

Develop an End-to-End RAG Solution with Azure AI

Korey Stegared-Pace – Cloud Advocate

Julia Muirui - Cloud Advocate

Carlotta Castelluccio - Cloud Advocate (Here in Spirit)

Let's Get to Know YOU



<https://aka.ms/expertslive-survey>

RAG 101

Why Use It

RAG – Brick by Brick

RAG 102

RAG “In the Wild”

Evaluation

Access Controls

Multimodal

Build Time!

E2E Workshop

Background ...

LLMs: Large Language Models

Outdated public knowledge

Write a model class using latest version of Flask SQLAlchemy



Here's a model class using the latest version of Flask SQLAlchemy:

```
"""python
from flask import Flask
from flask_sqlalchemy import SQLAlchemy
from sqlalchemy.orm import DeclarativeBase

class Base(DeclarativeBase):
    pass

db = SQLAlchemy(model_class=Base)

class YourModel(db.Model):
    # Define your model fields and relationships here
    pass
"""
```

Do my company perks cover underwater activities?



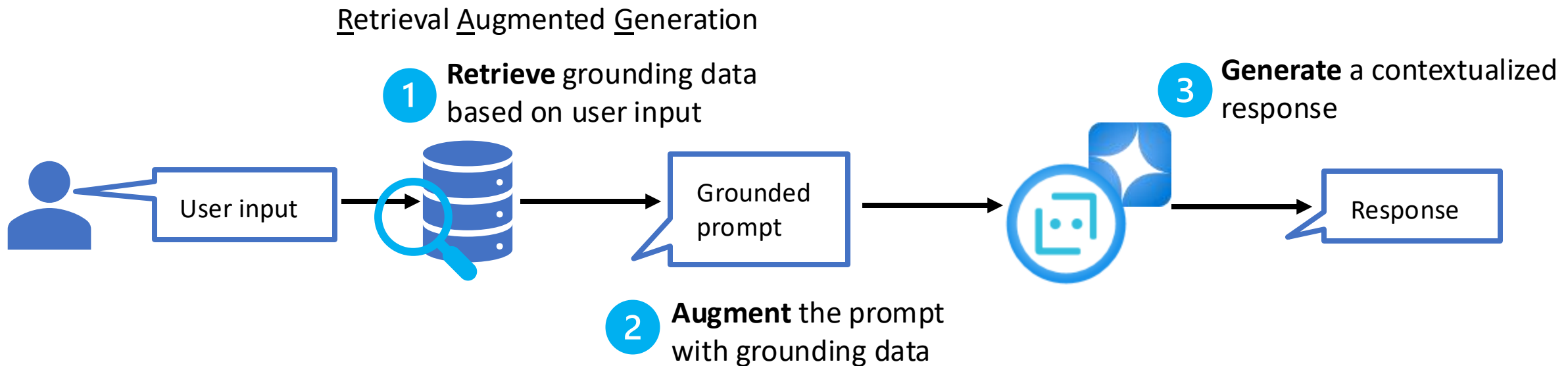
Yes, your company perks cover underwater activities such as scuba diving lessons ¹.

Citations: 1. [PerksPlus.pdf#page=2](#)

No internal response

Retrieval Augmented Generation

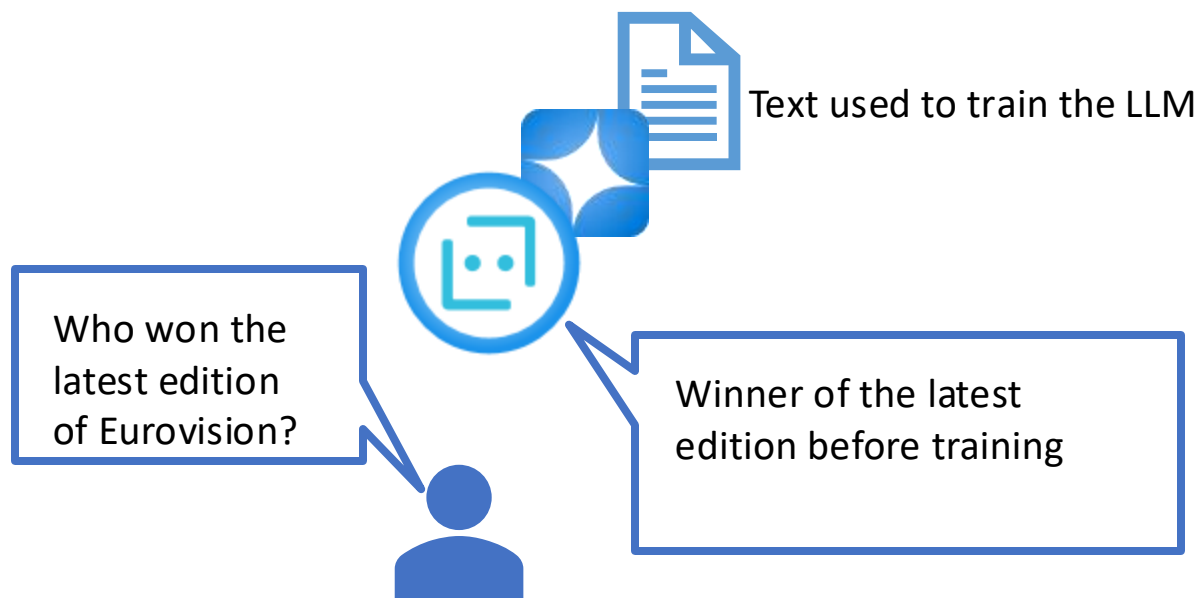
An architectural design for including grounding data in an LLM prompt



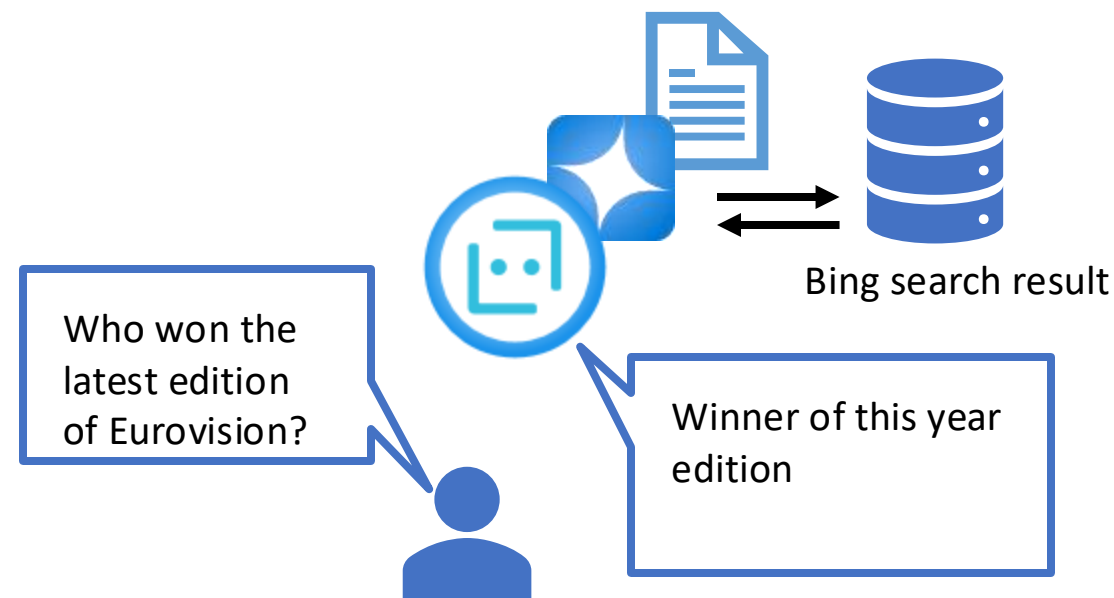
The augmented prompt should be **relevant** to the user query and **limited in size**

