Exercise 4-1: Source Verification

Objective

Set up a RAG (Retrieval-Augmented Generation) pipeline to give your bot access to the CAG Annual Report 2024/25, then verify whether its answers are grounded in the actual document.

Prerequisites

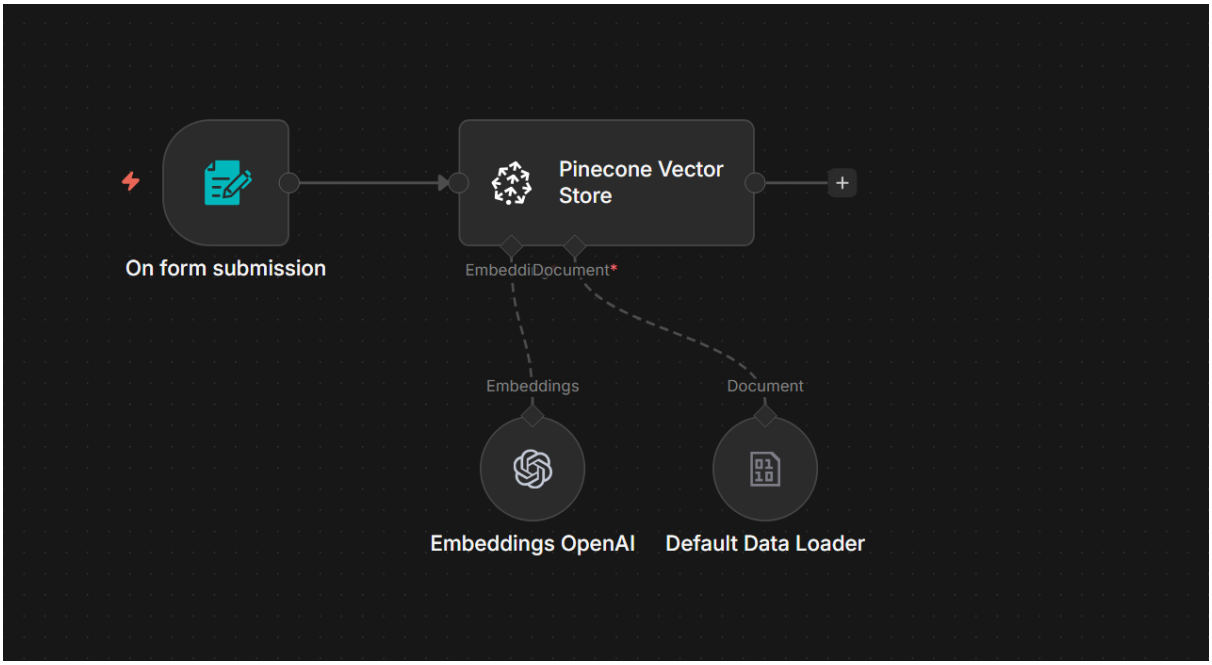
- **Exercise 3-1** completed (working workflow with system prompt)

n8n Nodes

Pipeline 1: Ingest (uploads documents to the vector store)

- **n8n Form** – On form submission
- **Pinecone Vector Store** – Add documents to vector store
  - **Embeddings OpenAI** – generates embeddings
  - **Default Data Loader** – loads the document
    - Type of data: **Binary**
    - Split pages: **Yes**
    - Text splitter: **Recursive Character Text Splitter**
    - Chunk size: **2000**
    - Chunk overlap: **200**

