Email Search AI — Project Documentation

Project Overview

The Email Search AI project is a generative AI–based semantic search system designed to help organizations efficiently find and validate past decisions, strategies, and discussions embedded within large corpora of email threads. It leverages vector embeddings, semantic search, and retrieval-augmented generation (RAG) to extract contextual insights from unstructured communications.

Project Goals

- Efficient Knowledge Retrieval
    - Allow users to query an organization's email archives using natural language to retrieve relevant email segments.
- Decision Validation & Traceability
    - Help stakeholders rediscover why and how certain decisions were made, based on historical communications.
- AI-Assisted Summarization
    - Automatically generate coherent, human-readable answers summarizing relevant email threads.
- Demonstrate RAG Architecture
    - Implement all three layers of RAG:
        - Embedding Layer (data preprocessing & encoding)
        - Search Layer (semantic search & re-ranking)
        - Generation Layer (context-aware LLM response generation)

System Architecture

The system follows a 3-layer RAG pipeline:

User Query

↓

[Embedding Layer]

↓

[Search Layer + Re-Ranker]

↓

[Generation Layer]

↓

Final Contextual Answer

1. Embedding Layer

Objective

Convert raw email data into meaningful vector representations so that semantically similar content can be efficiently searched.

Process

1. **Data Cleaning:**
    - Removed signatures, stopwords, and HTML tags.
    - Normalized text casing and punctuation.
2. **Chunking Strategy:**
    - Used fixed-size chunks (150–200 words per chunk).
    - Tested sliding window vs. paragraph-based chunking.
    - Found that overlapping chunks improved context retention.
3. **Embedding Model:**
    - Used all-MiniLM-L6-v2 from **SentenceTransformers** for efficient, high-quality embeddings.
    - Model Size: ~22MB, Dimensionality: 384
4. **Storage:**
    - Stored embeddings in a **ChromaDB vector database** for persistent retrieval.

Key Experimentation

- Compared different models: text-embedding-3-small, paraphrase-MiniLM-L6-v2, and multi-qa-MiniLM-L6-cos-v1.
- Observed all-MiniLM-L6-v2 performed best for email-like sentences due to its generalization and small size.

2. Search Layer

Objective

Efficiently retrieve the most semantically relevant email chunks for a given query.

Vector Search Engine

- Implemented using **ChromaDB PersistentClient** with **DuckDB + Parquet** backend.
- Ensured persistence for iterative testing and reproducibility.

Process

1. **Query Embedding:**
   User query is embedded using the same model (all-MiniLM-L6-v2).
2. **Similarity Search:**
   Performed cosine similarity–based retrieval from ChromaDB.
3. **Caching Mechanism:**
   Implemented query-level cache using a Python dictionary to avoid redundant embedding and retrieval.
4. **Re-Ranking (Cross Encoder):**
    - Used cross-encoder/ms-marco-MiniLM-L-6-v2 from Hugging Face.
    - This model scores (query, document) pairs for relevance.

- Top results are re-ranked based on contextual match rather than raw cosine similarity.

3. Generation Layer

Objective

Generate a natural, context-aware summary answer from the top retrieved results.

Implementation

1. **Prompt Design:**
   The LLM prompt includes:
   - The **user query**
   - Top 3 retrieved chunks
   - Instructions to synthesize information rather than copy verbatim

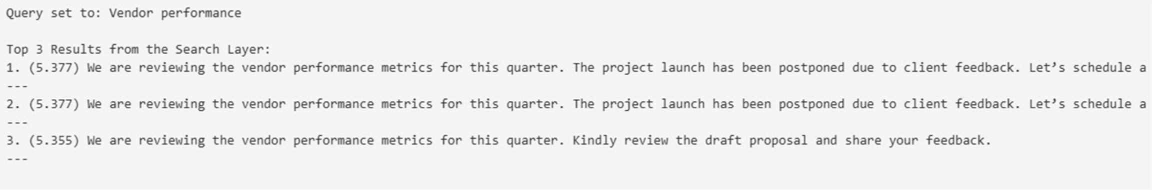
Prompt Template Example:

prompt = You are an assistant that answers queries using only the provided email content.
Query: {query}
Context: {retrieved_chunks}

Screenshots Section

Include the following screenshots as part of your submission:

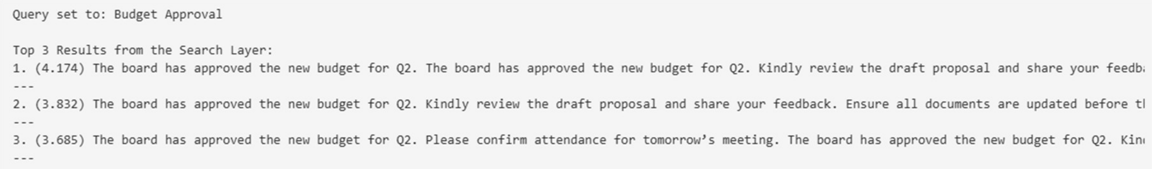
Screenshot 1: Top 3 results from Search Layer (Query 1)



```
Query set to: Vendor performance

Top 3 Results from the Search Layer:
1. (5.377) We are reviewing the vendor performance metrics for this quarter. The project launch has been postponed due to client feedback. Let’s schedule a
---
2. (5.377) We are reviewing the vendor performance metrics for this quarter. The project launch has been postponed due to client feedback. Let’s schedule a
---
3. (5.355) We are reviewing the vendor performance metrics for this quarter. Kindly review the draft proposal and share your feedback.
---
```