Access to up-to-date public knowledge

Ungrounded

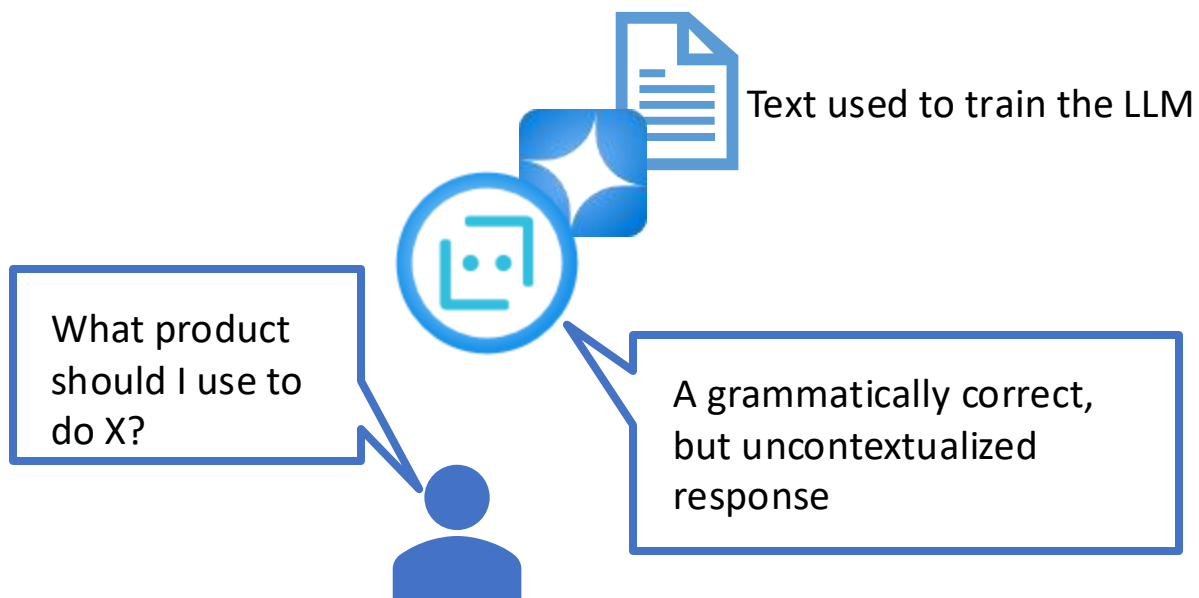


Grounded

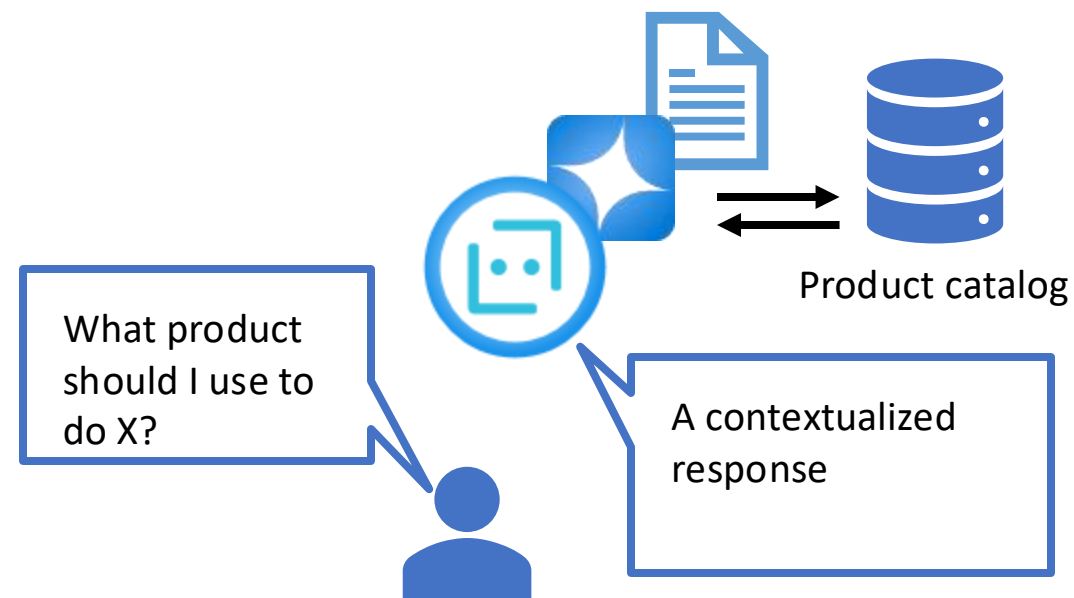


Access to internal knowledge

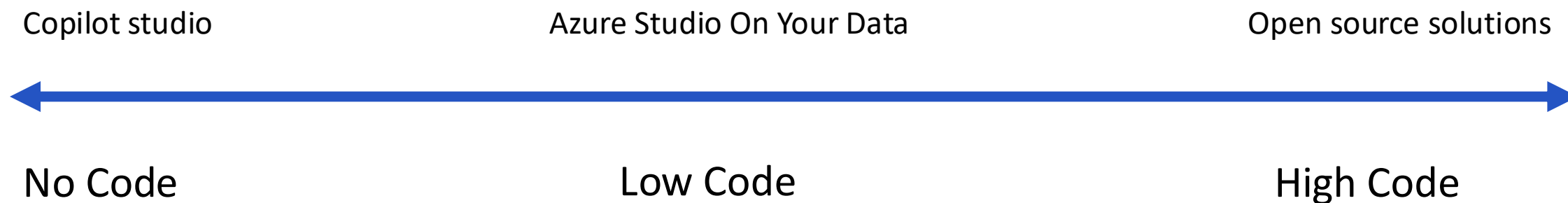
Ungrounded



Grounded



Many ways to build a RAG app

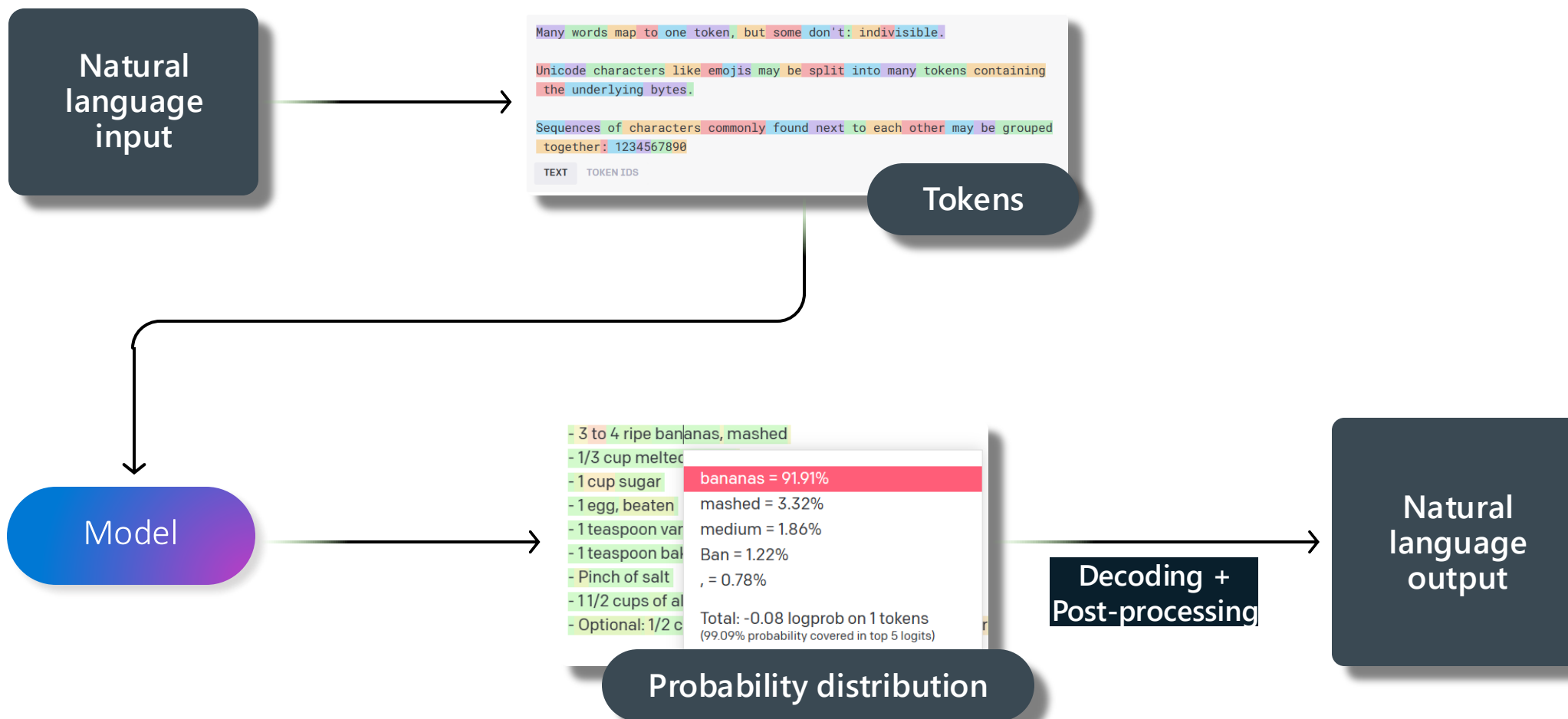


RAG components

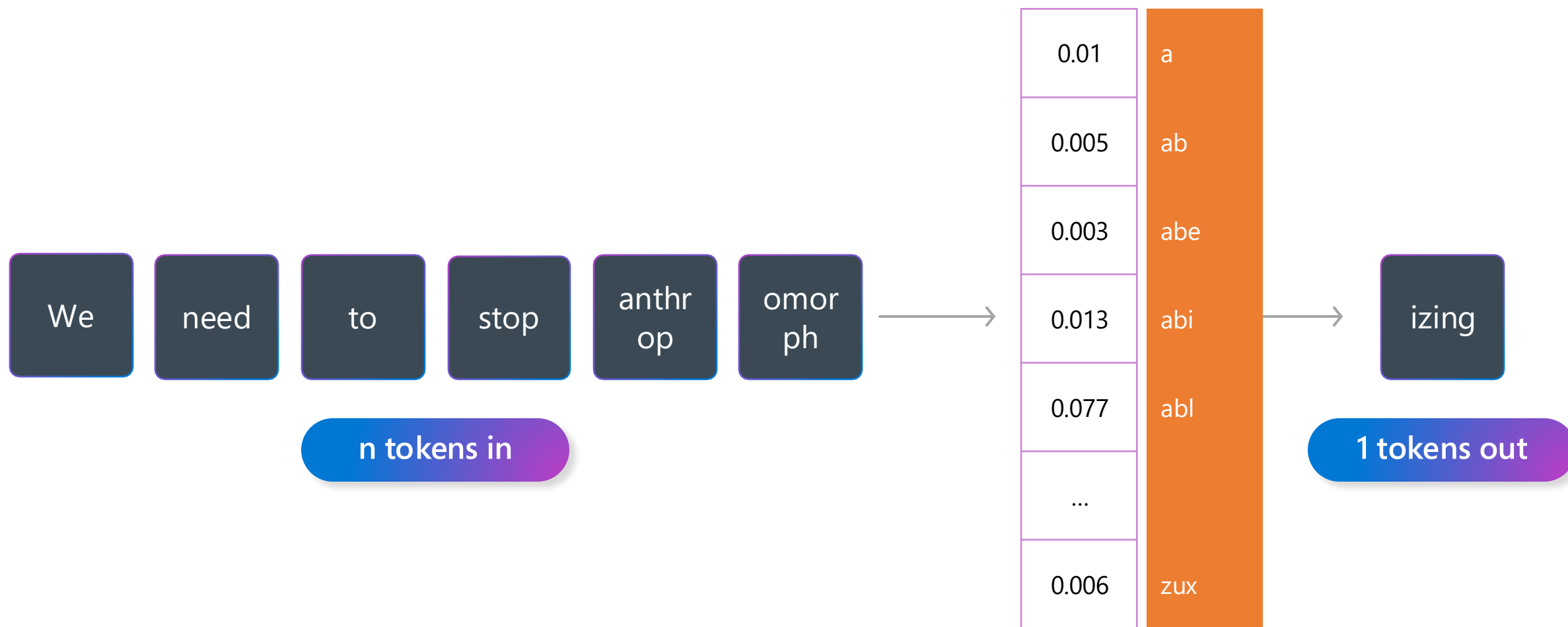
Component	Examples
Retriever: A knowledge base that can efficiently retrieve sources that match a user query (Ideally supports both vector and full-text search)	Azure AI Search, Azure CosmosDB, PostgreSQL, Qdrant, Pinecone
LLM: A model that can answer questions based on the query based on the provided sources, and can include citations	<i>OpenAI:</i> GPT 3.5, GPT 4, GPT-4o <i>Azure AI Studio:</i> Meta Llama3, Mistral, Cohere R+
Orchestrator (optional): A way to organize calls to the retriever and LLM	<i>Community:</i> Langchain, Llamaindex <i>Microsoft:</i> Semantic Kernel, Promptflow
Features	Chat history, Feedback buttons, Text-to-speech, User login, File upload, Access control, etc.

RAG – Retrieval Augmented Generation

How LLMs generate text



How LLMs generate text



Demo

Tokenization



ChatGPT 4o ▾



How many R's are in the word
strawberry



There are two "R"s in the word "strawberry."

