



Sinergija²⁴

Cooking RAG like a Chef de Cuisine

3-5th December, 2024, Crowne Plaza Hotel Belgrade

Andrey Vykhotsev
Head of AI & Data
Aprova GmbH.

Sponsored by

Microsoft

Goals

- Help navigate complex Gen AI space
- Help understand Azure 1st Party options
- Learn basic and advanced AI RAG concepts

Agenda

- Why RAG?
- Choosing RAG Components
 - LLMs
 - Retrievers
 - Orchestrators
- Running RAG on Azure
- RAG Evaluation and Security
- Beyond RAG (Agentic RAG, GraphRAG)

AI Development these days

Classic development



AI Development



Gen AI field - “let the flowers bloom”

GENERATIVE AI COMPANIES WITH >\$5MM RAISED (AS OF MARCH 2023)

Total Raised (\$MM USD)

Segment / Modality	\$5 – 15MM	\$16 – \$40MM	\$41 – 100M	\$100M+ Raised
Horizontal Application	Speech & Audio krisp NEAI MURFAI loora astrid. TENYX VOZY LOVO REPLICA	\$5 – 15MM skit Corti symbolai voicemod deepdubai supertone	\$16 – \$40MM mavenoid contents peppercontent writer regie.ai agolo	\$41 – 100M DeepL Kasisto inSTRIED texcio haptik
	Text, Chat, & Translation nlx LANDBOT copy.ai CERTAINLY LAVENDER Regulo personaldot verloop.p	\$16 – \$40MM autobound Tymely	\$16 – \$40MM ultimate.ai PhotoRoom uizard beautiful.ai	\$100M+ Raised Jasper ada grammarly Foresighted
	Image, Visual, & Design Simplified PRISMA skippr modifi ArtBlocks reface ZMO.AI VIZCOM	\$16 – \$40MM Colossyan Waymark YEPIC-AI detail	\$16 – \$40MM Imagen Facet Hour One Grain Twelve Labs	\$100M+ Raised Typeface tome Lighticks
	Video Peach Rephraser.ai GLOSSAI Colossyan Waymark YEPIC-AI	\$16 – \$40MM R VEED.JO video Hour One	\$16 – \$40MM rct.ai aletheia.ai	\$100M+ Raised synthesia DHD VIDEOVERSE
	3D, Simulation, & XR Kinetix FABLE UNEEQ METAPHYSIC scenario CSM Maria	\$16 – \$40MM MO	\$16 – \$40MM mem xembly fireflies.ai	\$100M+ Raised inworld Leia Inc soul machines
	General Productivity Prodigal onloop SPIRITT viable	\$16 – \$40MM ask-ai FATHOM	\$16 – \$40MM Air mem Hebbia vectara Weaviate	\$100M+ Raised Otter.ai Sana Labs
	Search seek nuclia Metaphorix	\$16 – \$40MM Perplexity DASHWORKS	\$16 – \$40MM Hebbia vectara Weaviate	\$100M+ Raised neeva YOU
	Marketing BI & Website Design madgicx Demandwell Dynamio16 The.com	\$16 – \$40MM Omnyx Outplay Quattr duriable MarketMuse Lately TOPLINE PRO	\$16 – \$40MM anyword unbounce	\$100M+ Raised Munny [PERSADO]
	Code anima	\$16 – \$40MM bloop.	\$16 – \$40MM codacy tabnine Magic	\$100M+ Raised warp zeplit
	Music WORLDBORN	\$16 – \$40MM moises	\$16 – \$40MM SPLASH	\$100M+ Raised nimji
Vertical Application	Health & Drug Discovery Valence Profluent sorceror* Antiverse	\$16 – \$40MM SYNTECTRA Atomica.ai ATROPOS HEALTH	\$16 – \$40MM wyso abridge BASECAMP RESEARCH Nabla retrace	\$100M+ Raised Generate Biomedicines Wooblet Health Instico Medicine Envedo MDCLONE Paige
	Other Specialized Verticals HARVONIAI ROSS KOKind Harvey	\$16 – \$40MM SWAPP TERM SCOUT zuma	\$16 – \$40MM Archistar DATA SKRIVE TUTORROW TESTFIT zuma	\$100M+ Raised casetext REPLICA MAD
Enabling Infrastructure	ML Ops / Dev Tools LatticeFlow OhiKise xetdata	\$16 – \$40MM featureform WHYLABS mystic	\$16 – \$40MM mosaicML aporia Iterative Outerbounds	\$100M+ Raised comet LightningAI ROBUST INTELLIGENCE
	Data (e.g., Synthetic, Labeling) MINTIUM Datomize	\$16 – \$40MM lang.ai SaturnCloud	\$16 – \$40MM MOSTLY-AI Synthesis AI Diveplane hazy	\$100M+ Raised zilliz gretel TOMIC datagen parallel domain
	Infra / Research / Model Develop. ALPHALPHA chooch Unlikely	\$16 – \$40MM generally intelligent	\$16 – \$40MM AI FOUNDATION	\$100M+ Raised AI21 Labs runway
Sources: Pitchbook, Crunchbase, public sources Note: Includes private, non-exited, companies with over \$5MM of total fundraising raised (as disclosed by Pitchbook) For companies that are multi-modal or multi-segment, we selected what we believe is the primary modality/segment				
Produced by Kelvin Mu (www.linkedin.com/in/kelvinmu)				

Retrieval augmented generation: bring your data to the prompt

Text input that provides some framing as to how the engine should behave

User provided question that needs to be answered

Sources used to answer the question

Prompt

You are an intelligent assistant helping Contoso Inc employees with their healthcare plan questions and employee handbook questions. Answer the following question using only the data provided in the sources below.

Question: Does my health plan cover annual eye exams?

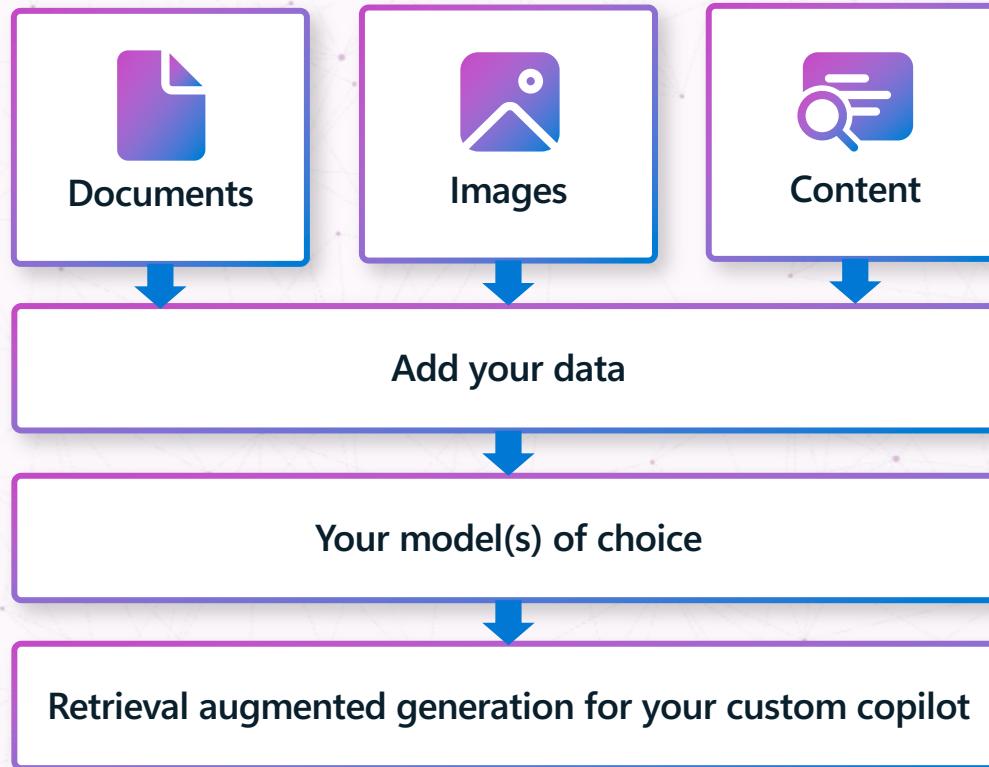
Sources:

1. Northwind Health Plus offers coverage for vision exams, glasses, and contact lenses, as well as dental exams, cleanings, and fillings.
2. Northwind Standard only offers coverage for vision exams and glasses.
3. Both plans offer coverage for vision and dental services.

Response

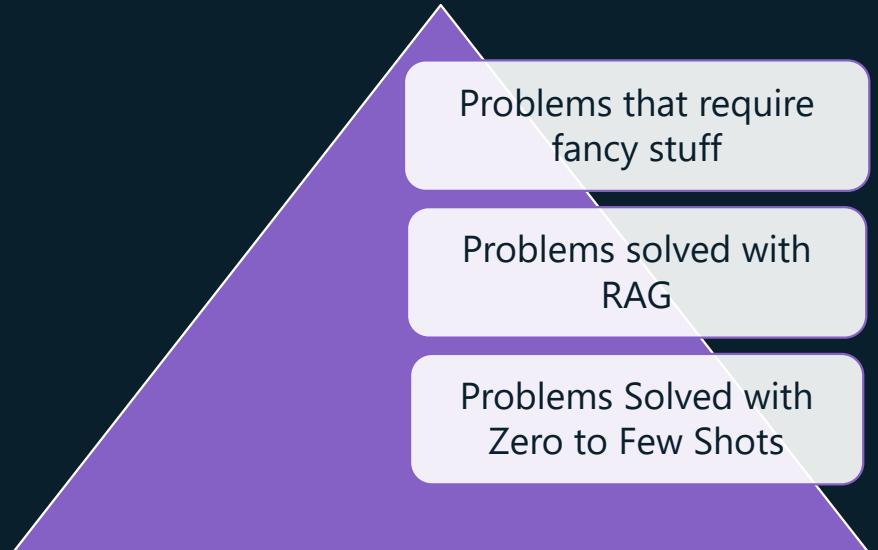
Based on the provided information, it can be determined that both health plans offered by Northwind Health Plus and Northwind Standard provide coverage for vision exams. Therefore, your health plan should cover annual eye exams.

Retrieval Augmented Generation



Why RAG?

