Building a Context Aware RAG Pipeline

Yue Jun Yuan

Challenges

- Siloed knowledge
- Limited search capabilities
- Operational risks

Difficulty Accessing Relevant Information

Employees struggle to **locate the correct SOP information** quickly due to the growing volume and complexity of enterprise documentation

Limitation of Keyword Based Search

Traditional keyword-based search tools **lack contextual understanding**, frequently **returning irrelevant or incomplete results** that lead to delays and mistakes

Operational Risks from SOP Non-compliance

Inefficient knowledge access increases operational risk, onboarding friction, and **inconsistent adherence to standardized procedures** across teams

Solution

- RAG Pipeline built on corpus of organization policy documents

Integrated Domain Knowledge

Incorporate and **connect organizational data** and facts for tailored accurate responses

Improve accessibility

Make complex or lengthy SOPs more **accessible** by providing answers in natural, conversational language

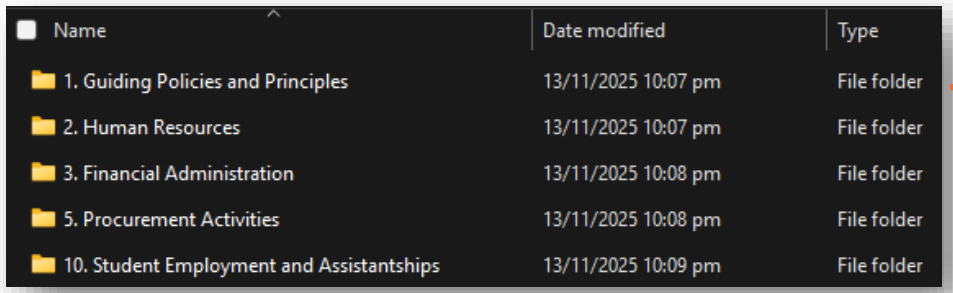
Source Traceability

Clear **source attribution** ensures trustworthy answers and reduces misinformation

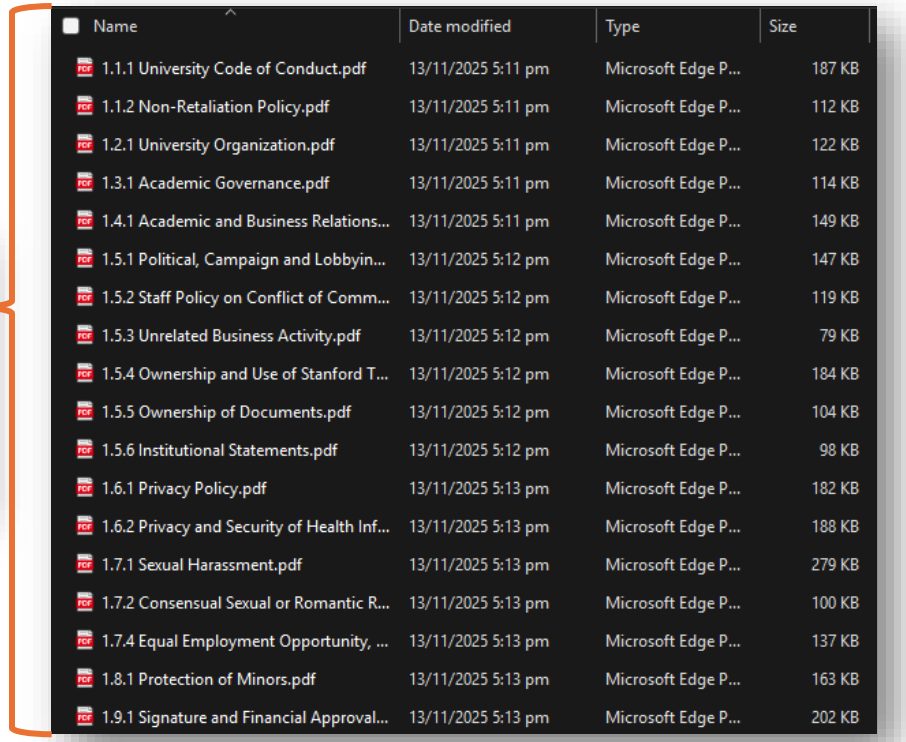
Datasets

- Corpus used in this pipeline: Stanford Administrative Guide (Chapters | Administrative Guide)

- Selected chapters were extracted from 5 specific domains
  - ➢ Total of 101 individual documents extracted as PDF
  - ➢ Can be scaled up as required