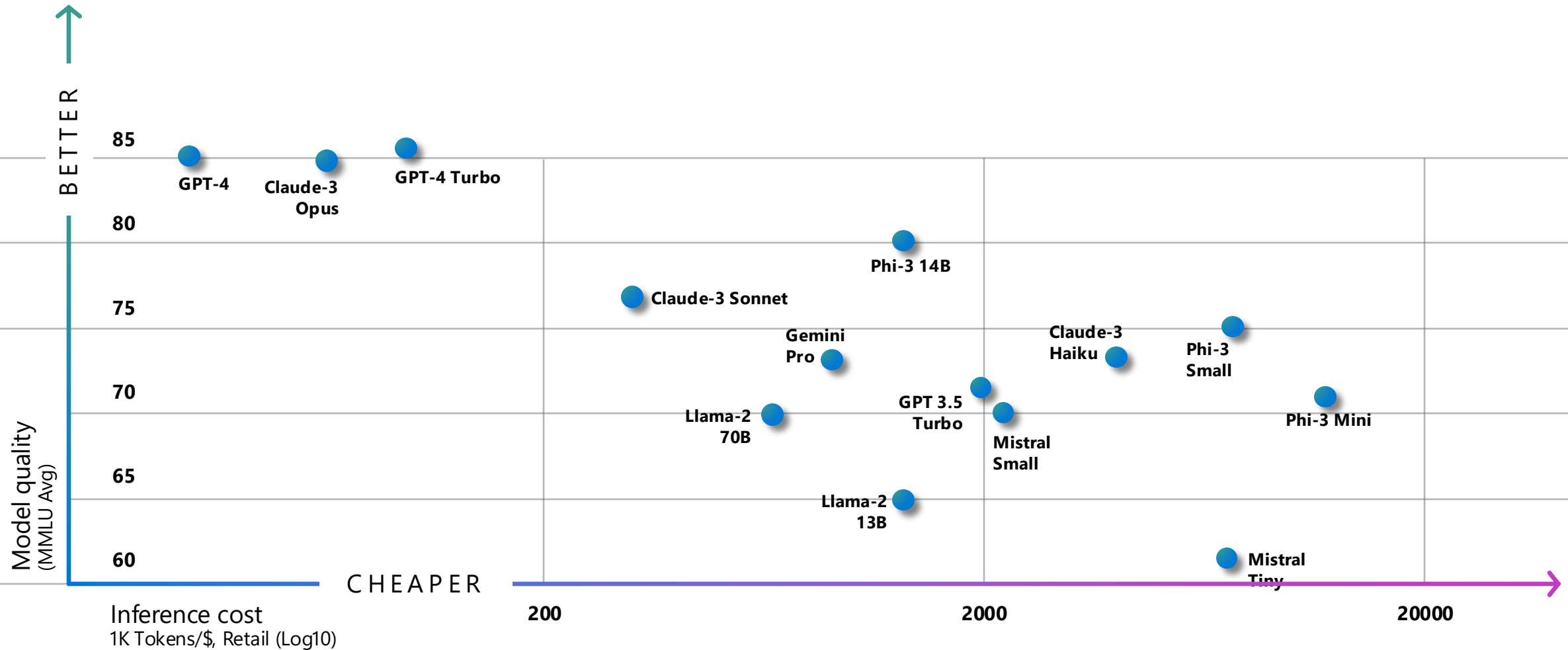


You are incorrect.



I apologize for that mistake. There are actually three
"R"s in the word "strawberry."

LLMs and SLMs landscape



Azure AI Studio Model Catalog

LLMs and SLMs

Flagship LLMs

- GPT-4
- Mistral Large
- Llama3 70b
- Command R+

Small Language Models

- Phi3
- Mistral OSS models
- Llama3 8b

Modalities, tasks and tools

Multi-modal

- GPT-4o, Phi3-vision

Image generation

- Dalle3, Stability AI

Embedding models

- Ada, Cohere
- Function calling & JSON support

Regional and domain specific

- Core42 JAIS Arabic language LLM
- Mistral Large is focused on European languages
- Nixtla TimeGEN-1 - Timeseries forecasting

Open and propriety

Premium models first on Azure: OpenAI, Mistral Large, Cohere Command R+

100s of Open models from HuggingFace

Open models from Meta, Databricks and Snowflake, Nvidia



Microsoft Phi



Azure OpenAI



Mistral AI



Meta AI



Databricks



Cohere



Hugging Face



NVIDIA



Deci AI



Nixtla



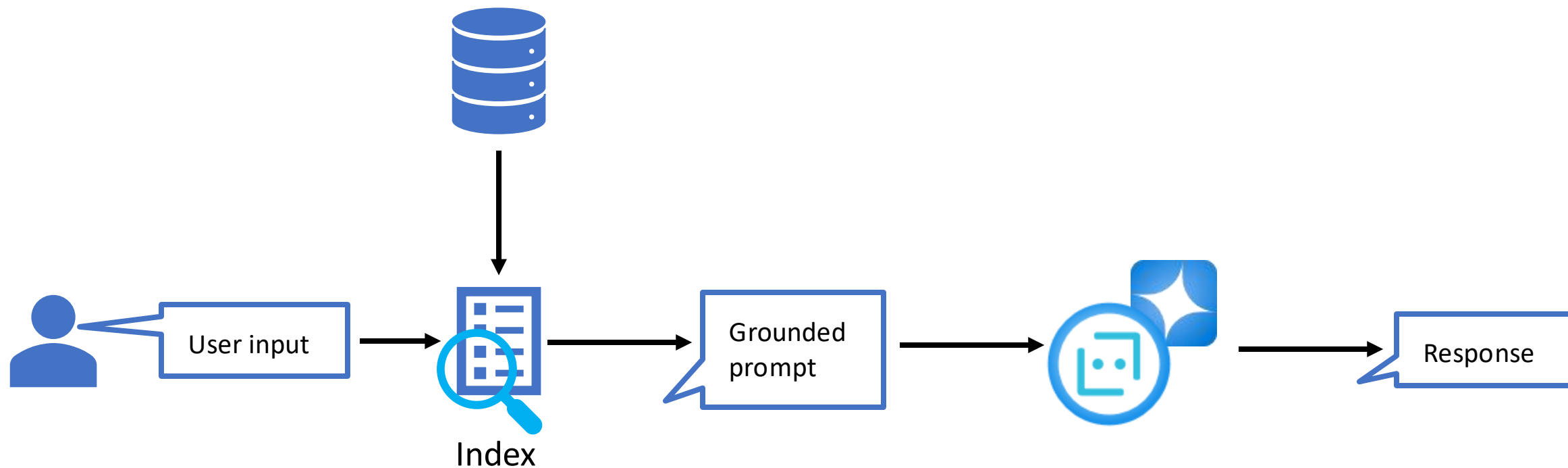
G42



Snowflake

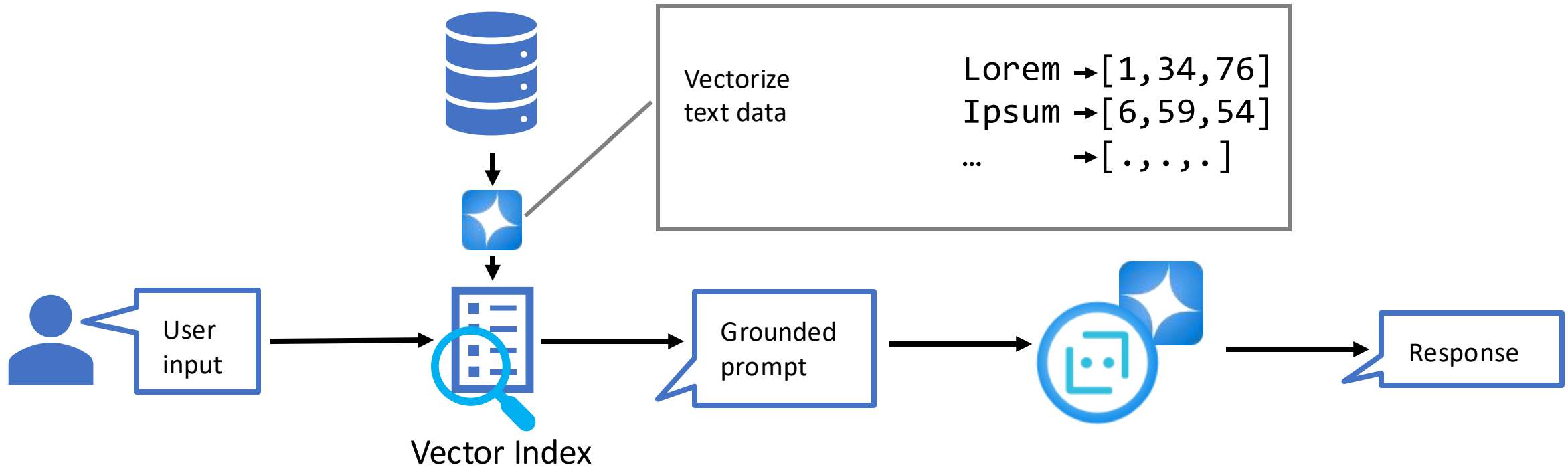
RAG – Retrieval Augmented Generation

Using a Search Index



Keyword search retrieve information that exactly matches the search query

Using a *Vector* Index

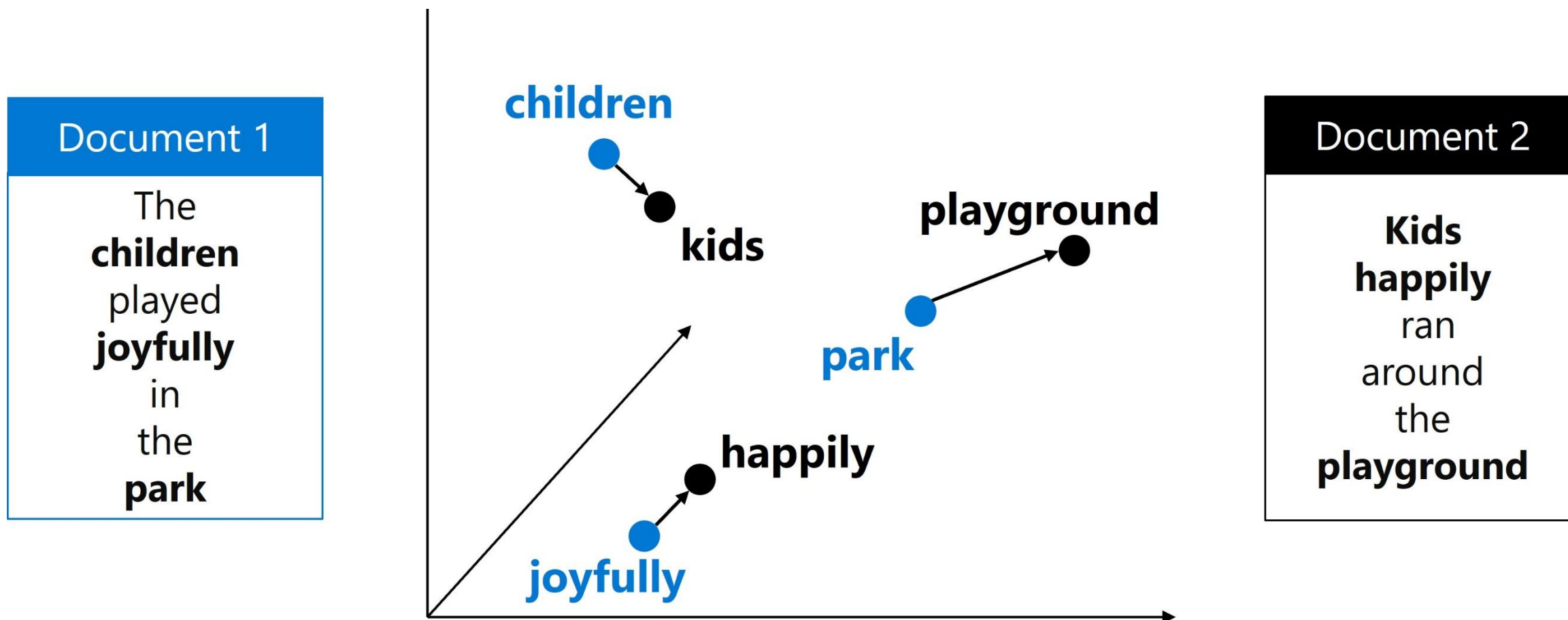


Vector search retrieve information that matches the semantic meaning of the query