- 80% of the customer use cases can be reduced to RAG
- RAG is the easiest way to give model “knowledge” beyond training data
- RAG is solving 2 problems:
 - Models are limited in context size (128k/256k)
 - Given very large context, models are getting confused and struggling to answer



Retrieval Augmented Generation

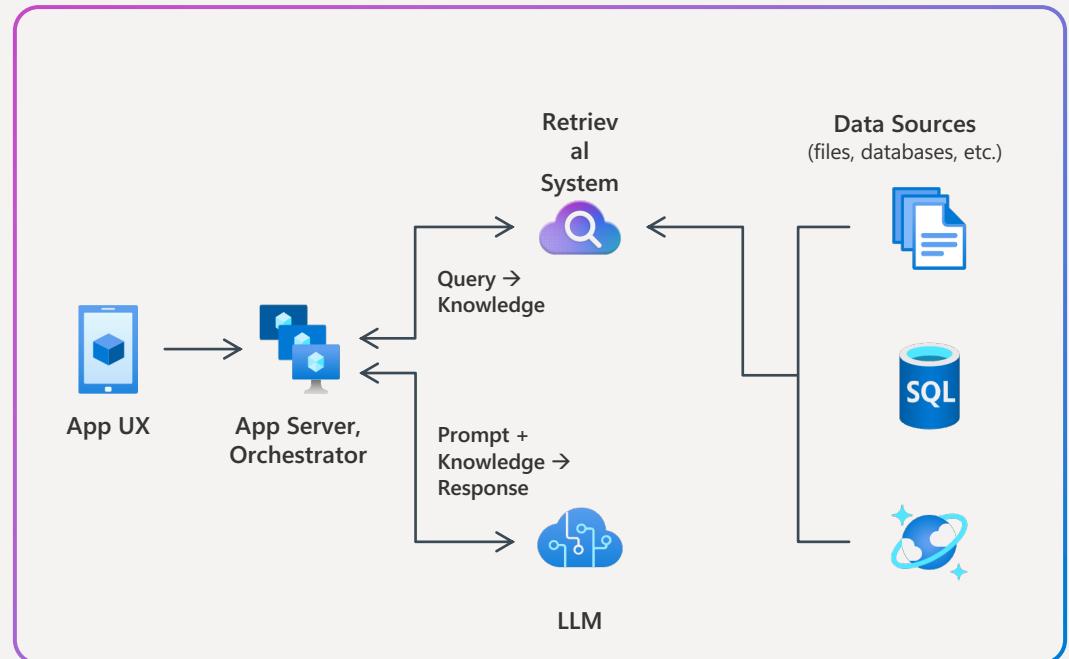
Combine reasoning + knowledge

Key elements

- Externalized knowledge
- Orchestrator drives interaction
- Prompts = instructions + context + grounding data

Achieving quality results

- Different workflows for different tasks
- Evaluation & RAI



Naïve RAG vs Advanced RAG

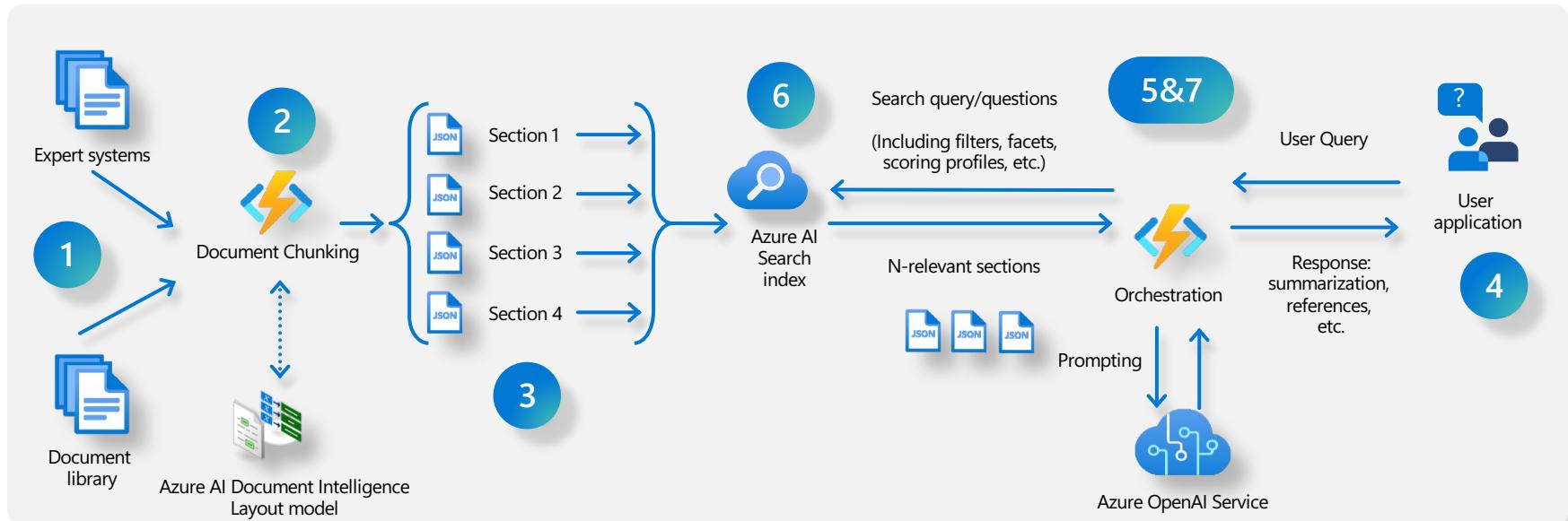
Naïve RAG

- Very little orchestration
- Unsophisticated query rewrites
- Straightforward retrieval
 - Vector
 - Hybrid
- One data source / document type
- One modality
- Simple filtering

Advanced RAG

- Complex orchestration (ReACT, Chain or Tree of Thought)
- Sophisticated query rewrites
 - Rephrasing, decomposition, step back, HyDe
- Complex retrieval / embedding
 - Small-to-big, hierarchical, reference etc.
- Knowledge Graphs
- Agentic RAG

Anatomy of RAG



1. Data ingestion

Different data formats and system of records

2. Chunking

What is the best Chunking strategy?

3. Indexing

Shall I use vector embeddings data transformation, mappings?

4. User interface

Chatbot for Q&A surfaced to end users

5. Orchestration

Communication coordination and prompting—Prompt to get retriever query

6. Data retrieving

Shall I use vector, semantic, keyword or hybrid approach?

7. Orchestration

Communication coordination: create user response based on retrieve data and send to User app

Azure OpenAI on your data



Connect or ingest your data



Ground Azure OpenAI models using your data



Retrieval augmented generation made easy



Restrict responses to your data



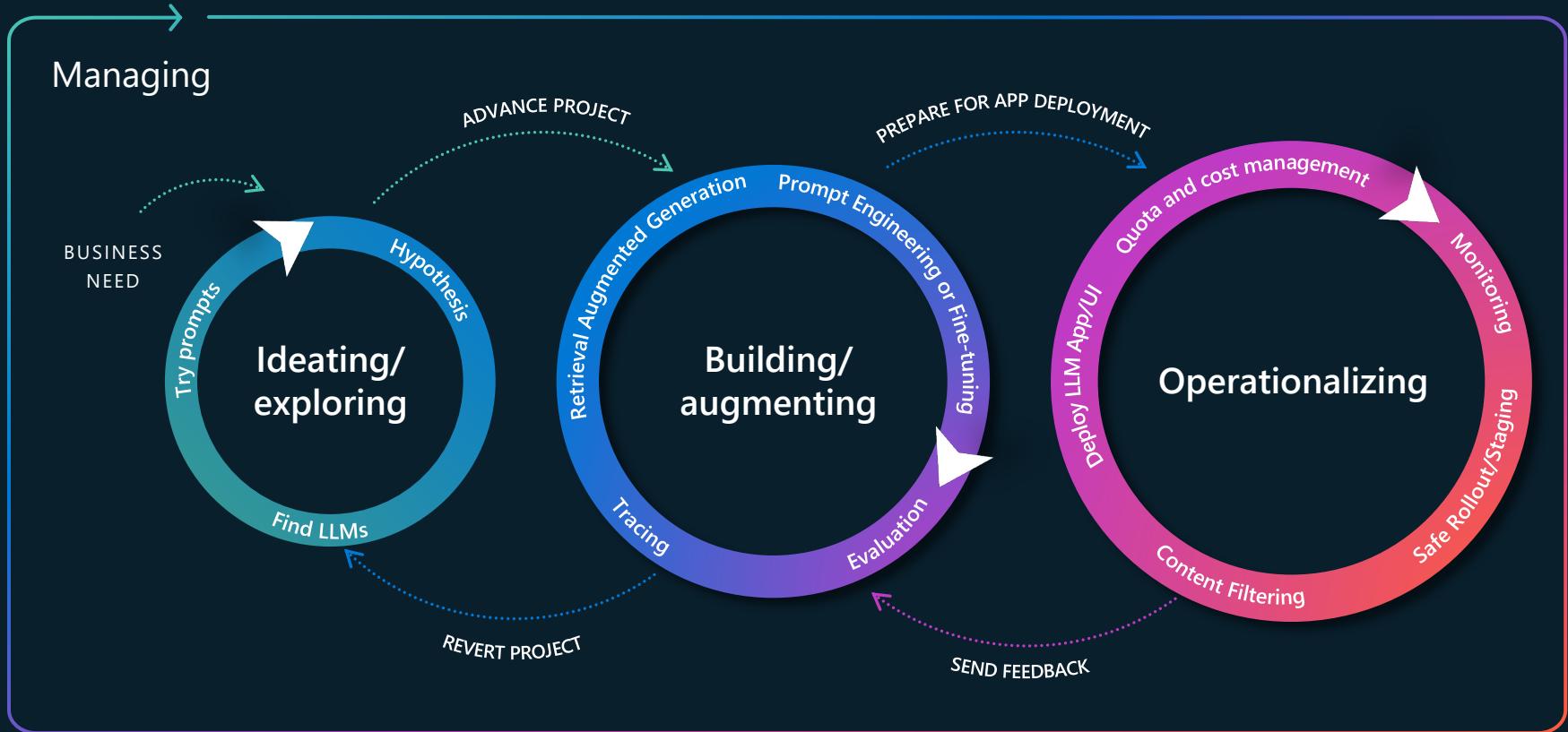
Enterprise grade security including private network, VPN, document level access control, etc.