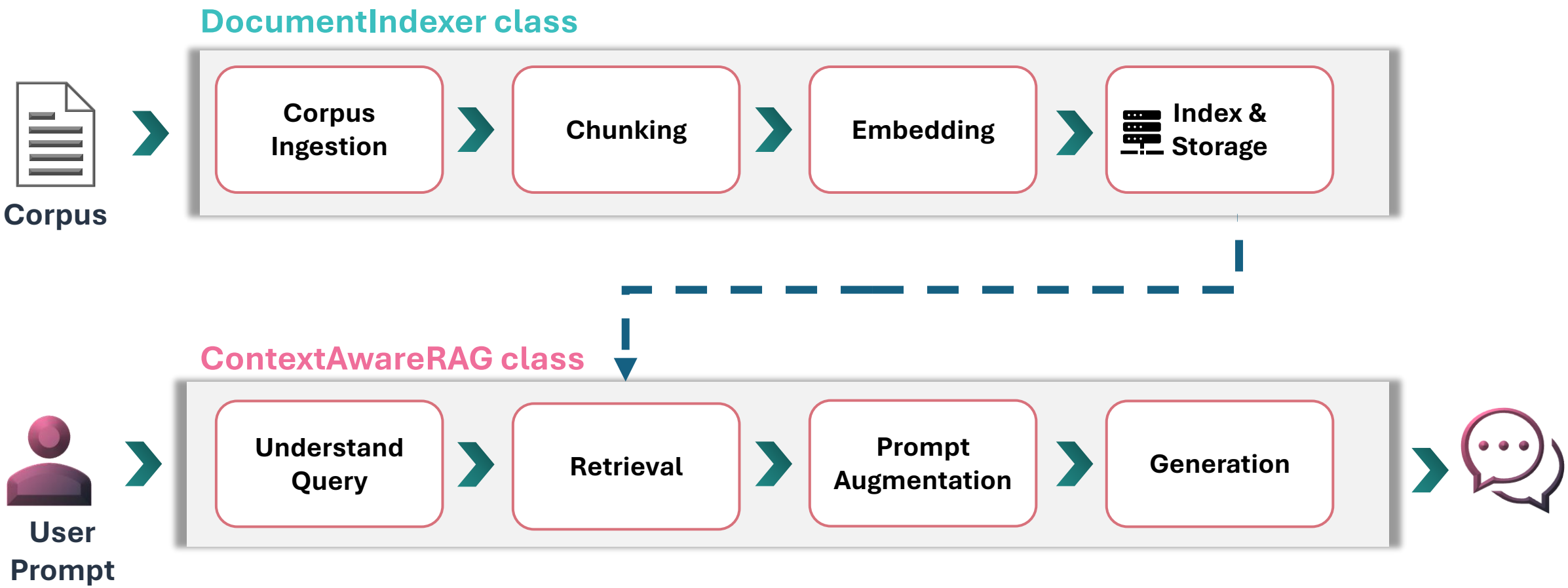
Name	Date modified	Type
1. Guiding Policies and Principles	13/11/2025 10:07 pm	File folder
2. Human Resources	13/11/2025 10:07 pm	File folder
3. Financial Administration	13/11/2025 10:08 pm	File folder
5. Procurement Activities	13/11/2025 10:08 pm	File folder
10. Student Employment and Assistantships	13/11/2025 10:09 pm	File folder



