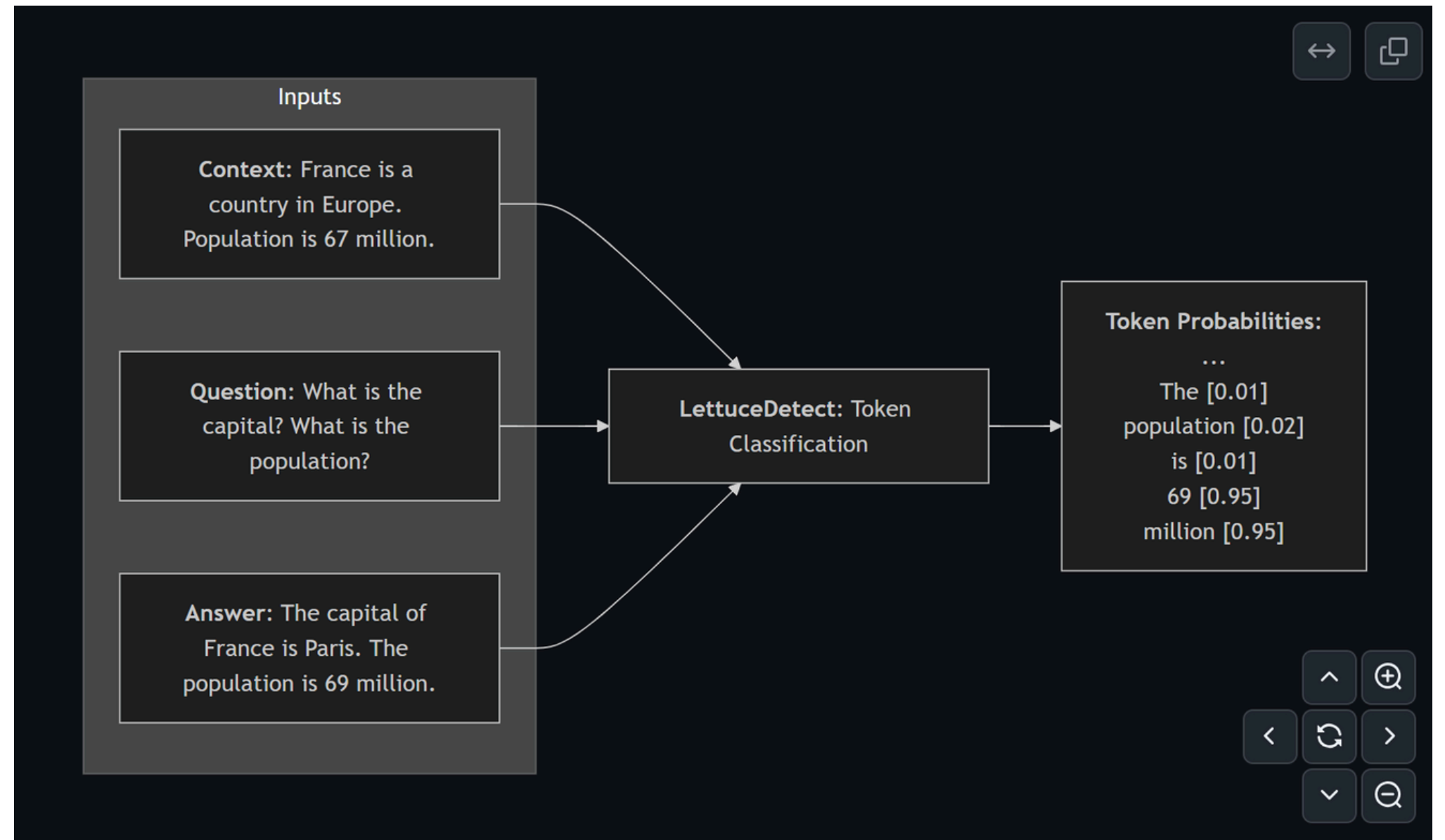


NnA301023/cygen

Contribute to NnA301023/cygen development by creating an account on GitHub.

 GitHub

EVALUATION - 101



CONCLUSION

- ▶ INGESTION
- ▶ RETRIEVAL
- ▶ EVALUATION





Sesi QnA

Dibuka

THANKS