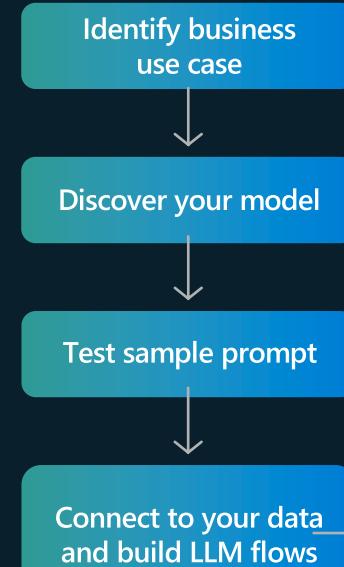
Easy deployment as a copilot

Demo – Azure OpenAI “On your data”

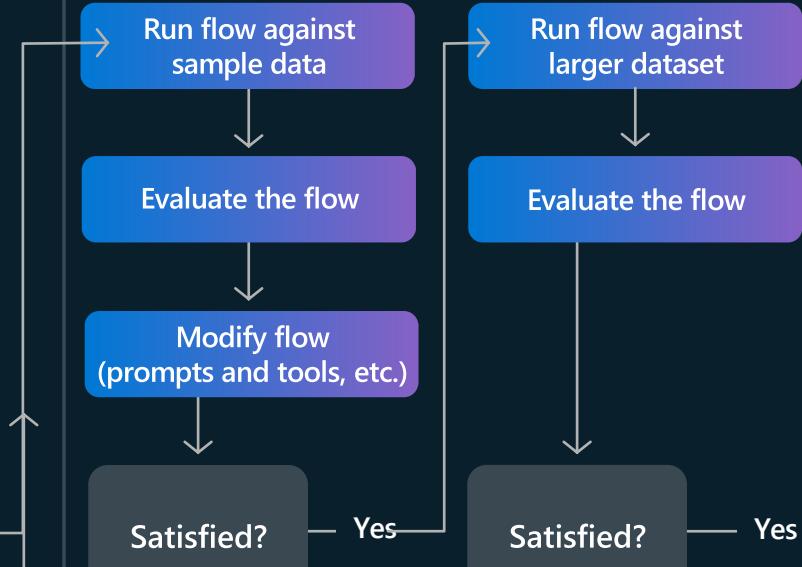
GenAI Lifecycle in the real world



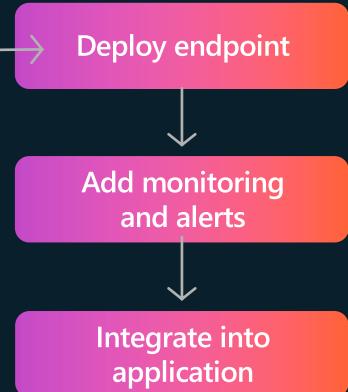
1. Ideating/exploring



2. Building/augmenting



3. Operationalizing



What can go wrong with Naïve RAG

- At node in the above 2 diagrams, things can go wrong

Low precision/Recall

Right data / wrong answer

Hallucinations / Irrelevant answers

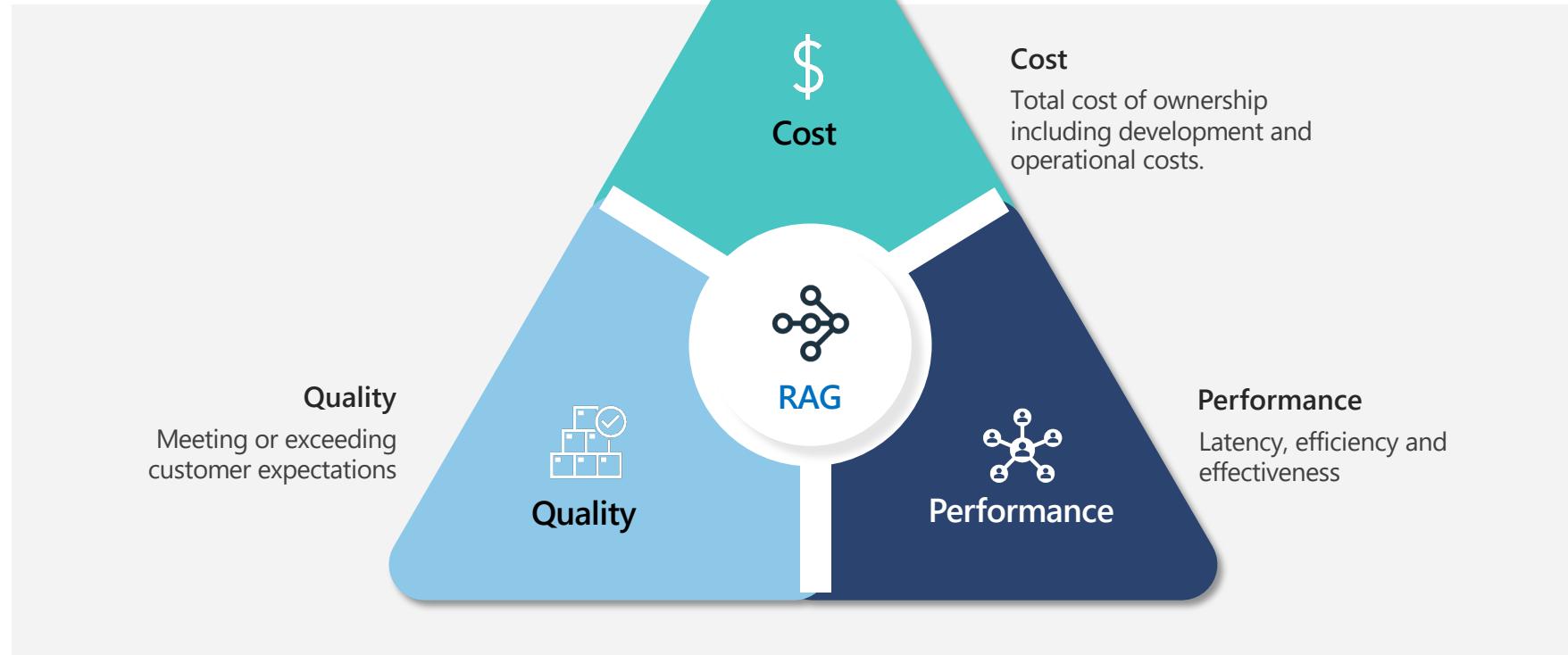
Wrong data / Confident (wrong) answer

Outdated data / no data recency sorting

Fail to report information is missing

Optimization

is a Long Journey



Step 1: prompt engineering

A year of rapid growth and innovation

2023 Ignite

GA of Model Catalog
and Launched Models as a Service (MaaS) at Ignite in partnership with Meta Llama

2024 Jan-Oct

Launched 50+ MaaS models
Cohere, Mistral, AI21, G42, Phi, Meta, Nixtla
250+ OSS models on Managed compute
Launched models on Github

2024 Ignite

Over 1800+ models in the catalog
and continuing the momentum with new models
Providing flexible deployment options
Seamless experimentation with Azure AI Model Inference API

Azure AI Model Catalog continues to evolve, providing a diverse range of AI models and unparalleled flexibility for enterprise use cases

Find the best model for every use case.

Avoid model lock-in:

- Flexible model selection
- Curated for enterprise
- Explore, Compare and Swap models quickly

Azure AI Model Catalog

LLMs and SLMs

- Flagship LLMs
 - GPT-4
 - Mistral Large
 - Llama3 70b
 - Llama 405B
 - Command R, R+

- Small Language Models
 - Phi3
 - Mistral OSS models
 - Llama3 8b
 - Minstral

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 proprietary

- 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



Model benchmarking

Model Benchmarking enables you to review and compare the performance of various AI models, simplifying the selection process and allowing users to make confident choices with their modeling needs.

- Gain quality metrics for Azure OpenAI Service, Llama 2 family, Code Llama, and Mistral models
 - Access pre-built metrics and benchmark comparison models within the same build, train, deploy environment
 - Compute accuracy scores at both the task (dataset) and model levels by utilizing public datasets, yielding a model score for each dataset
 - Compare scores of multiple models across datasets and tasks
 - Benchmark results originate from public datasets; Azure AI evaluation pipelines download data from original sources, extract prompts from each row, generate model responses, and then compute relevant accuracy metrics

OSS version:Chatbot Arena

API and model choice

Azure AI Studio All hubs / hub_contoso-AI All projects / ContosoPharmaceutical-0034 Model benchmarks

Filters Hide

Tasks

Quality benchmarks Embeddings benchmarks

Compare benchmarks across models and datasets available in the industry to assess which one meets your business scenario. Learn more about how model performance is scored

All filters Tasks Collections Models Datasets Dashboard List

Charts

Model accuracy

Accuracy scores are presented at the dataset and the model levels. At the dataset level, the score is the average value of an accuracy metric computed over all examples in the dataset. [Learn more](#) about accuracy.

The chart displays accuracy scores for numerous datasets and models. The x-axis lists datasets such as 'confidence-7b', 'confidence-12b', 'llama-2-7b', 'llama-2-70b', 'llama-2-70b-chat', 'llama-2-7b-chat', 'llama-2-13b', 'llama-2-13b-chat', 'llama-2-70b-instruct', 'llama-2-7b-instruct', 'llama-2-7b', 'microstr-phi-2', 'mistral-community-mistral-8x22b-v0-1', 'mistral-mistral-7b-instruct-v0-1', 'mistral-mistral-7b-v0-1', and 'mistral-large'. The y-axis represents accuracy scores ranging from 0.2 to 1.0. A blue line indicates the mean accuracy across these datasets, showing a general upward trend from approximately 0.35 to 0.85.

Image may not reflect actual user interface.



Enable the comparison of models based on accuracy and empower users to make data-driven decisions, ensuring their AI solutions are optimized for the best performance.

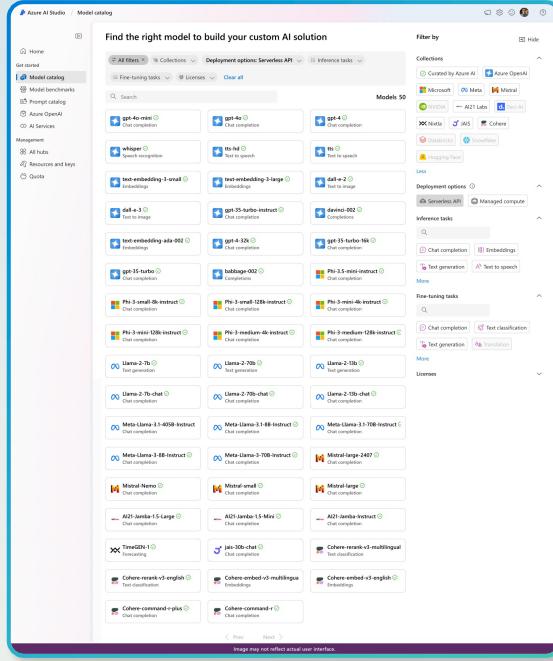


Models as a Service (MaaS)

Offering within the model catalog

Models as a Service (MaaS) lets you fine-tune models without provisioning compute, making it easier for generative AI developers to build custom copilots.

- Ready-to-use APIs with pay-as-you-go billing based on tokens
- Customize models with your own data without the need to set up and manage GPU infrastructure
- Easily integrate the latest AI models as an API endpoint to applications
- Integrate with preferred orchestration tools like prompt flow, Semantic Kernel, or LangChain
- Achieve serverless fine-tuning without provisioning GPUs
- Fine-tune Llama 2 with your own data to enhance the model's ability to generate more precise predictions



MaaS provides simplified management with ready-to-use GPU provisioning, lowering costs and reducing barriers to adoption by eliminating complexity.