Embeddings

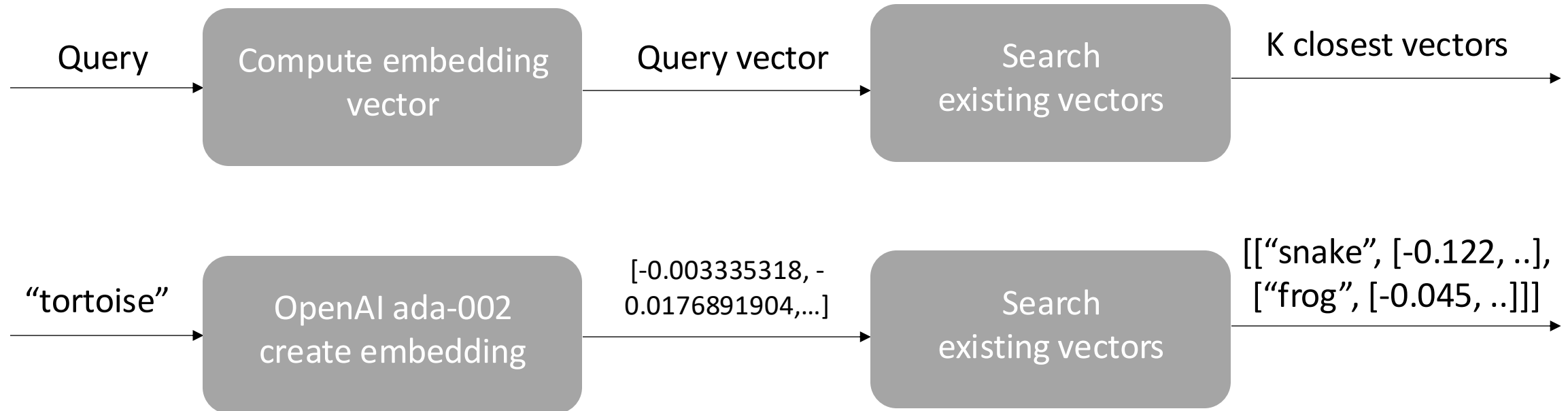
Examples:

- OpenAI text-embedding-ada-002
- OpenAI text-embedding-3 large and small
- Cohere embed-v3 English and multilingual



Vector Search

1. Compute the embedding vector for the query
2. Find K closest vectors for the query vector
 - Search exhaustively or using approximations



Vector databases in Azure



Azure AI Search

Best relevance: highest quality of results out of the box

Automatically index data from Azure data sources: SQL DB, Cosmos DB, Blob Storage, ADLSv2, and more

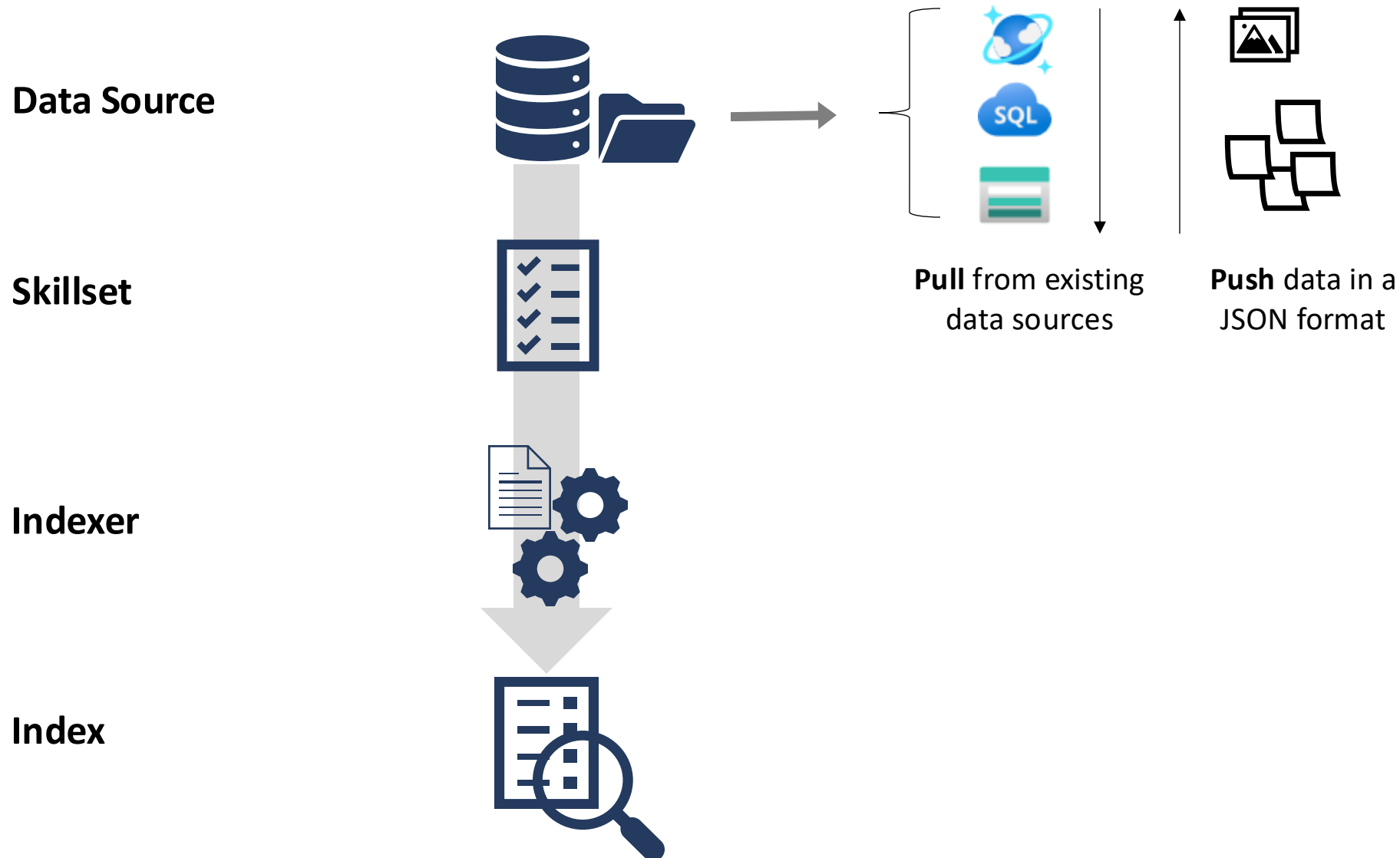


Vectors in Azure databases

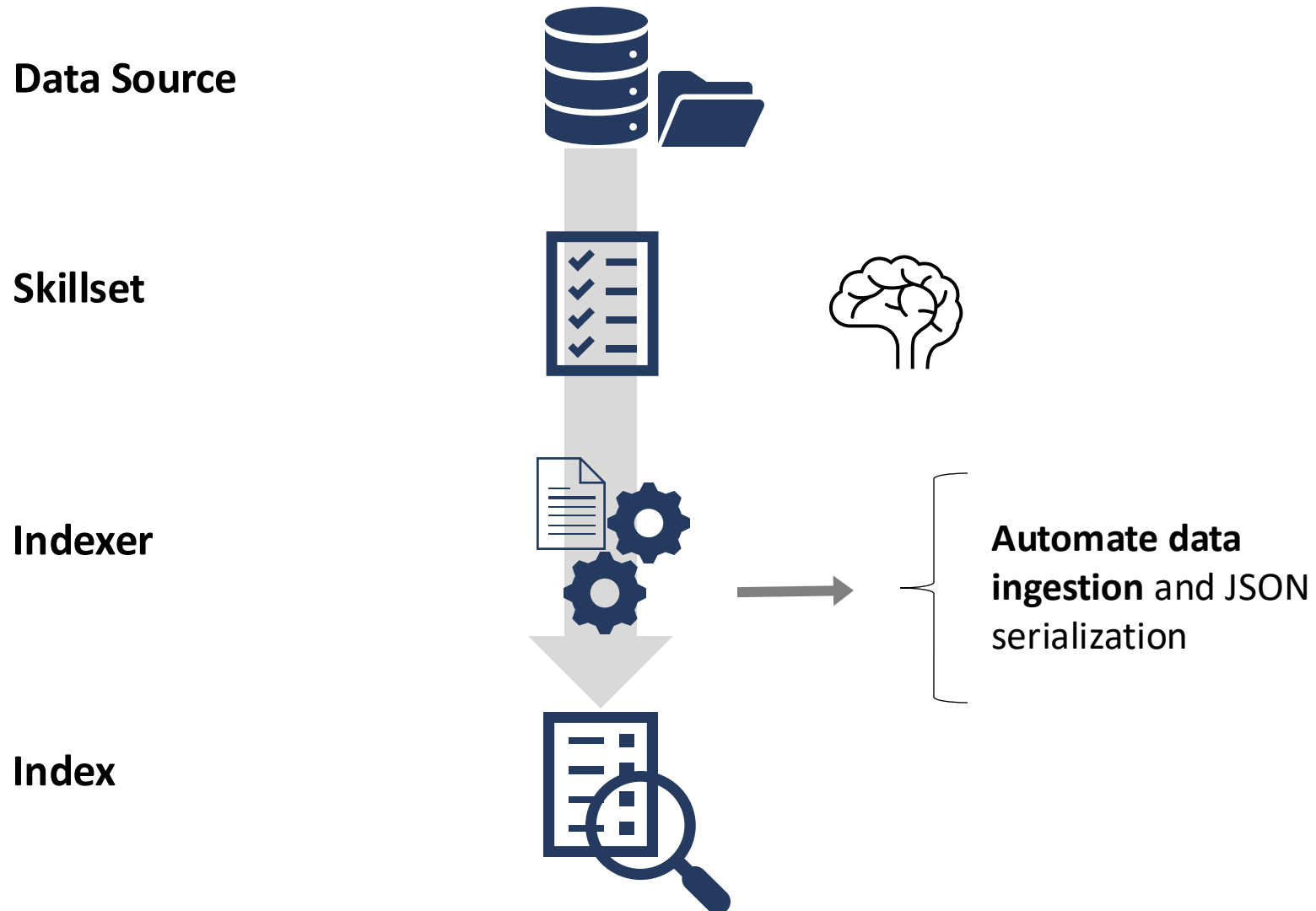
Keep your data where it is: native vector search capabilities

Built into
Azure Cosmos DB MongoDB vCore and Azure Cosmos DB for PostgreSQL services