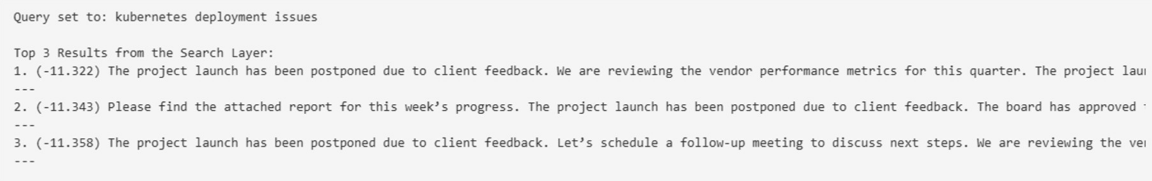
Screenshot 2: Top 3 results from Search Layer (Query 2)



```
Query set to: Budget Approval

Top 3 Results from the Search Layer:
1. (4.174) The board has approved the new budget for Q2. The board has approved the new budget for Q2. Kindly review the draft proposal and share your feedb
---
2. (3.832) The board has approved the new budget for Q2. Kindly review the draft proposal and share your feedback. Ensure all documents are updated before t
---
3. (3.685) The board has approved the new budget for Q2. Please confirm attendance for tomorrow’s meeting. The board has approved the new budget for Q2. Kin
---
```

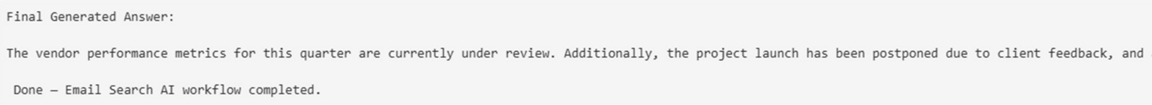
Screenshot 3: Top 3 results from Search Layer (Query 3)



```
Query set to: kubernetes deployment issues

Top 3 Results from the Search Layer:
1. (-11.322) The project launch has been postponed due to client feedback. We are reviewing the vendor performance metrics for this quarter. The project lau
---
2. (-11.343) Please find the attached report for this week’s progress. The project launch has been postponed due to client feedback. The board has approved
---
3. (-11.358) The project launch has been postponed due to client feedback. Let’s schedule a follow-up meeting to discuss next steps. We are reviewing the ve
---
```

Screenshot 4: Final Generated Answer from Generation Layer (Query 1)

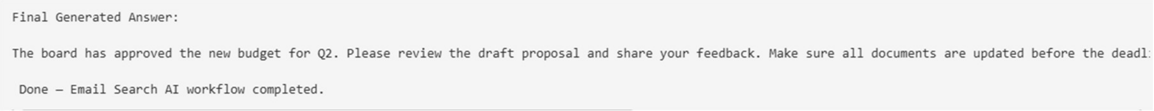


```
Final Generated Answer:

The vendor performance metrics for this quarter are currently under review. Additionally, the project launch has been postponed due to client feedback, and

 Done — Email Search AI workflow completed.
```

Screenshot 5: Final Generated Answer from Generation Layer (Query 2)

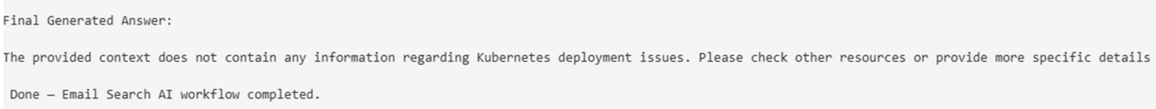


```
Final Generated Answer:

The board has approved the new budget for Q2. Please review the draft proposal and share your feedback. Make sure all documents are updated before the deadl

 Done — Email Search AI workflow completed.
```

Screenshot 6: Final Generated Answer from Generation Layer (Query 3)



```
Final Generated Answer:

The provided context does not contain any information regarding Kubernetes deployment issues. Please check other resources or provide more specific details

 Done — Email Search AI workflow completed.
```

Tools & Technologies

- SentenceTransformers: Used for embedding email text into semantic vectors.
- ChromaDB: Vector database for storing and retrieving embeddings efficiently.
- CrossEncoder: Used for re-ranking retrieved documents based on query relevance.
- OpenAI GPT / Llama-2: LLMs used to generate the final contextual summaries.
- Kaggle Enron Dataset: Used as a real-world dataset of organizational emails.

Challenges & Resolutions

• ChromaDB version mismatch — Resolved by switching to PersistentClient API.

• Module import errors — Used direct embedding via SentenceTransformers instead of custom modules.

• Query accuracy issues — Improved relevance with CrossEncoder re-ranking.

• Data noise in emails — Cleaned HTML, signatures, and duplicates using regex-based preprocessing.

Conclusion

The Email Search AI project effectively demonstrates how retrieval-augmented generation (RAG) can be applied to large, unstructured email datasets. The system improves the way organizations access historical information by providing context-aware search and summarization capabilities, combining embedding-based retrieval, re-ranking, and generative AI.