Walkthrough – Azure AI foundry Model Catalogue

LiteLLM.ai

<https://docs.litellm.ai/docs/>

LiteLLM - Getting Started

LiteLLM - Getting Started

<https://github.com/BerriAI/litellm>

Call 100+ LLMs using the OpenAI Input/Output Format

- Translate inputs to provider's `completion`, `embedding`, and `image_generation` endpoints
- Consistent `output`, text responses will always be available at `['choices'][0]['message']['content']`
- Retry/fallback logic across multiple deployments (e.g. Azure/OpenAI) - [Router](#)
- Track spend & set budgets per project [LiteLLM Proxy Server](#)

How to use LiteLLM

You can use litellm through either:

1. [LiteLLM Proxy Server](#) - Server (LLM Gateway) to call 100+ LLMs, load balance, cost tracking across projects
2. [LiteLLM python SDK](#) - Python Client to call 100+ LLMs, load balance, cost tracking

When to use LiteLLM Proxy Server (LLM Gateway)



TIP

Use LiteLLM Proxy Server if you want a **central service (LLM Gateway) to access multiple LLMs**

Typically used by Gen AI Enablement / ML Platform Teams

- LiteLLM Proxy gives you a unified interface to access multiple LLMs (100+ LLMs)
- Track LLM Usage and setup guardrails
- Customize Logging, Guardrails, Caching per project

When to use LiteLLM Python SDK



TIP

Use LiteLLM Python SDK if you want to use LiteLLM in your **python code**

Typically used by developers building llm projects

GPT model impact on TCO

Model	Context	Cost per 1,000 tokens		Cost Comparision (1M tokens)		
		Input	Output	Input (20%)	Output (80%)	Total
GPT-4o-2024-08-06 Global Deployment	128K	\$0.003	\$0.010	\$0.50	\$8.00	\$8.50
GPT-4o Global Deployment	128K	\$0.005	\$0.015	\$1.00	\$12.00	\$13.00
GPT-4o-mini Global Deployment	128K	\$0.00015	\$0.0006	\$0.03	\$0.48	\$0.51
GPT-4	32K	\$0.03	\$0.06	\$6.00	\$48.00	\$54.00
GPT-4	8K	\$0.06	\$0.12	\$12.00	\$96.00	\$108.00
GPT-3.5-Turbo-0125	16K	\$0.0005	\$0.0015	\$0.10	\$1.20	\$1.30
GPT-3.5-Turbo-Instruct	4K	\$0.0015	\$0.002	\$0.30	\$1.60	\$1.90
GPT-3.5-Turbo-0613	16K	\$0.003	\$0.004	\$0.60	\$3.20	\$3.80
GPT-3.5-Turbo-1106	16K	\$0.001	\$0.002	\$0.20	\$1.60	\$1.80



Empowering GitHub Developers to Build with AI

The screenshot displays the GitHub Marketplace interface. At the top, there's a search bar with placeholder text "Type ⌘ to search or ask Copilot". Below it, a sidebar on the left lists categories: "Featured", "Copilot", "Models", "Apps", "Actions", and a button "+ Create a new extension". The main content area has a heading "Enhance your workflow with extensions" and a sub-section "Tools from the community and partners to simplify tasks and automate processes". A search bar below this says "Search for Copilot extensions, apps, actions, and models". The central part of the page features a section titled "Models for your every use case" with three cards: "Mistral Large" by Mistral, "OpenAI GPT-4o" by Azure OpenAI Service, and "Phi-3 medium instruct (128k)" by Microsoft. Below this, another section titled "Discover apps with Copilot extensions" shows three app cards: "Copilot" (with a hand icon), "Copilot" (with a person icon), and "Copilot" (with a speech bubble icon). A note at the bottom left says "Image may not reflect actual user interface.".

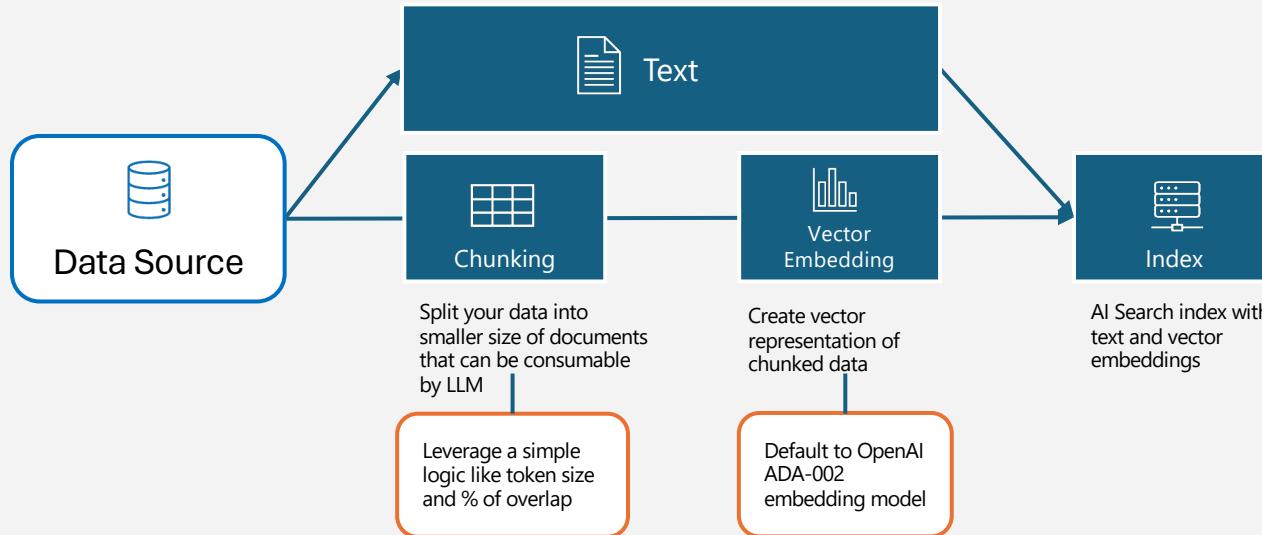
Discover in GitHub Models
<https://github.com/marketplace/models>

- Get access to a curated list of our top AI models, without leaving GitHub
- Mix, match, and experiment with models in the GitHub playground for free*
- Unlock the full capabilities of the latest models with Azure

*Usage limits apply

Data Ingestion

Prepare your data for chat experience



- Index text representation of your data / documents
- Basic index schema, expecting vector representatives to be enough for high accuracy retrieval.

Document parsing

- Azure Document Intelligence
- LLamaParse
- Document loaders
 - Python / langchain / llamacindex etc
- Azure AI services for multi-modal
- Azure Content Understanding

The image shows two screenshots demonstrating document parsing and content understanding.

Microsoft Fabric Copilot Results:

lang en
Download Markdown [Download Markdown](#)

Results

78a8afb5-fa6a-42d6-9ccf-a6413bb5beaa

Markdown Text JSON Images XLSX

Microsoft | Microsoft Fabric

Copilot in Microsoft Fabric

Speaker

The image shows a title slide for a presentation about "Copilot in Microsoft Fabric". The Microsoft logo and "Microsoft Fabric" text are displayed in the top left corner. The main visual element is a stylized, semi-circular shape divided into sections of different colors – turquoise, orange, pink, and purple. This abstract shape appears to be a modern, artistic representation possibly symbolizing the various components or capabilities of Microsoft Fabric. The background is white, giving the slide a clean and professional appearance.

—
| 87% | of organizations believe AI will give them a competitive edge |
|—————|—————|

Source: MIT Sloan Management Review

2 Microsoft Fabric

CIOs are accelerating their efforts to bring AI to their data estate

—
Introducing Microsoft Fabric

Document Intelligence Studio - General Document:

Document Intelligence Studio > General Document

General document

API version: 2023-07-31 (3.1 General Availability) Service resource: av-docintel-demo

Drag & drop file here or Browse for files or Fetch from URL

Run analysis Analyze options

Fields Content Result Code

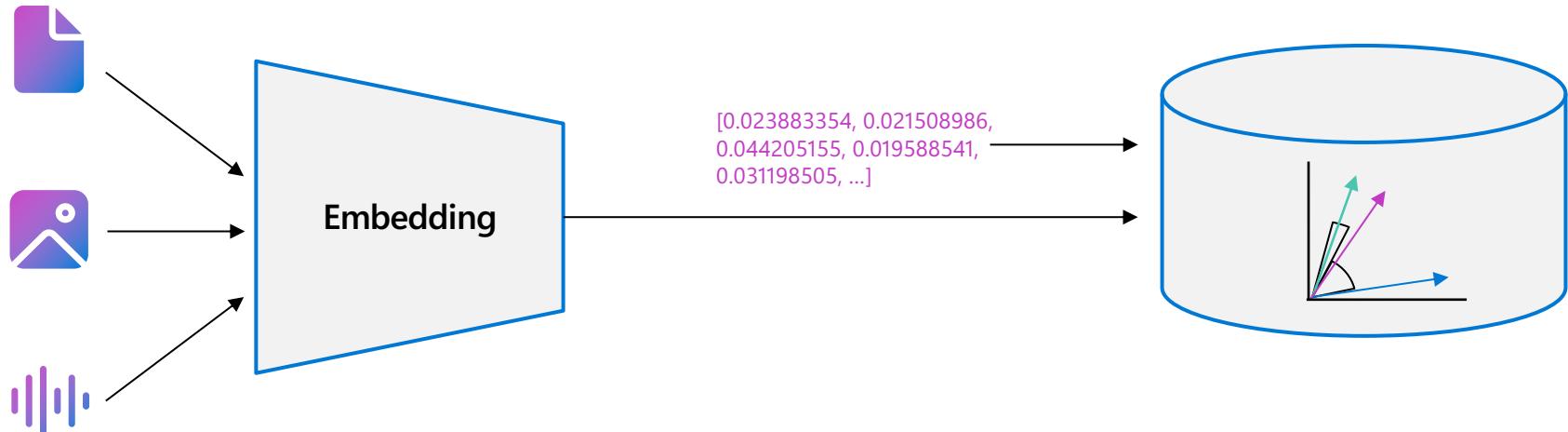
Key-value pairs

U.S. Department of the Interior Bureau of Safety and Environmental Enforcement (BSEE)
Application for Permit to Drill (APD)

Submit original plus THREE copies, with ONE copy marked "Public Information."

Key	Value	Confidence
NEW WELL	N/A	selected: 88.60%
SIDETRACK	N/A	unselected: 90.90%
BYPASS	N/A	unselected: 90.90%
DEEPEN	N/A	unselected: 90.90%
OPERATOR NAME and ADDRESS	N/A	91.30%

How embeddings work



Find the most relevant information in large datasets using hybrid search with keywords, vectors, and semantic ranker

LLMs for embeddings

queries with `;`)

Search for a model and press enter...