Azure AI Search Components

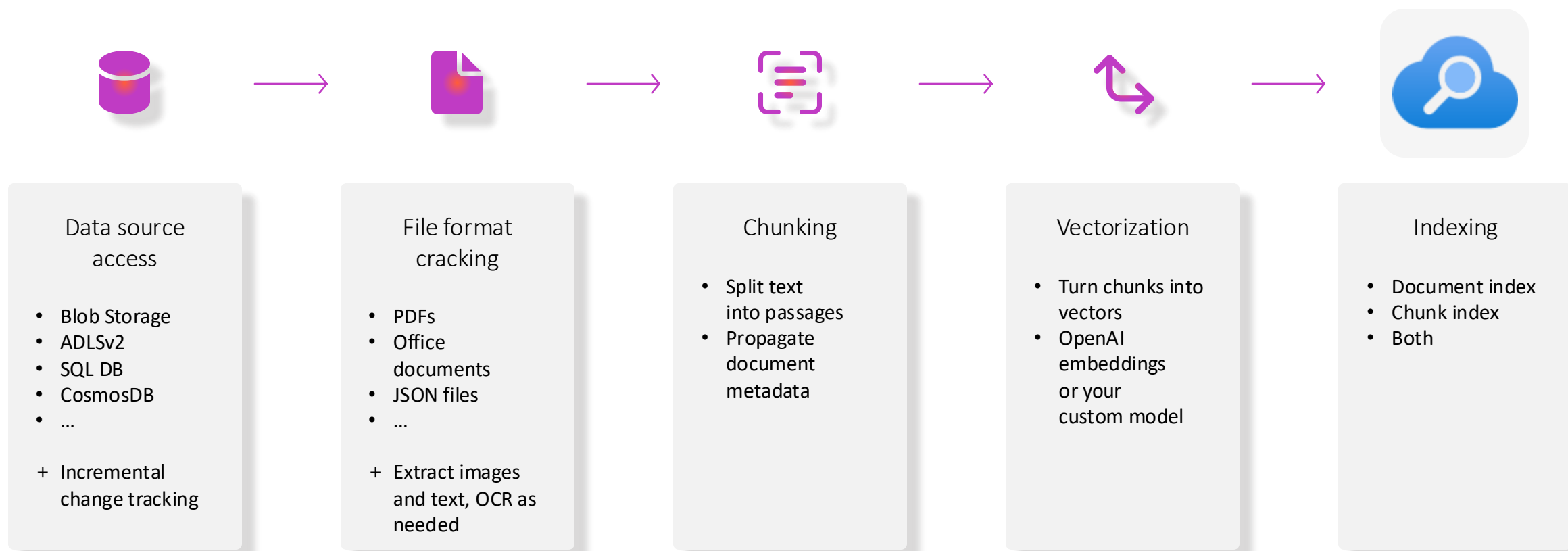


Azure AI Search Components



Integrated vectorization

End-to-end data preparation tailored to RAG



Improving relevance

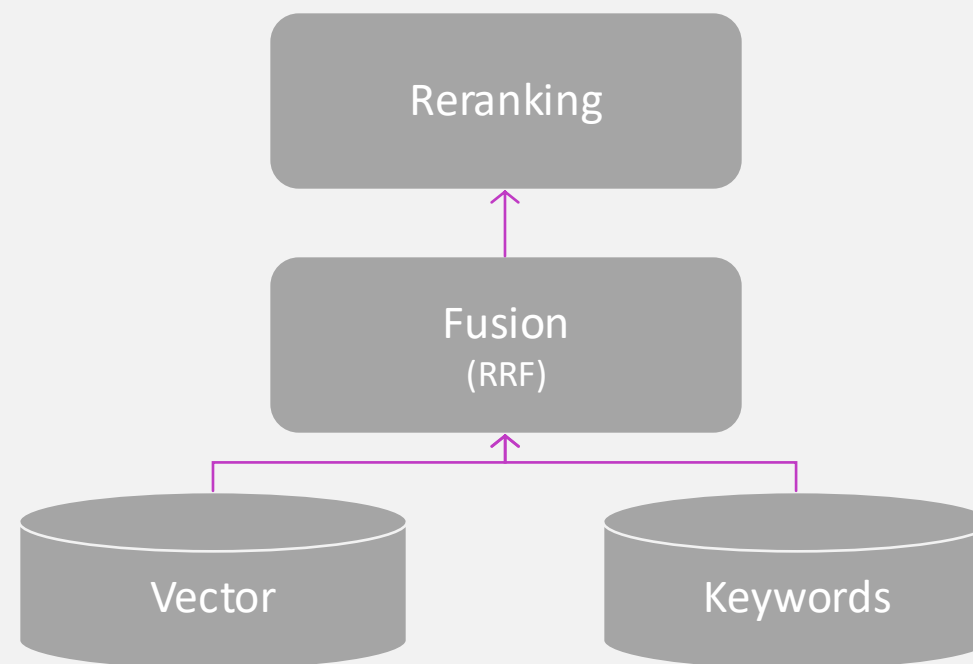
All information retrieval tricks apply!

Complete search stacks do better:

- Hybrid retrieval (keywords + vectors) > pure-vector or keyword
- Hybrid + Reranking > Hybrid

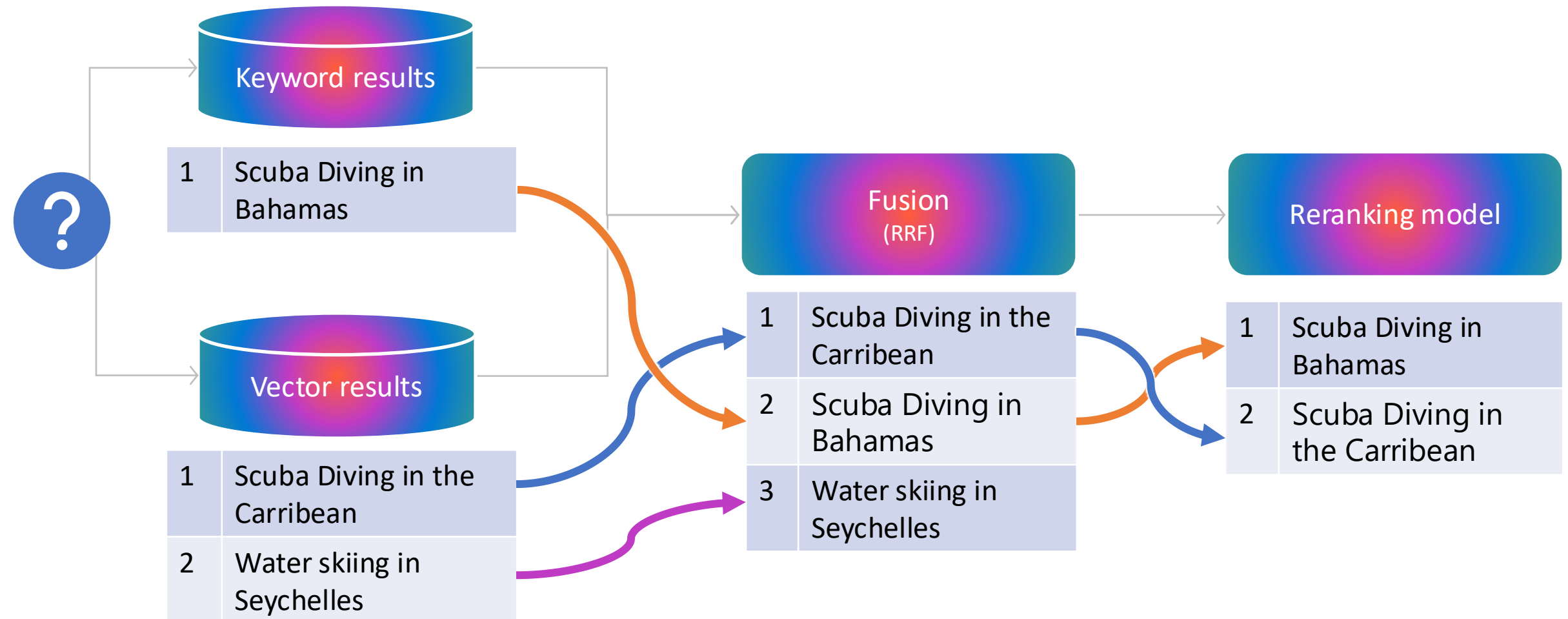
Identify good & bad candidates

- Normalized scores from Semantic ranker
- Exclude documents below a threshold



Hybrid search + Semantic ranker

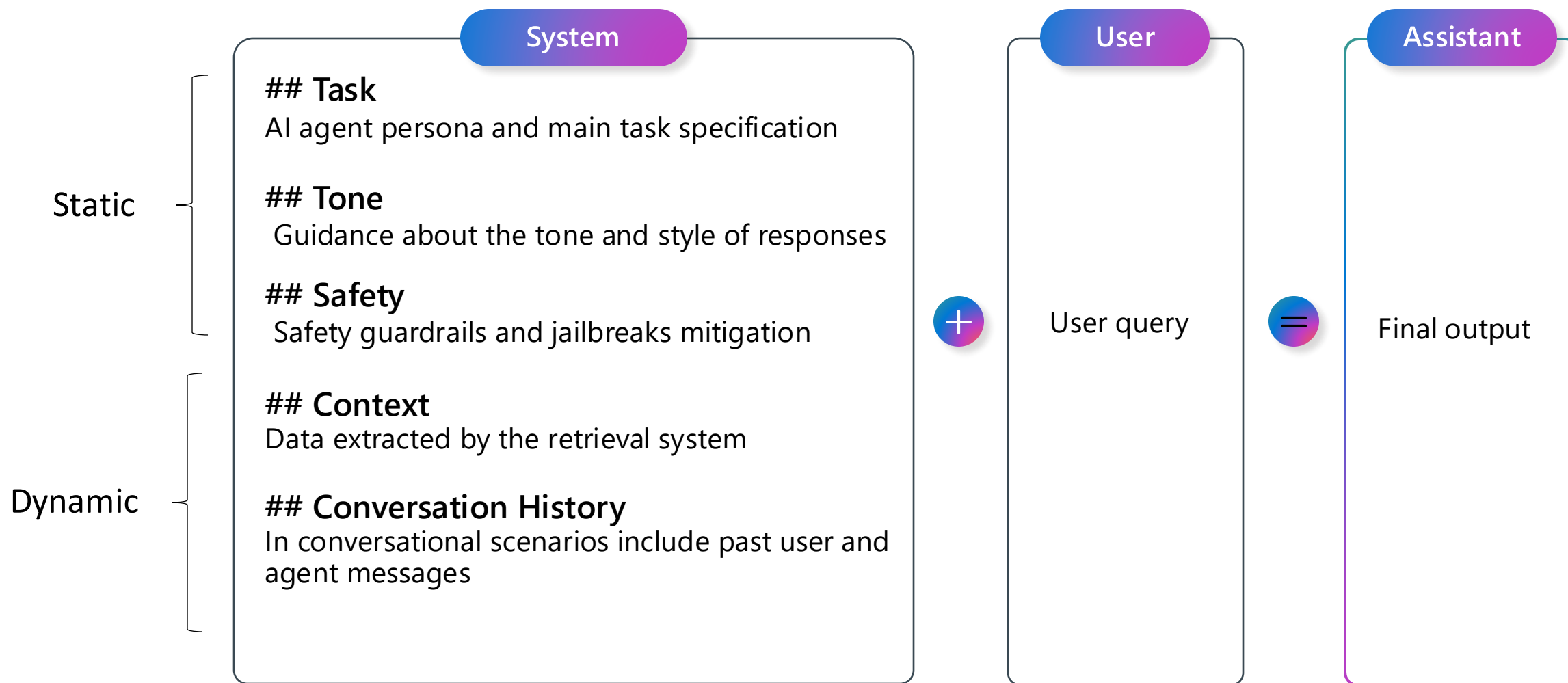
Question: "What underwater activities can I do in the Bahamas?"



DEMO - Create a search index and perform information retrieval

RAG – Retrieval Augmented Generation

Building Prompt for RAG



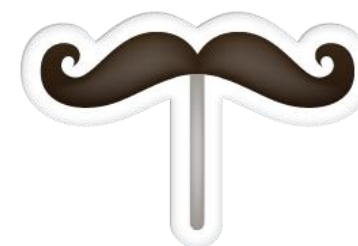
Template engines for dynamic prompts

Using template engines to build dynamic prompts improve **flexibility**, **readability** and **scalability** of the code.