Name	Date modified	Type	Size
1.1.1 University Code of Conduct.pdf	13/11/2025 5:11 pm	Microsoft Edge P...	187 KB
1.1.2 Non-Retaliation Policy.pdf	13/11/2025 5:11 pm	Microsoft Edge P...	112 KB
1.2.1 University Organization.pdf	13/11/2025 5:11 pm	Microsoft Edge P...	122 KB
1.3.1 Academic Governance.pdf	13/11/2025 5:11 pm	Microsoft Edge P...	114 KB
1.4.1 Academic and Business Relations...	13/11/2025 5:11 pm	Microsoft Edge P...	149 KB
1.5.1 Political, Campaign and Lobbyin...	13/11/2025 5:12 pm	Microsoft Edge P...	147 KB
1.5.2 Staff Policy on Conflict of Comm...	13/11/2025 5:12 pm	Microsoft Edge P...	119 KB
1.5.3 Unrelated Business Activity.pdf	13/11/2025 5:12 pm	Microsoft Edge P...	79 KB
1.5.4 Ownership and Use of Stanford T...	13/11/2025 5:12 pm	Microsoft Edge P...	184 KB
1.5.5 Ownership of Documents.pdf	13/11/2025 5:12 pm	Microsoft Edge P...	104 KB
1.5.6 Institutional Statements.pdf	13/11/2025 5:12 pm	Microsoft Edge P...	98 KB
1.6.1 Privacy Policy.pdf	13/11/2025 5:13 pm	Microsoft Edge P...	182 KB
1.6.2 Privacy and Security of Health Inf...	13/11/2025 5:13 pm	Microsoft Edge P...	188 KB
1.7.1 Sexual Harassment.pdf	13/11/2025 5:13 pm	Microsoft Edge P...	279 KB
1.7.2 Consensual Sexual or Romantic R...	13/11/2025 5:13 pm	Microsoft Edge P...	100 KB
1.7.4 Equal Employment Opportunity, ...	13/11/2025 5:13 pm	Microsoft Edge P...	137 KB
1.8.1 Protection of Minors.pdf	13/11/2025 5:13 pm	Microsoft Edge P...	163 KB
1.9.1 Signature and Financial Approval...	13/11/2025 5:13 pm	Microsoft Edge P...	202 KB

1. Guiding Policies and Principles
2. Human Resources
3. Financial Administration
4. Giving to Stanford
5. Procurement Activities
6. Computing
7. Health and Safety
8. Services
9. Organization Charts
10. Student Employment and Assistantships
11. Global Activities

Pipeline Architecture



DocumentIndexer

Initial build:

```python
def build_vector_store_from_raw(self, path):
    self.load_documents(path)
    self.build_faiss_index()
    self.save_index()
```

Subsequent load:

```python
def load_vector_store_from_disk(self):
    self.load_index()
```

Corpus Ingestion → **Chunking** → **Embedding** → **Index & Storage**

- Automated PDF ingestion using **PyPDF**
- **Customizable** chunk size and overlap
- Preserves **metadata** – source document, etc
- **193** total chunks created from 101 PDFs
- Chunks saved as pickle and json files


```
{
  "chunk_id": 0,
  "text": "1.1.1 University Code of Conduct\nAuthority\nThis Guide Memo was approved
  "source": "1.1.1 University Code of Conduct.pdf"
},
```


- Model: **OpenAI's text embedding 3 small**
- Embeddings with **1,536 dimensions**
- Maximum input token**: 8,191 tokens**
- Saved as **.npy** file

- Indexed embeddings in a **FAISS vector store** for high-speed similarity search and low-latency retrieval
- Saved as **.faiss** file

ContextAwareRAG

01 **Track conversation context**

- Retains the latest **3 question–answer** interactions to preserve context continuity and enhance response relevance

02 **Build Weighted Query embedding**

- Blends the current query embedding at **2x weight** with the **mean** of the last 3 queries (representing the semantic center or "average meaning" of the historical queries) before normalizing to unit length
  - **Current query: 66.7% weight**
  - **Historical queries: 33.3% weight**