Open Proprietary Sentence Transformers Cross-Encoders <100M 100M to 250M 250M to 500M
 Bi-Encoders Uses Instructions No Instructions 500M to 1B >1B

Overall Bitext Mining Classification Clustering Pair Classification Reranking Retrieval STS Summarization MultilabelClassification
Retrieval w/Instructions

English Chinese French Polish Russian

Overall MTEB English leaderboard ⓘ
Metric: Various, refer to task tabs
Languages: English

Rank	Model	Model Size (Million Parameters)	Memory Usage (GB, fp32)	Embedding Dimensions	Max Tokens	Average (56 datasets)	Classification Average (12 datasets)	Clustering Average (11 datasets)
1	NV_Embd-v2	7851	29.25	4096	32768	72.31	90.37	58.46
2	bge-en-ic1	7111	26.49	4096	32768	71.67	88.95	57.89
3	stella_en_1.5B_v5	1543	5.75	8192	131072	71.19	87.63	57.69
4	SFR_Embd-2_R	7111	26.49	4096	32768	70.31	89.05	56.17
5	ctrl-ctrl-2B-1shot-1024-K-M	7111	26.49	4096	32768	70.24	86.59	56.09

Find the right model to build your custom AI solution

_collections Industry Deployment options Inference tasks: Embeddings
Fine-tuning tasks Licenses Clear all

Search Models 10

 text-embedding-3-small  Embeddings	 text-embedding-3-large  Embeddings	 text-embedding-ada-002  Embeddings
 Cohere-embed-v3-multi...  Embeddings	 Cohere-embed-v3-english  Embeddings	 MedImageInsight  Embeddings
 OpenAI-CLIP-Image-Text-E...  Embeddings	 Facebook-DinoV2-Image-E...  Embeddings	 Facebook-DinoV2-Image-E...  Embeddings

<https://huggingface.co/spaces/mteb/leaderboard>

Integrated Vectorization

End-to-end data ingestion, chunking, vectorization, and advanced retrieval



Chunking

- Built-in Chunking skill (updates to split skill) and updates to index to manage Chunks vs. full documents
- Configure Chunking parameters (e.g., **pages**, overlap window, etc.)
- Automated via Indexer orchestration



Vectorization

- Bring-your-own Azure OpenAI endpoint
- Bring-your-own Azure AI Studio model catalog embedding model (Preview)
- Use AI Vision multimodal embeddings model deployed in AI multi-service account (Preview)
- Support for other embedding model REST endpoints
- Query Vectorization capability

Your document Chunking strategy matters

Chunking solves context length problems and improves retrieval accuracy

1. Fixed Sizes
2. Overlapped chunking
3. Semantic
4. Percentile
5. Interquartile chunking
6. ...

Retrieval Configuration	Single vector per document [Recall@50]	Chunked documents [Recall@50]
Queries whose answer is in long documents	28.2	45.7
Queries whose answer is deep into a document	28.7	51.4

Chunk boundary strategy	Recall@50
512 tokens, break at token boundary	40.9
512 tokens, preserve sentence boundaries	42.4
512 tokens with 10% overlapping chunks	43.1
512 tokens with 25% overlapping chunks	43.9

Retrieval Configuration	Recall@50
512 input tokens per vector	42.4
1024 input tokens per vector	37.5
4096 input tokens per vector	36.4
8191 input tokens per vector	34.9

Advanced Chunking strategies

Chunking methods

Fixed-size Chunking

Simple and common, maintaining a balance between overlap and semantic context

Specialized Chunking

For structured content like Markdown and LaTeX

“Content-aware” Chunking

Content splitting using NLP libraries such as AI Document Intelligence Layout API

Recursive Chunking

Divides text into smaller Chunks in a hierarchical manner

→ AI Search Built-in Split Skill

→ AI Search Custom Skill

Determining the best Chunk size

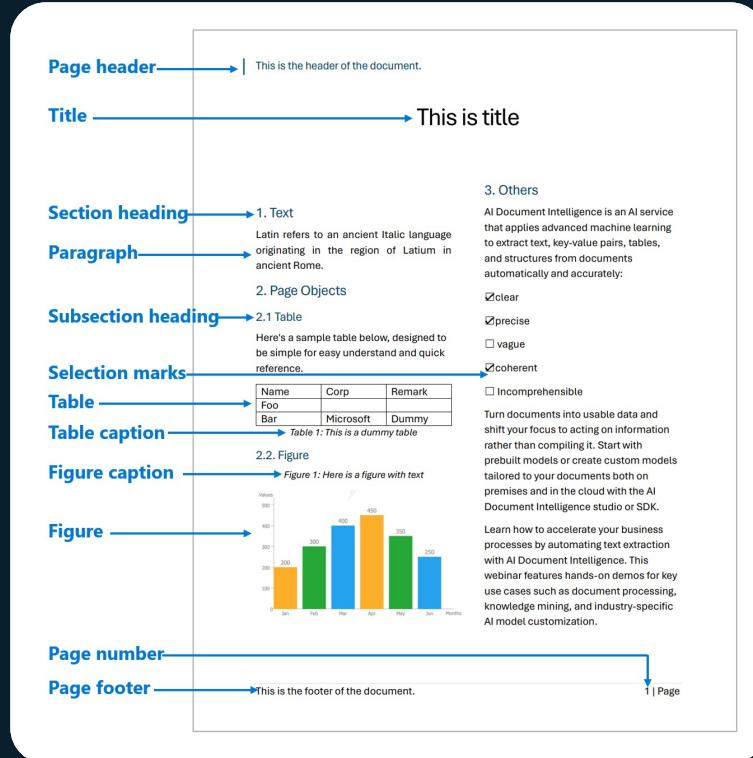
Preprocessing data
for quality

Selecting a range of potential
Chunk sizes considering
content nature and embedding
model capabilities

Evaluating performance
of each Chunk size

🔗 [Tech blog with suggested Chunking sizes depending on scenario](#)

Indexing-Side Improvements: Structure-Aware Chunking



→ Azure AI Search now supports both fixed and content aware chunking

- Powered by Azure AI Document Intelligence
- Headings help understand content hierarchy
- Handles tables, lists, figures, etc.

→ new markdown parsing mode for fast and simple heading handling

Indexing (Pull indexers)

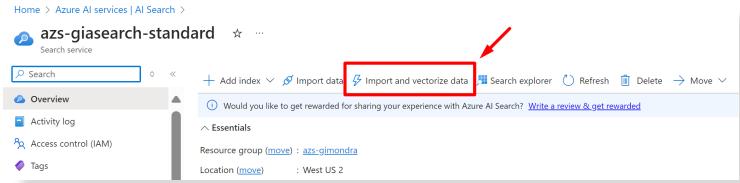
Indexer (pull model)

- Crawler that extracts data from cloud data sources and populates a search index
- Drive skillset execution (AI enrichment)
- Can be run on demand or on a recurring data refresh schedule (incremental indexing)

Best practices

- Understand indexing limits
- Run on a schedule and continuously to correct errors
- Identify data source needs (multiple data sources going to a single index, data source support, limits and configuration)
- Configure data source document deletion policy if needed
- Recognize data types for proper parsing (JSON, JSON lines, CSV, etc.)
- If handling PDFs or images with embedded text, consider AI enrichment with OCR or Azure AI Document Intelligence
- Configure batch size based on workload
- Consider integrated Vectorization if using vector search
- Identify AI enrichment needs, such as language translation or recognition, key phrase extraction, etc.
- Test with various data types/sizes to understand indexing time and optimize the process
- Identify type of authentication being used (Managed identity is recommended)
- Recognize if private endpoints will be used for secure connections and configure shared private links appropriately

Indexing example with Integrated Vectorization



Indexing (Push method)

- Push method uses APIs to upload documents into a search index
- No restrictions on data source type or frequency of execution
- Connectivity and the secure retrieval of documents under your control

Key benefits

Versatility: JSON documents from any source can be uploaded. You can use AI Document Intelligence for data parsing or your preferred parsing library

Flexibility: No restrictions on frequency of data push (no specific schedule, just API restrictions apply)

Control: Retrieval of documents under your own control

Pushing data to Azure AI Search index

- APIs for single or multiple documents upload
- Follow best practices for indexing large documents

Indexing actions

Upload: Insert new documents or update existing ones

Merge: Updates existing document, replacing values

Merge or upload: Behaves like merge for existing documents and like upload for new ones

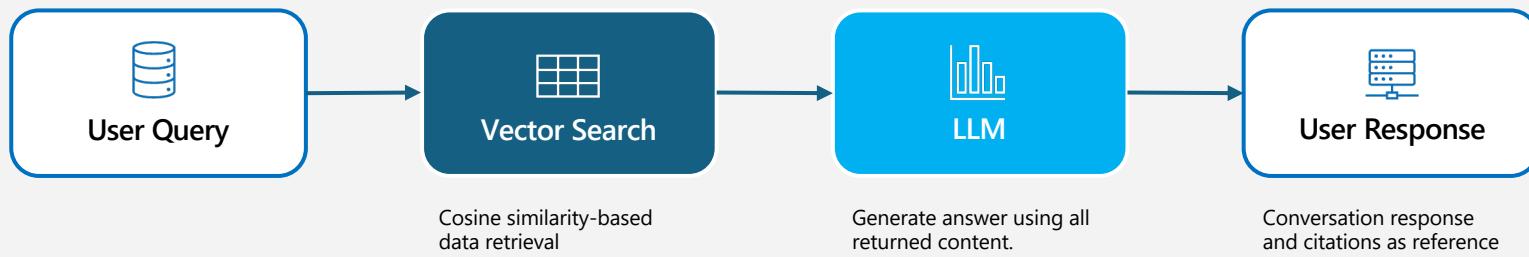
Delete: Removes entire document from index

 [Push method sample for RAG](#)

Demo: Push indexing from Doc Intel

RAG – Data retrieval

Respond user query in conversational chat experience



- Relying on “vector search” for all queries
- Quick and simple “Hello World” implementation

Retrieval strategies



Keyword search

- **For exact, plain text matches**
- “Vocabulary gap” in Q&A systems like Copilot



Vector search

- **For conceptual similarity, or underlying meaning**
- Weak performance on exact matches (like a product ID or code)



Hybrid search

- **Best of both vectors and keywords**
- Brings more accurate responses across various scenarios