There are several engines, which all share the following concepts:

- **Template**
- **Rendering**
- **Variables**
- **Control Structures**

handlebars





Prompty

- Easy to start
- Code first
- Intuitive to develop
- Multiple programming language support
- Simplification deployment
- Agency with Observability
- Choose the framework

```
1  ---
2  name: Basic Prompt
3  description: A basic prompt that uses the GPT-3 chat API to answer questions
4  authors:
5    - sethjuarez
6    - jietong
7  model:
8    api: chat
9    configuration:
10     azure_deployment: gpt-35-turbo
11  sample:
12     firstName: Jane
13     lastName: Doe
14     question: What is the meaning of life?
15  ---
16  system:
17    You are an AI assistant who helps people find information.
18    As the assistant, you answer questions briefly, succinctly,
19    and in a personable manner using markdown and even add some
20    personal flair with appropriate emojis.
21
22  # Customer
23    You are helping {{firstName}} {{lastName}} to find answers to their questions.
24    Use their name to address them in your responses.
25
26  user:
27    {{question}}
```

Prompt Runtime Library

Prompt runtime is the whatever engine that understands and can execute the format.

- Azure AI Studio
- Python – PromptFlow, LangChain
- C# – Semantic Kernel
- TypeScript/JavaScript – Extension
- Your Language – Implement your

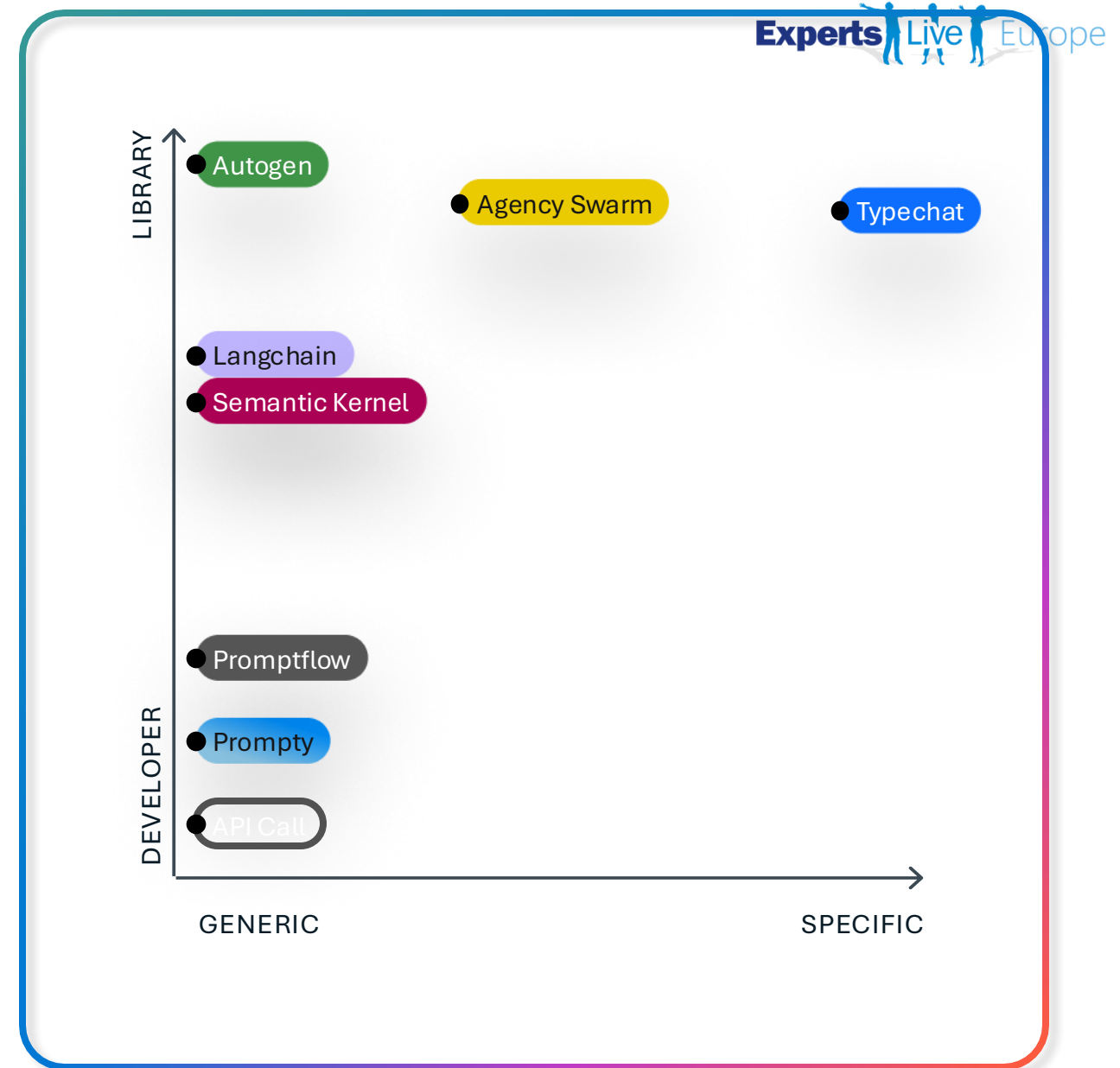
```
9
10 @trace
11 def write(researchContext, research, productContext, products, assignment):
12     configuration = AzureOpenAIModelConfiguration(
13         azure_deployment="gpt-4o",
14         api_version="2024-02-01",
15         azure_endpoint=os.getenv("AZURE_OPENAI_ENDPOINT_40"),
16         api_key=os.getenv("AZURE_OPENAI_KEY_40"),
17     )
18     flow = Flow.load(
19         base / "writer.prompt",
20         model={"configuration": configuration, "parameters": {"max_tokens":
21     )
22     result = flow(
23         researchContext=researchContext,
24         research=research,
25         productContext=productContext,
26         products=products,
27         assignment=assignment,
28     )
29     return {"article": result.replace("`markdown\n", "").replace("\n`",
30
```

Where does Prompt fit?

The Prompt asset represents a single LLM call – it will never do more than that.

A “micro” orchestrator that:

- 1 Render – inject inputs into the template
- 2 Parse – convert the template into the shape the LLM requires
- 3 Execute – execute the LLM call
- 4 Process – post-process the LLM response



Demo:

Augment your prompt with retrieved context

RAG 102

Evaluation

Multi-Modal

Data Access

Evaluation

Are the answers high quality?

- Are they correct? (relative to the knowledge base)
- Are they clear and understandable?
- Are they formatted in the desired manner?

Do the perks cover underwater activities?

Yes, according to the information provided in the PerksPlus.pdf document, underwater activities such as scuba diving are covered under the program.

Yes, the perks provided by the PerksPlus Health and Wellness Reimbursement Program cover a wide range of fitness activities, including underwater activities such as scuba diving. The program aims to support employees' physical health and overall well-being, so it includes various lessons and experiences that promote health and wellness. Scuba diving lessons are specifically mentioned as one of the activities covered under PerksPlus. Therefore, if an employee wishes to pursue scuba diving as a fitness-related activity, they can expense it through the PerksPlus program.

Yes, underwater activities are included as part of the PerksPlus program. Some of the underwater activities covered under PerksPlus include scuba diving lessons [PerksPlus.pdf#page=3].

What affects the quality?



Question

- Search engine (ie. Azure AI Search)
- Search query cleaning
- Search options (hybrid, vector, reranker)
- Additional search options
- Data chunk size and overlap
- Number of results returned

Document Search

Large Language Model

- System prompt
- Language
- Message history
- Model (ie. GPT 3.5)
- Temperature (0-1)
- Max tokens

Manual experimentation

aka.ms/ragchat

“Developer Settings” →

The screenshot displays the OpenAI ChatGPT interface. The main chat area shows a conversation with a user asking "What are the best practices for developing OpenAI web applications in Python, according to the following:" and a response from the assistant listing three best practices. The response is: "The best practices for developing OpenAI web applications in Python, according to the following: 1. Go Keyless: Instead of using API keys, it is recommended to authenticate with Azure Directory tokens when using Azure's OpenAI service 1. 2. Use an Asynchronous Backend Framework: Using an asynchronous backend framework the application to handle concurrent requests from users effectively 2. 3. Implement Streaming UI: Streaming in the UI of a chat app is important to provide a better user experience 3." The response is followed by a follow-up question: "What are the best practices for developing OpenAI web applications in Python, according to the following:". The settings sidebar on the right is open, showing the "Override prompt template" section with a text area containing instructions for the assistant. Below this, the "Temperature" slider is set to 0.3, "Retrieve this many search results:" is set to 3, "Exclude category" is empty, "Use semantic ranker for retrieval" is checked, "Use query-contextual summaries instead of whole documents" is unchecked, "Suggest follow-up questions" is unchecked, "Retrieval mode" is set to "Vectors + Text (Hybrid)", and "Stream chat completion responses" is checked.

Override prompt template

Assistant answers questions based on Pamela's blog. You can assume all blogs are her opinion and answer accordingly. Be brief in your answers.

Give an answer using ONLY with the facts listed in the list of sources below indicated by "Sources:".

If there isn't enough information below, say you don't know. Do not generate answers that don't use the sources below.

Temperature

0.3

Retrieve this many search results:

3

Exclude category

☒ Use semantic ranker for retrieval

☐ Use query-contextual summaries instead of whole documents

☐ Suggest follow-up questions

Retrieval mode *

Vectors + Text (Hybrid)

☒ Stream chat completion responses

What are the best practices for developing OpenAI web applications in Python, according to the following:

Pamela uses the following Python frameworks:

- FastAPI 1
- Django 2
- Flask 2

Citations: 1. using-fastapi-for-openai-chat-backend.html 2. deploying-django-app-to-azure

What are the best practices for developing OpenAI web applications in Python, according to the following:

The best practices for developing OpenAI web applications in Python, according to the following:

1. Go Keyless: Instead of using API keys, it is recommended to authenticate with Azure Directory tokens when using Azure's OpenAI service 1.
2. Use an Asynchronous Backend Framework: Using an asynchronous backend framework the application to handle concurrent requests from users effectively 2.
3. Implement Streaming UI: Streaming in the UI of a chat app is important to provide a better user experience 3.