03 **Return the top results**

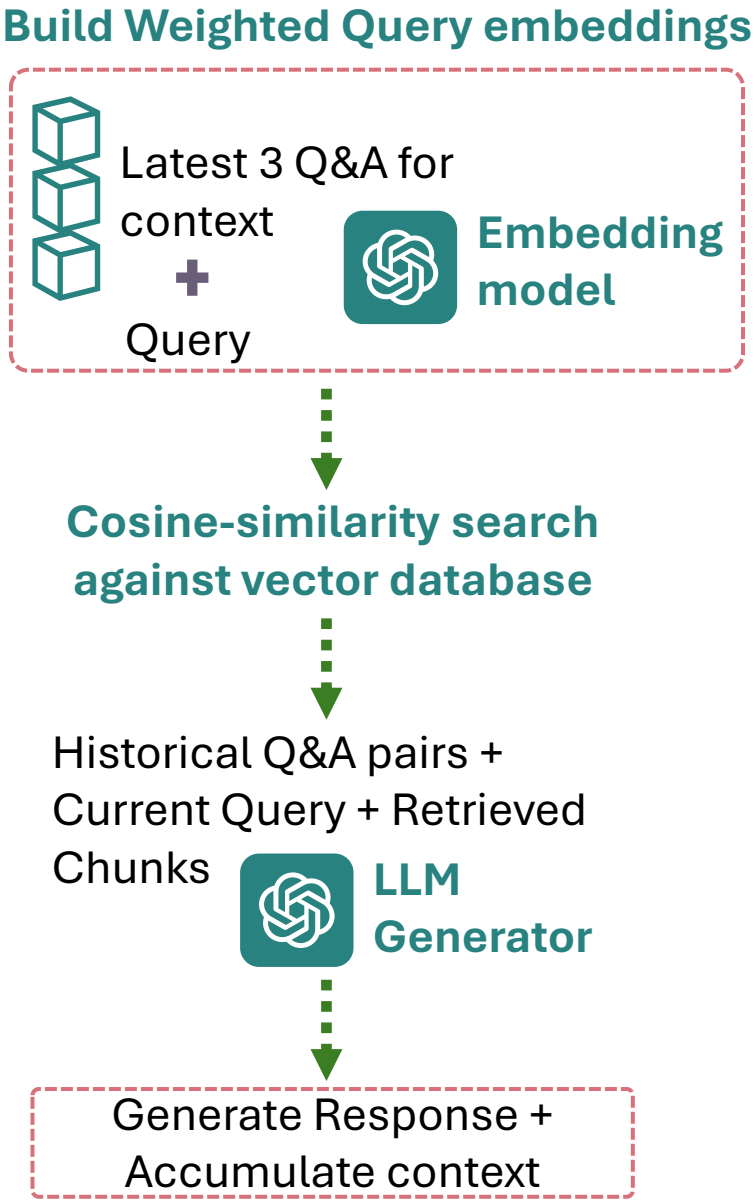
- Retrieve **top 3** most relevant chunks from vector store based on cosine similarity with weighted query embedding

04 **Construct the prompt with context + history**

- Appends retrieved chunks and conversation history to the query, using **recency-based truncation** to enforce token limit

05 **Store the query and answer for future weighted retrieval**

- Saves each new question–answer pair so future searches can leverage recent conversational context



Multi-turn questions

```python
# First Question
question1 = "What is the process for entering into a lease agreement for office space that cost approximately $2,000,000?"
answer1 = rag.query(question1)
print("Answer 1:", answer1)
```

✓ 10.9s Python

```
Answer 1: To enter into a lease agreement for office space that costs approximately $2,000,000, you would need to follow the outlined process in the Real Estate Leases policy. Here are the steps you should take:

1. **Capital Planning Process**:
   - Ensure that the lease is included as a capital request in the Capital Planning Process, which is managed by the Vice President for Land, Buildings & Real Estate. This process occurs in conjunction with the annual budgeting process for the University.

2. **Approval Threshold**:
   - Since the total anticipated contracted cash payments over the term of the lease are $2,000,000, you will need to obtain approval from the Provost, as this amount is above the $1,000,000 threshold.

3. **Requisition Submission**:
   - Prior to negotiating lease terms, submit an off-campus lease requisition form to the Associate Vice President, Real Estate. This form can be downloaded from the Land, Buildings & Real Estate website.

4. **Lease Negotiation**:
   - The Associate Vice President, Real Estate or their designee will assist in negotiating the lease terms.
```

```python
# Second question (relying on context)
question2 = "Who should be the approving authority?"
answer2 = rag.query(question2)
print("Answer 2:", answer2)
```

✓ 7.8s Python

```
Answer 2: The approving authority for entering into a lease agreement for office space that costs approximately $2,000,000 would be the Provost. This is because the total anticipated contracted cash payments over the term of the lease exceed the $1,000,000 threshold, which requires approval from the Provost as per the university's policies.
```

- Historical context provides for **seamless**, flowing conversation with user
- Useful for complex policy or procedures requiring **multiple follow up** prompts

Deployment Considerations

- Decoupling the DocumentIndexer from the RAG query engine allows the vector store to be refreshed independently, without interrupting live RAG service
  - ➢ New documents can be ingested, chunked, and embedded offline, and the vector index can be swapped in only when ready—ensuring continuous availability.

- Current public cloud deployment using API-based models (text-embedding-3-small and gpt-4o-mini) appropriate for corpus not containing highly sensitive or confidential data

- OpenAI's hosted embedding and generation models offer state-of-the-art quality and low operational latency
  - ➢ Operates on consumption-based pricing model – inference cost scales with usage volume

- Where strict data privacy, regulatory, or data-residency requirements exist – local LLMs and embedding models may be necessary
  - ➢ More infrastructural requirements while potentially trading off response quality or latency