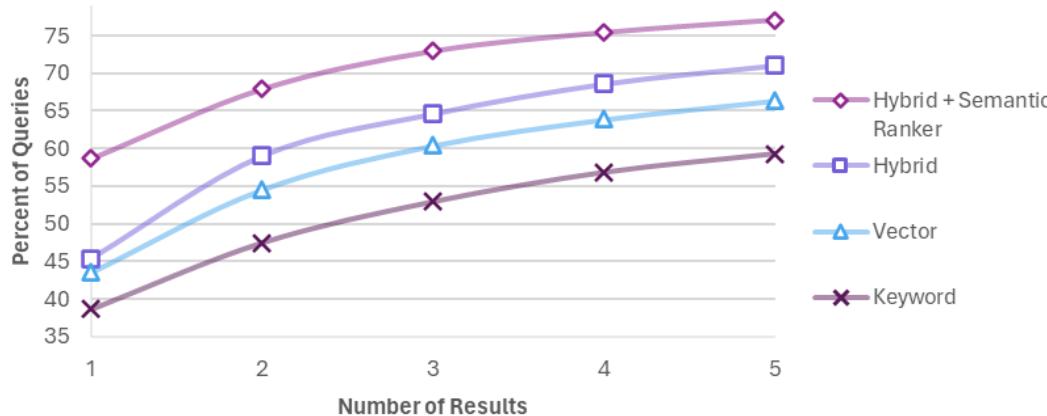


Search re-ranking

- **Scores and ranks all retrieved documents by relevance**
- Reranking runs after performing search strategy (can't retrieve information)

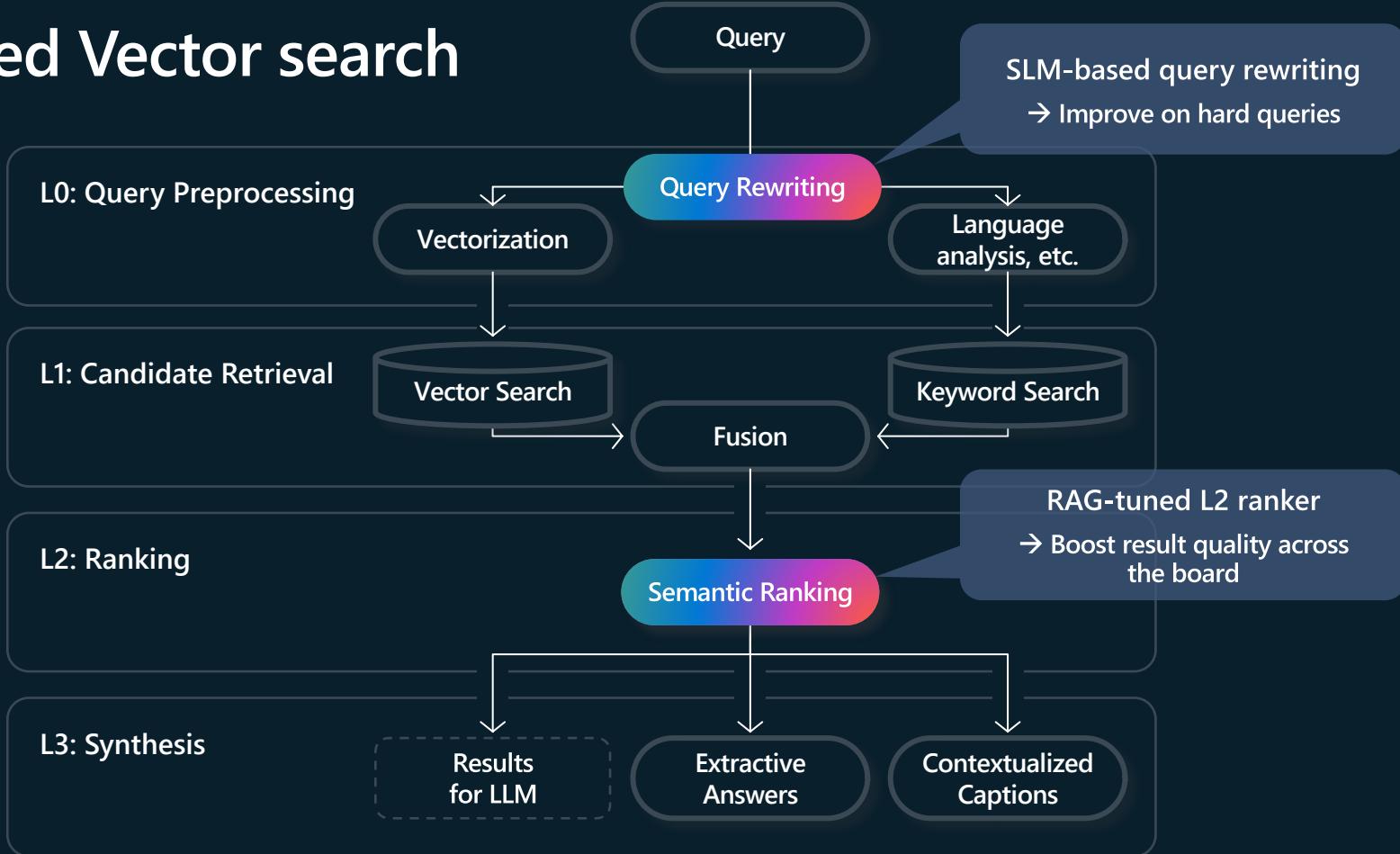
Retrieve using semantic similarity and hybrid search

Hybrid retrieval with semantic ranking
outperforms vector-only search



Find the most relevant information in large datasets using hybrid search with keywords, vectors, and semantic ranker

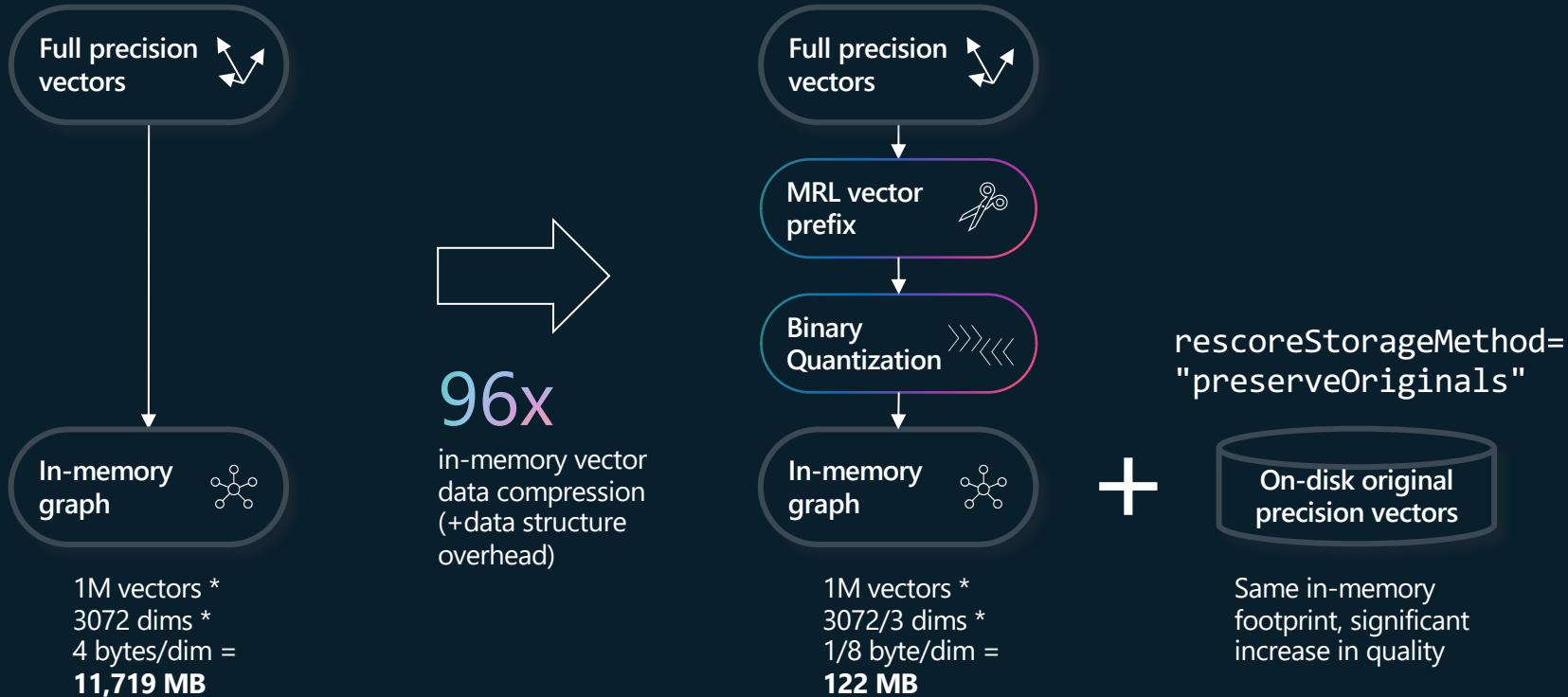
Advanced Vector search



Vector Compression:

Cost-effective retrieval at all data scales

For example, for 1 million vectors, using OpenAI's text-embedding-003-large (3072 dims)



Powering RAG in GitHub Models

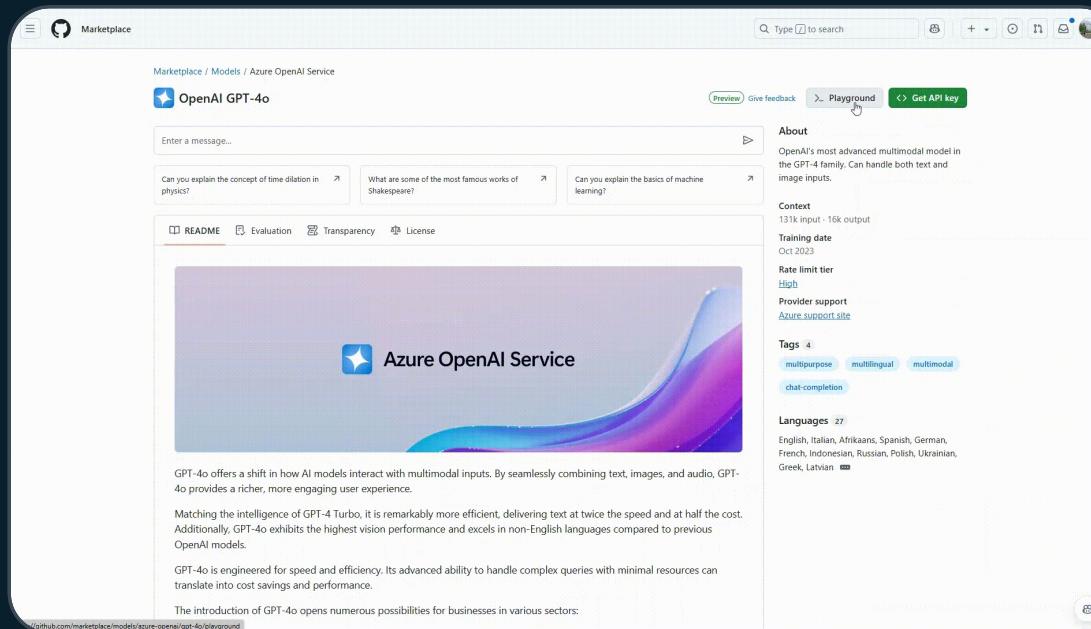
aka.ms/rag-in-github-signup

Easy access glide path from playground to IDE

Free AI Search service created automatically,
without having to switch screens

Full-featured retrieval: use hybrid search,
reranking, query rewriting

From code, test out and modify various features



Summary – Retrievers

- If you want turnkey advanced solution – AI Search
- If you want to keep vectors where your data is, avoid data duplication and ingestion costs – Cosmos, PostgreSQL, MongoDB, Databricks