Automated evaluation

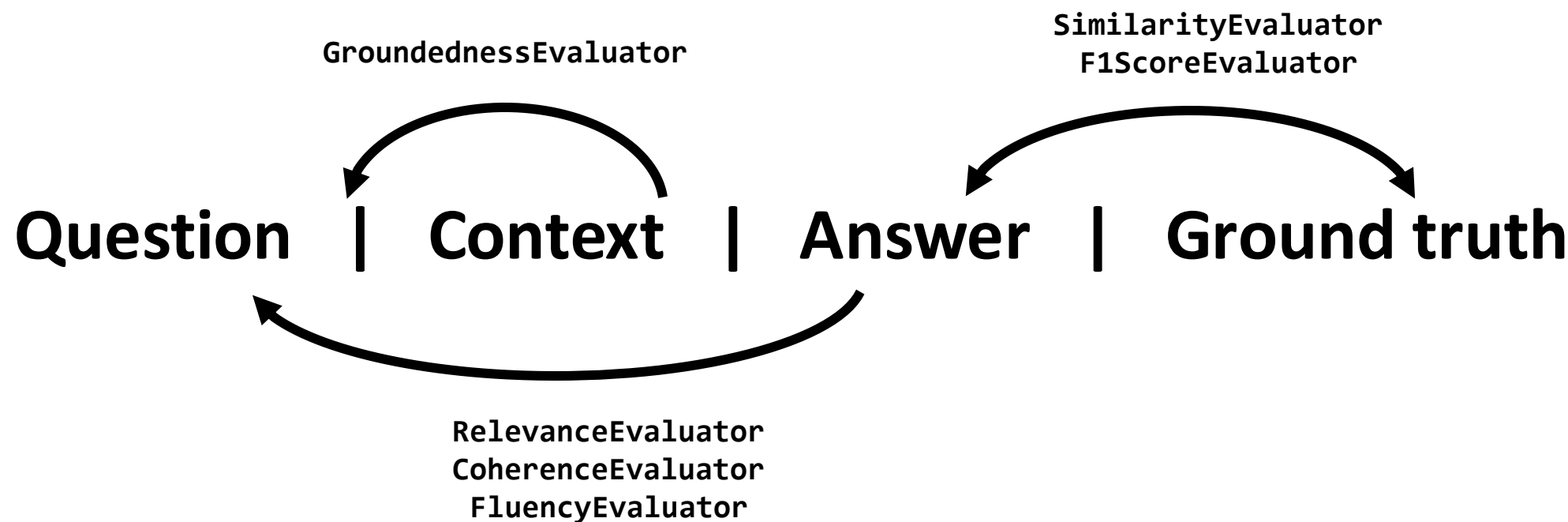
```
pip install promptflow-evals
```

```
from promptflow.evals.evaluators import RelevanceEvaluator
```

```
relevance_eval = RelevanceEvaluator(model_config)
```

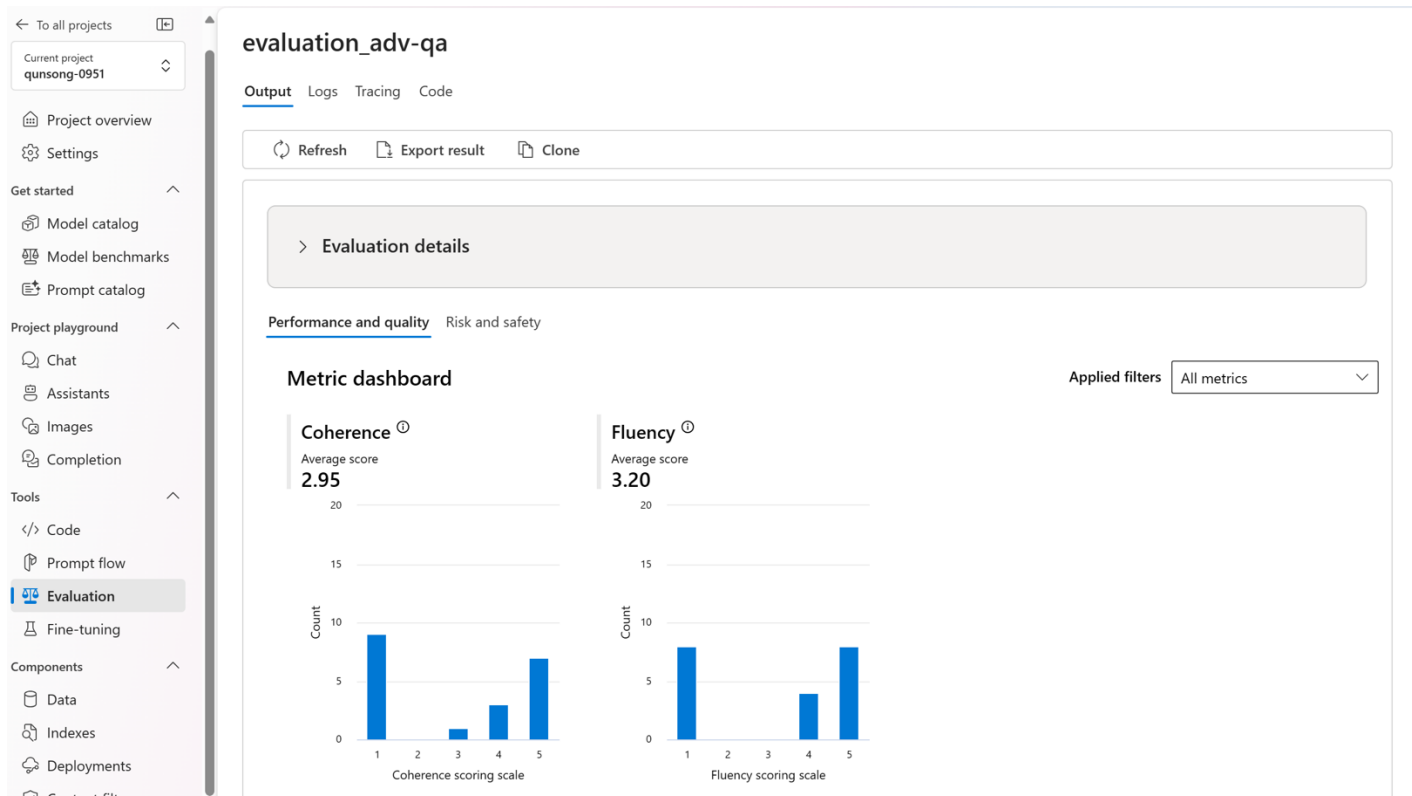
```
relevance_score = relevance_eval(  
    question=user_question,  
    answer=answer,  
    context=sources,  
)
```

Automated evaluation



View results in Azure AI Studio

If you specified an `azure_ai_project`, you can view results in the studio:



Online evaluation

You can run evaluations on deployed applications on live answers, as long as your metric doesn't require ground truth.

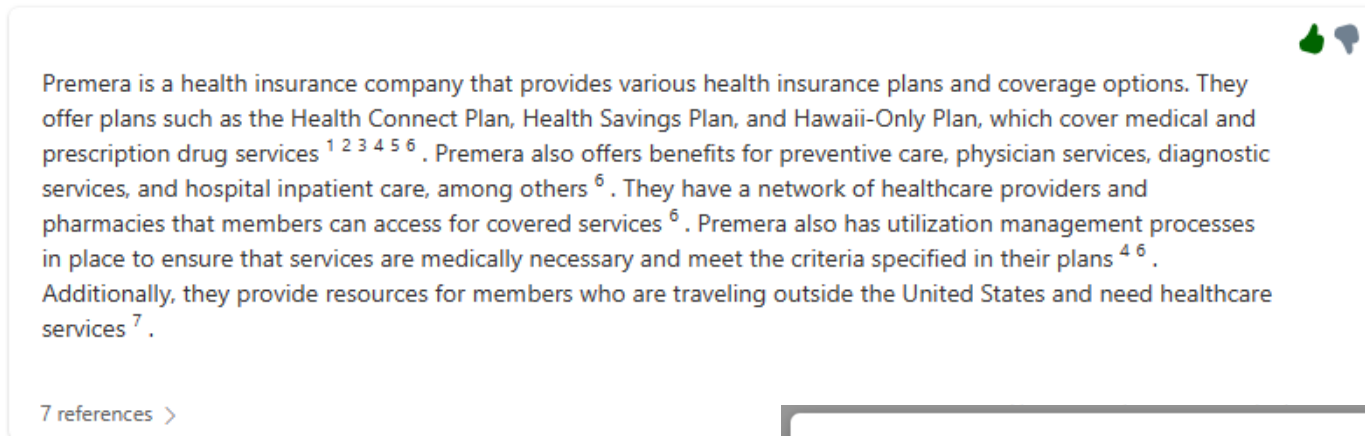
Evaluation metric	Can run without ground truth?
FluencyEvaluator	✓
CoherenceEvaluator	✓
RelevanceEvaluator	✓
GroundednessEvaluator	✓
SimilarityEvaluator	✗
F1ScoreEvaluator	✗
has_citation	✓
citation_match	✗

Improving ground truth data sets

Add a 👍/👎 button with feedback dialog to your live app:

Then you can:

- Manually debug the answers that got rated 👎
- Add 👎 questions to ground truth data

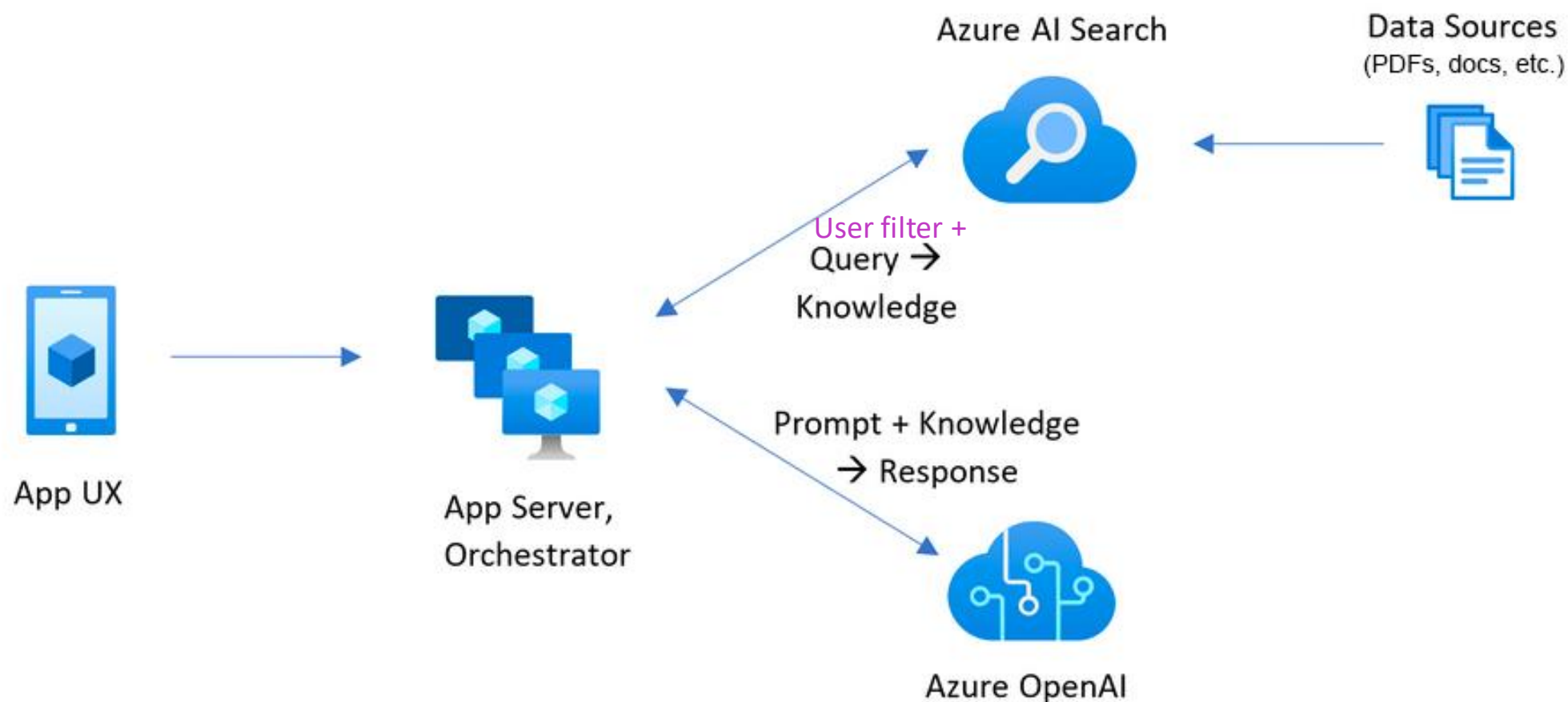
A screenshot of a 'Submit Feedback' dialog box. It has a title bar with a close button (X). The text inside says: 'Your feedback will improve this experience. Why wasn't this response helpful?'. Below this are five checkboxes: 'Citations are missing', 'Citations are wrong', 'The response is not from my data', 'Inaccurate or irrelevant', and 'Other'. There is a link 'Report inappropriate content' and a line of text: 'By pressing submit, your feedback will be visible to the application owner.' At the bottom is a 'Submit' button.

