RAG Email Pipeline Prototype

This repository contains a modular Retrieval-Augmented Generation (RAG) pipeline designed to extract and process email data, generate vector embeddings, and provide AI-assisted answers using the OpenAI API. Currently email data from local outlook client is the only input, the input options should be extended soon.

Project Overview

This project demonstrates a complete RAG workflow using email content as input:

1. **Email Extraction** – Fetch emails from Outlook folders
2. **Text Cleaning** – Normalize content, remove signatures and quoted replies
3. **Chunking** – Split emails into semantically coherent text chunks
4. **Embedding** – Convert chunks into vectors using local or API models
5. **Retrieval** – Match queries against embedded chunks
6. **Answer Generation** – Construct prompts and query OpenAI to synthesize answers

Project Structure

```
configs/            # YAML configuration files (general + task-specific)
data/               # Cleaned emails, chunks, logs, embeddings
debug/              # Query debug output
outputs/            # Final generated answers
scripts/            # Modular pipeline components
tests/              # Test cases and sample inputs
docs/               # Notes, planning, and requirements
```

Configuration

Main settings live in YAML files under `configs/`. Task configs include:

- `config.yaml` – Core settings
- `test_full_api.yaml`, `test_full_local.yaml` – API and local embedding runs

Use `RAGPipeline.configure_task()` to create a new config dynamically.

Requirements

Install dependencies with:

```
pip install -r docs/requirements.txt
```

Recommended:

- Python 3.10+
- Windows with Outlook installed (for email extraction)

Running the Pipeline

Example full pipeline run:

```
from scripts.pipeline.rag_pipeline import RAGPipeline

pipeline = RAGPipeline(config_path="configs/tasks/test_full_api.yaml")
pipeline.run_full_pipeline(query="How can I find deleted POLs in Alma?")
```

Or build it step by step using `add_step()` and `run_steps()`.

API Keys

Set your OpenAI key in an environment variable:

```
export OPEN_AI=your-api-key
```

Or supply it directly when creating the `APIClient`.

License

MIT License

Acknowledgments

Built with:

- OpenAI API
- FAISS
- sentence-transformers
- Microsoft Outlook COM API (via `pywin32`)