Choosing an orchestrator

Options

- 1st party
- Promptflow / Prompty
- Semantic Kernel
- LangChain
- LlamaIndex
- Haystack

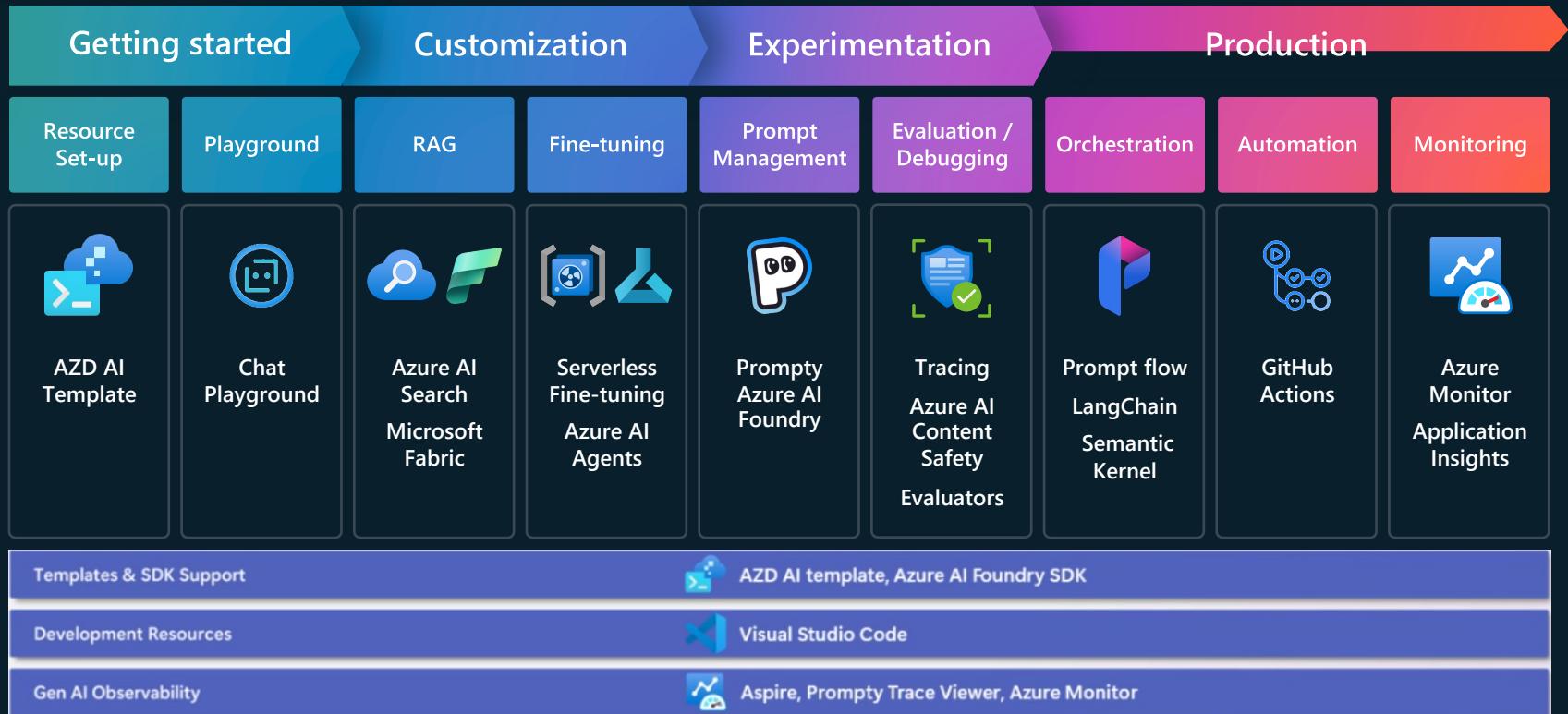
Considerations

- Deployment simplicity and options
- Commercial vs. OSS features
- Integrations (Loaders, Vector Stores)
- Prompt templates / libraries
- Orchestration features (chains, pipelines), streaming, async
- Evaluation features
- CI/CD, LLMOps
- Agentic features

PromptFlow can be used in combination with other orchestrators

AI Foundry tooling

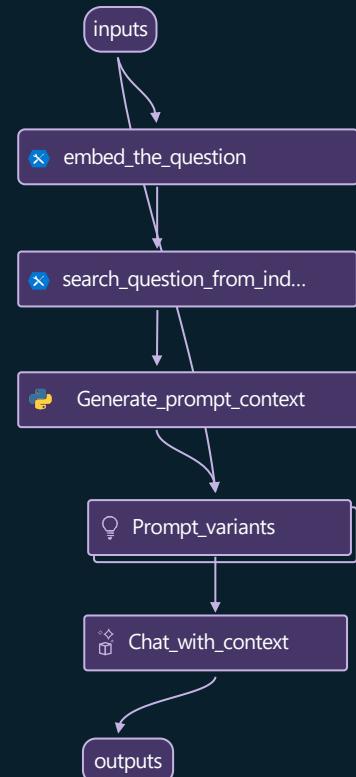
Bringing your AI apps to production



RAG with Azure AI prompt flow

Leverage your own data for generating responses

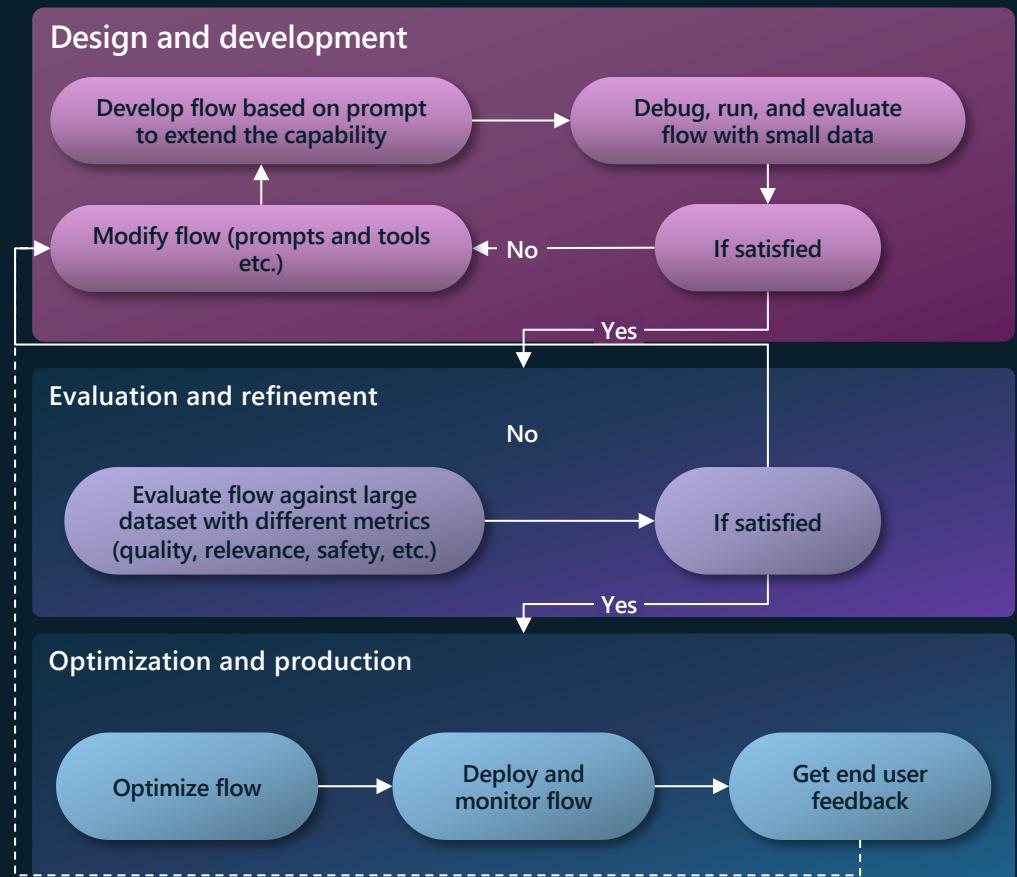
1. **Data Chunking:** Your source data needs to be converted to plain text and chunked into smaller pieces.
2. **Vectorization:** The text chunks are converted into vectors, also known as embeddings.
3. **Data Storage:** These vectors are stored efficiently, often with metadata, to assist in generating responses.
4. **Integration:** Azure AI integrates with Azure OpenAI Service for large language models and supports Azure AI Search as vector stores.
5. **Customization and Monitoring:** Azure AI offers a wizard-based UI for managing data and prompt flows, and also allows for the measurement and enhancement of RAG workflows.
6. **Security:** RAG workflows can be used with private networks in Azure Machine Learning to secure user's data.



Operationalize LLM app development with prompt flow

LLMops is a complex process.
Customers want:

- Private data access and controls
- Prompt engineering
- CI/CD
- Iterative experimentation
- Versioning and reproducibility
- Deployment and optimization
- Safe and Responsible AI



Azure AI prompt flow

Capabilities Overview

- Manage connections

- Develop and manage flows that connect to a variety of foundation models, vector databases, APIs, prompts and Python tools.