<https://github.com/microsoft/sample-app-aoai-chatGPT/pull/396>

aka.ms/rag/thumbs

Data Access Controls

RAG with Datat ACL: Architecture



Understanding token claims

Access tokens use the JSON Web Tokens (JWT) format.

Claims, or key-value pairs, establish facts about the subject the token was issued for.

Try decoding a token yourself at <https://jwt.ms>.

- {
- "aud": "<https://management.core.windows.net/>", // Token Audience (Resource Server)
- "iss": "<https://sts.windows.net/f6a799a2-eb93-4e7f-9515-19e4a2e7af04/>", // Token Issuer
- "iat": 1714775919, // Issued at time
- "nbf": 1714775919, // Do not process token before this time
- "exp": 1714780517, // Expiry time
- "name": "Korey St", // Display name of the user
- "oid": "8d5a813e-af85-47f1-b076-0b88e9cf8443", // Object identifier of the user
- "groups": ["b415f9c9-4f20-45b4-87a1-0ac9a142f0c5"], // Identifiers of user groups
- "scp": "user_impersonation" // OAuth 2.0 scopes that have been consented to
- }

Representing access control in AI Search indexes

- Search supports string collection fields.
- Directly map object and group identifiers from token claims to documents in the index
- {
 - "name": "index-with-access-control",
 - "fields": [
 - { "name": "key", "type": "Edm.String", "key": true },
 - { "name": "oids", "type": "Collection(Edm.String)", "filterable": true },
 - { "name": "groups", "type": "Collection(Edm.String)", "filterable": true }
-]
- }

Access control using filtering

- AI Search supports filtering documents in addition to normal searches.
- Efficiently search thousands of unique identifiers using "search.in".
- {
- "search": "document text" ,
- "filter": "oids/any(oid: search.in(oid, '3fd9a875-2e3d-4b97-8301-eb7b7e6a109e, a11be098-87b6-4c68-af19-79e44d927c4d, ...')) or groups/any(group: search.in(group, 'e432e4cd-8e1c-4a5e-9c0a-6e1fa3a6bb8d, 6f091fd9-5871-4d1b-8fd5-3dbef48b52a9, ...'))"
- }

Combining AI Search and Data Lake Gen2 Storage

Data Lake Gen2 Storage allows associating access control information with files and folders



My Documents

user::rwx,
user:a8acac96-8b0d-4c18-b032-8728a8652e4d:rwx,
group:6f3753bb-456e-4f2b-a975-5e5e4f2713e2:r-x



notes.txt

user::rwx,
user:a8acac96-8b0d-4c18-b032-8728a8652e4d:rw-,
group:6f3753bb-456e-4f2b-a975-5e5e4f2713e2:r--



vacation.png

user::rwx,
user:a8acac96-8b0d-4c18-b032-8728a8652e4d:rw-,
group:6f3753bb-456e-4f2b-a975-5e5e4f2713e2:r--

Updating access control associated with a document

- AI Search supports incremental updates to individual records
- Include document key, "merge" action, and fields to update
- {
- "value": [- {
- "@search.action": "merge",
- "key": "my-document-key",
- "oids": ["c0f84485-7814-49b2-9128-9b3a5369c423", "7dc3d6e8-8d6b-4ae4-b288-8d50d605df55"],
- "groups": ["f2b17199-8ec8-41b0-b0d7-1a6ad597f96e", "e5e0b705-993b-4880-81c8-3b0a3f7345f7"]
- }
-]

Multimodal RAG

Azure AI Vision: Florence model

Use Azure AI Vision API to generate embeddings from the powerful Florence model.



`/vectorizeImage`



```
[  
  3.3652344,  
  0.8413086,  
  1.2783203,  
  ...],
```

"creepy reptile that looks
kind of like a snake but
also like a lizard"

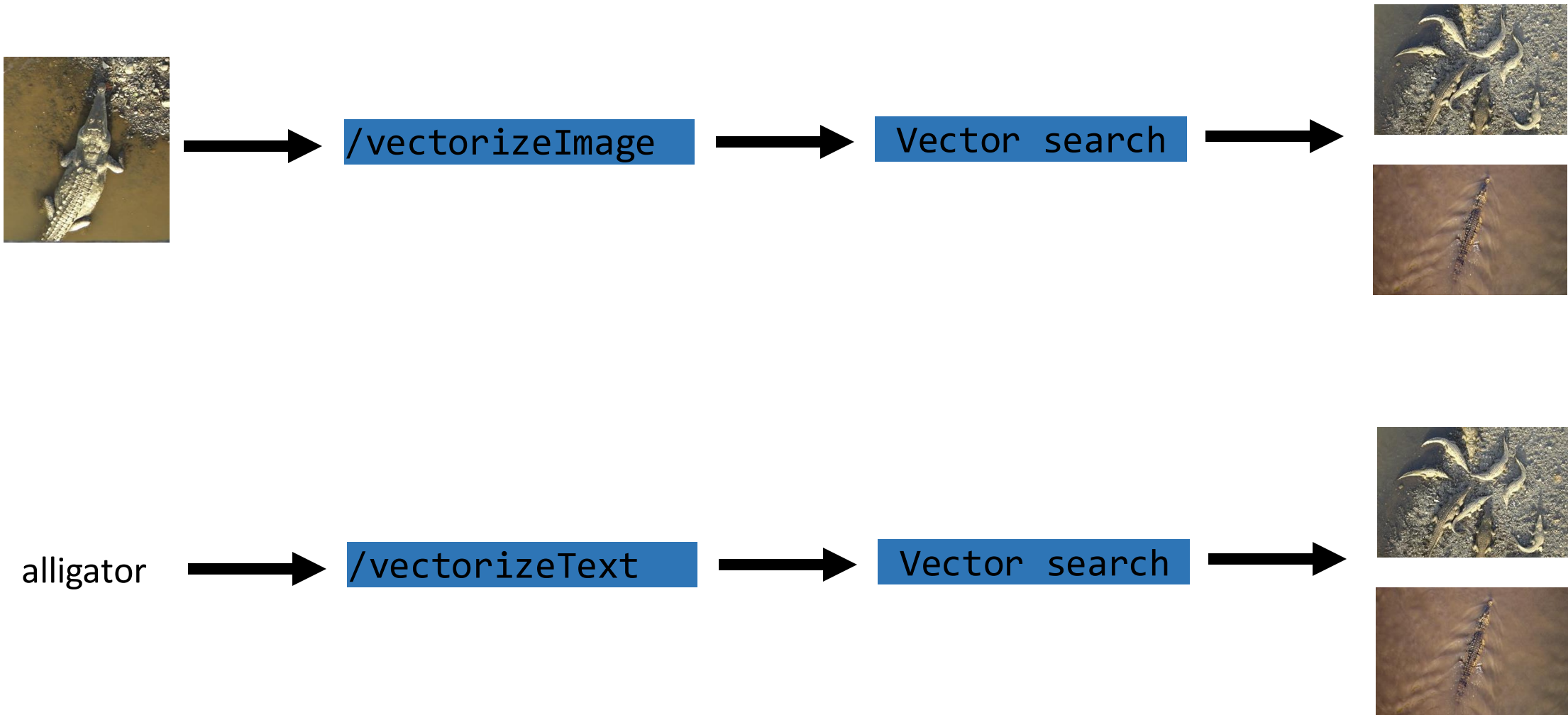


`/vectorizeText`



```
[  
 -0.027022313,  
 -0.011945606,  
  0.019690325,  
  ...],
```

Vector search with multimodal embeddings

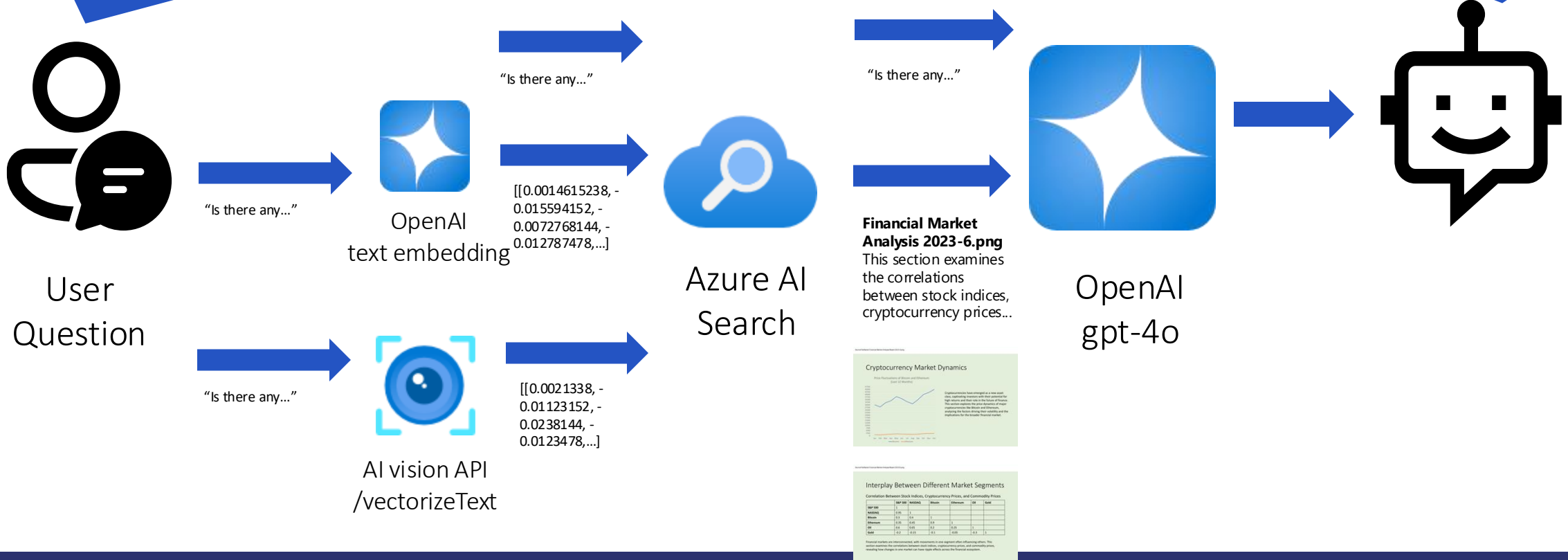


aka.ms/aitour/azure-search

RAG on multimodal documents: Azure architecture

Is there any correlation between oil prices and stock market trends?

Yes, there is a correlation between oil prices and stock market trends²



Workshop

Workshop Repo

<https://aka.ms/expertslive-rag>

Workshop Repo

<https://www.microsoftazurepass.com/>

Important Links



<https://aka.ms/expertslive-links>



Koreyspace – LinkedIn and GitHub



@juliamuiruri4 – LinkedIn and GitHub

Thank you to our Sponsors