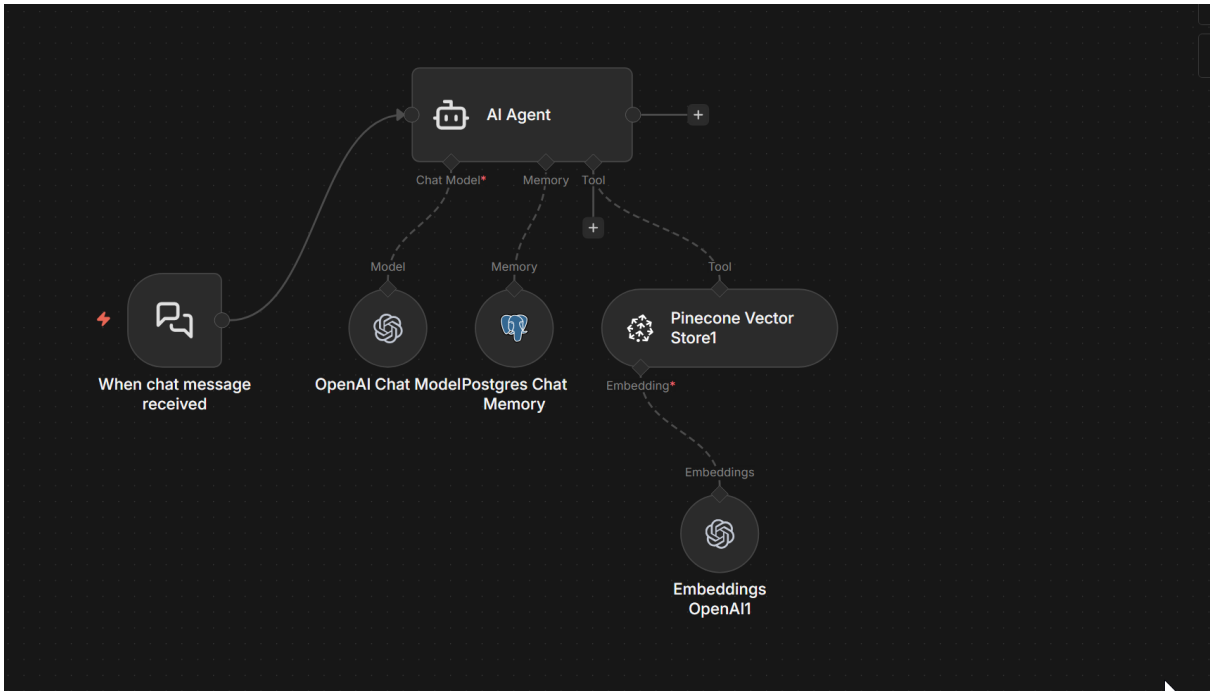
An example of how this pipeline should look like.



Pipeline 2: Chat (retrieves documents during conversation)

- **Chat Trigger**
- **AI Agent**
- **OpenAI Chat Model** (gpt-4.1-mini)
- **Postgres Chat Memory**
- **Pinecone Vector Store** – Retrieve documents for AI Agent as Tool

An example of how this pipeline should look like.



Setup

Pipeline 1: Ingest

1. Create a **new workflow** for ingestion (separate from your chat workflow)
2. Add an **n8n Form** node (trigger: On form submission)
3. Add a **Pinecone Vector Store** node (mode: Add documents to vector store)
4. Connect the Form to the Pinecone Vector Store
5. Add an **Embeddings OpenAI** node and connect it to the Pinecone Vector Store
6. Add a **Default Data Loader** node and connect it to the Pinecone Vector Store
    - Type of data: **Binary**
    - Split pages: **Yes**
    - Text splitter: **Recursive Character Text Splitter**

- Chunk size: **2000**
- Chunk overlap: **200**

Pipeline 2: Chat

1. Return to your **chat workflow**
2. Add a **Pinecone Vector Store** node (mode: Retrieve documents for AI Agent as Tool)
3. Connect it to the **AI Agent** as a tool
4. Add **Embeddings OpenAI** and connect it to this Pinecone Vector Store

Pinecone Configuration (Both Pipelines)

- **Credentials**: Pinecone API Key (provided by instructor)
- **Pinecone Index**: After setting the credentials, choose **From list**, then select **summer** from the dropdown
- **Pinecone Namespace** (Add Option): Use `admin` OR `user<ID>` (as assigned by instructor)

Ingestion Pipeline Only

- **Clean Namespace** (Add Option): Toggle ON – every upload removes old documents first

Retriever Pipeline Only

- **Description**: Changi Airport Group Knowledge Base

Pinecone Hierarchy

In Pinecone, the hierarchy flows from **Organization → Project → Index → Namespace**. An organization contains multiple projects, each project can have multiple indexes, and within each index you can create namespaces to partition your data logically. The vector dimension (size) is defined at the index level when

you create it, meaning all vectors stored in that index must have the same dimensionality, regardless of which namespace they belong to.

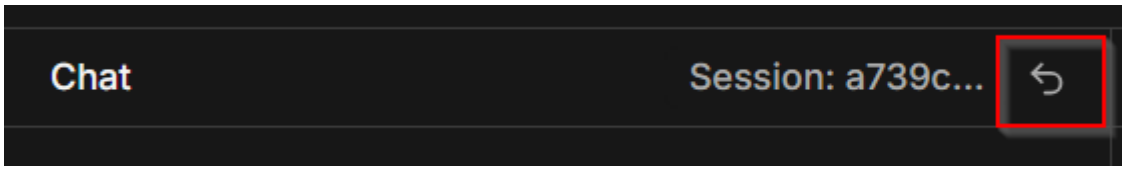
Run Ingestion

1. Activate the ingestion workflow
2. Upload the **CAG Annual Report 2024/25** PDF via the form
3. Wait for ingestion to complete

About the Reset Icon ( )

At this stage, the reset icon clears:

- Chat memory (conversation history)
- Current session context



Your Task

Part 1: Without RAG (Vector Store disconnected)

First, **disconnect** the Pinecone Vector Store tool from the AI Agent (or remove the connection temporarily). Then try these prompts:

Specific data questions (shows RAG value):

- “How many cities was Changi connected to as at 31 March 2025?”
- “What are the newest cities you can fly to from Singapore?”
- “What was CAG’s FY2024/25 revenue?”

Also try these general questions:

- “When was Changi Airport Group founded?”

- “How many terminals are planned in Changi airport?”

Record the bot’s answers.

Part 2: With RAG (Vector Store connected)

Now **reconnect** the Pinecone Vector Store tool to the AI Agent. Reset the chat ( ) and ask the **same prompts** again.

Record the bot’s answers and compare with Part 1.

Part 3: Source Verification

1. Ask: **“How many cities was Changi connected to as at 31 March 2025?”**
2. The bot should answer **170 cities**
3. Verify against the annual report (page 2)
4. Does the answer match what’s in the document?
5. Did the bot retrieve the correct chunk, or could it have pulled from an unrelated section?

Part 4: Stump the Bot

Try questions **NOT** in the knowledge base:

- “How many people eat lunch in Changi?”
- “What is the best hour to fly out of Singapore?”
- “What is the technical architecture of a boarding pass?”
- “How much does a passenger normally have to pay for a flight?”

Observe: Does the bot admit uncertainty, or does it hallucinate an answer?

Part 5: Improve the System Prompt

Add the following **GROUNDING** section to your system prompt (from Exercise 3-1):

```
GROUNDING:

- When answering from retrieved documents, cite the source by providing the
exact quote

from the document as inline text (no links)

- If no relevant information is found, say "I don't have specific information
on this in

the annual report"
```

Test the grounding:

1. Ask: **“Which new restaurants have been opened in Changi?”** – Does it cite the source?
2. Ask something **NOT** in the knowledge base – Does it admit uncertainty?

Discussion Questions

- How would you detect if RAG retrieved the wrong chunk?
- What happens if the answer spans multiple chunks?
- How confident are you in the bot’s source attribution?

What to Submit

1. The bot’s answers **without** RAG vs **with** RAG – what changed?
2. Did the verified answer (170 cities) match the annual report?
3. Did the bot admit uncertainty on the “Stump the Bot” questions before and after adding GROUNDING?
4. Your thoughts on how to detect wrong-chunk retrieval