- Test and evaluate

- Test flows with large datasets in parallel
- Evaluate the AI quality and safety of the workflows

- Tracing and debugging

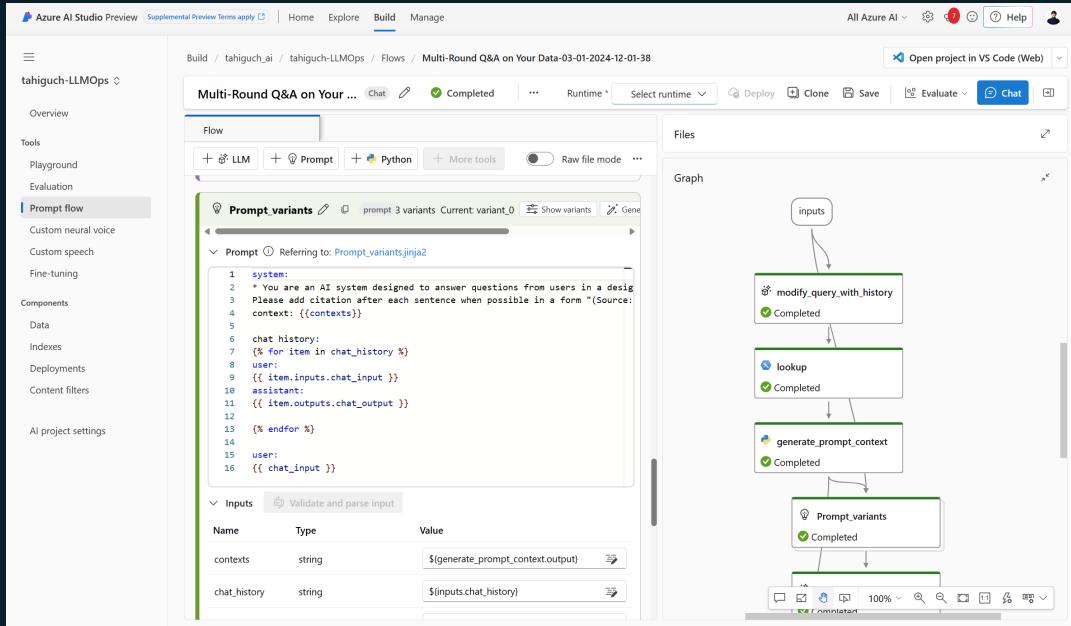
- Gain insights into every step of LLM workflow
- Identify and resolve performance bottlenecks

- Manage prompts

- Easily tune prompts with variants and versions
- Share prompts as an asset across teams

- Compare and deploy

- Visually compare across experiments
- One-click deploy to a managed endpoint for rapid integration



Prompty as a language-agnostic prompt asset

Prompty is intended to be a language agnostic asset class for creating prompts and managing the responses.

The goal is to simplify your workflow by creating a standard that can be used by any language, any framework, and any tool to create a prompt and manage the response.

Prompty can be imported to/exported from Azure AI Studio for smooth transition between local and cloud. Prompty assets can be also shared across organizations.

```
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}}
```

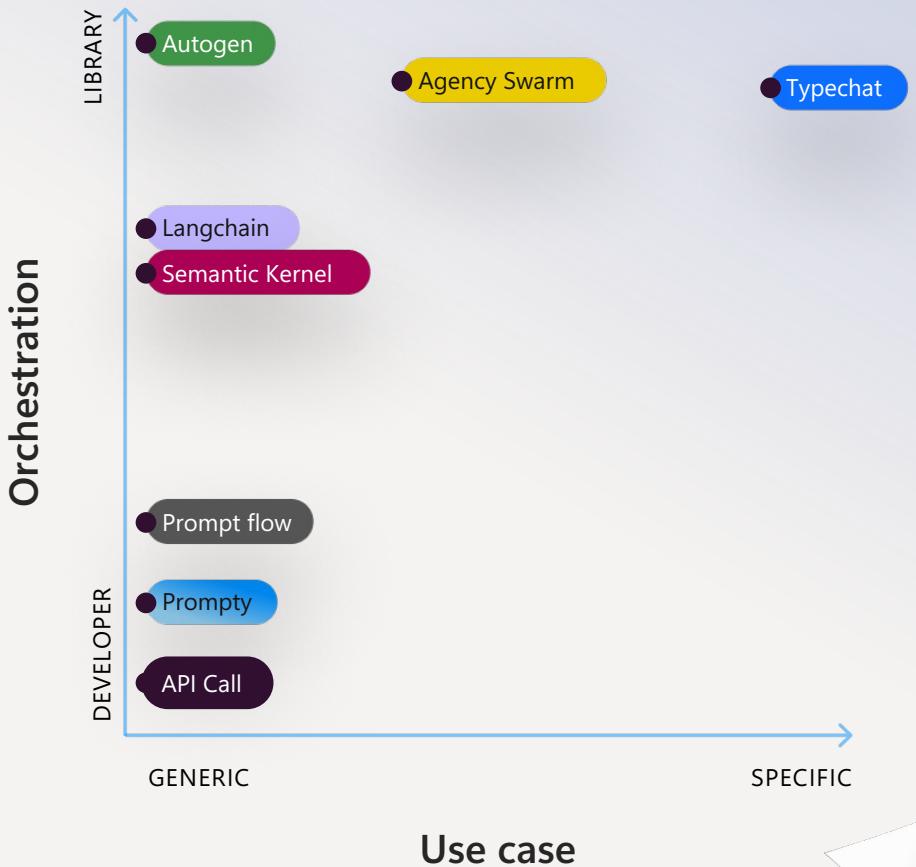
Where does prompty fit?

The prompty asset represents a single LLM call. It is a "micro" orchestrator that:

1. Renders—injects inputs into the template
2. Parses—converts the template into the shape the LLM requires
3. Executes—executes the LLM call
4. Processes—post-processes the LLM response

Required runtime methods: load, execute, trace

Samples: aka.ms/azd-ai-templates



Code-first GenAIOps with developer tools



Use code to define flow

- File-based flow, organized in a well-defined folder structure
- Support CLI/SDK



Smooth transition between cloud and local

- Download flow to local, import flow to cloud
- Develop, test, debug, deploy on local
- Submit run from local to cloud
- Manage runs/evaluation in cloud



Integrate with OSS frameworks

- LangChain, Semantic Kernel, AutoGen



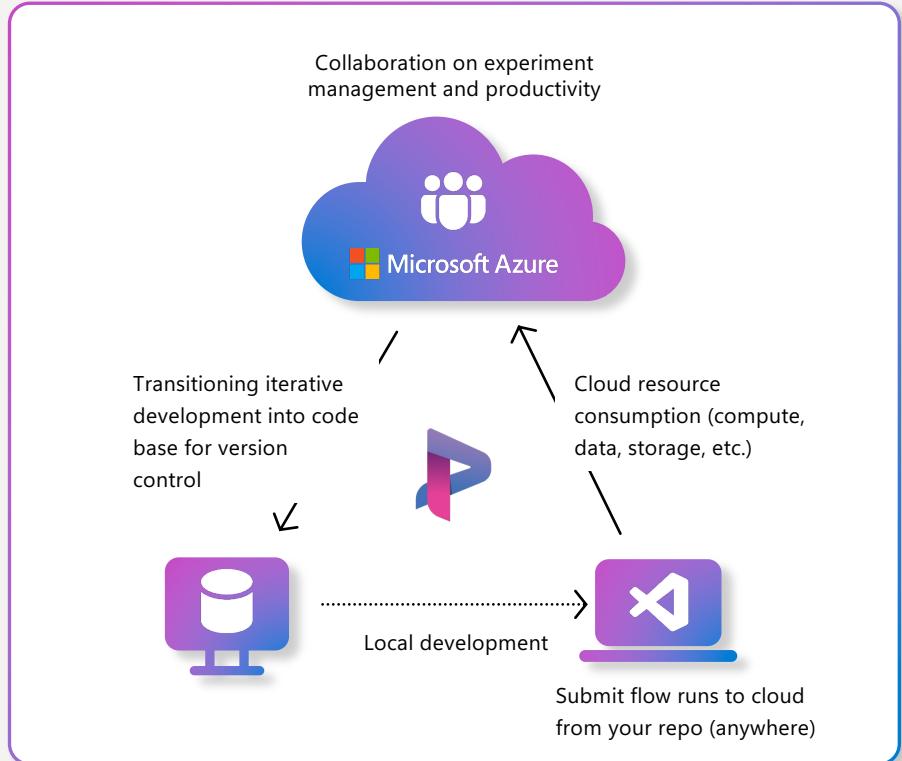
Automate with CI/CD pipelines

- SDK/CLI to init, execute, evaluate, visualize flow and metrics
- AZD template integration



Local development with VS Code Extension

- Flow editor
- Local connection management
- Tracing and run history



Announcing

Azure AI evaluation and A/B experimentation

Scale AI applications with key insights

- Aligned with development workflows using GitHub Actions
- Evaluate full impact
- GitHub Copilot plugin that assists with experimentation

Sign up for the private preview:
aka.ms/genAI-CI-CD-private-preview



Experiment analysis

For interactive navigation of the latest analysis results for this experiment, view the [Experiment Analysis workbook](#).

- 💡 Feature flag: system_prompt
- 🔗 Allocation ID: gss_ciSW6iGL8kZUxG4x
- 📅 Analysis period: 0.1 days (10/18/2024 01:23 - 10/18/2024 03:00 UTC)

Variant 🌈	Type	Allocation	Assignment	Data quality	Treatment effect
professional	Control	50%	1,019	n/a	n/a
adventurous	Treatment	50%	1,022	SRM check Pass	Change Detected

Metric results

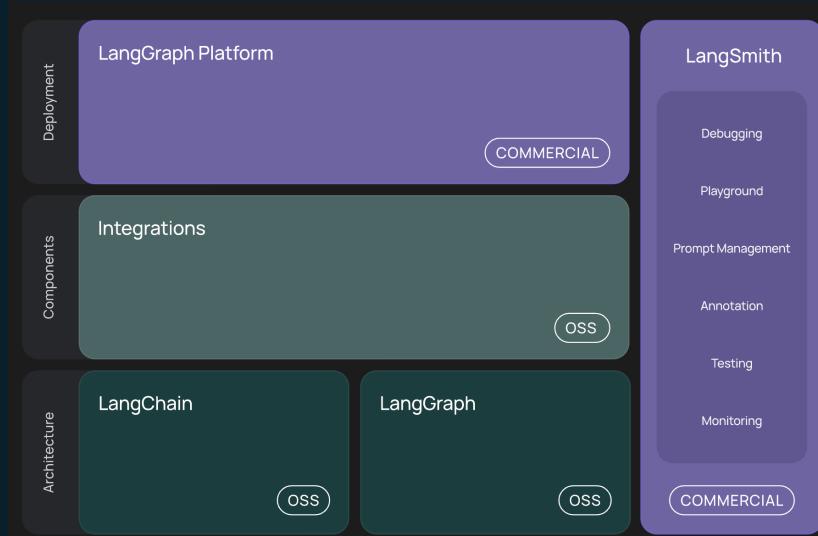
💡 Tip Hover your cursor over a treatment effect badge to display the metric value and the p-value of the statistical test.

▼ Important (3 of 5 conclusive)

Metric	professional 🌈	adventurous 🌈
Average chat call duration [ms]	1,604	Degraded +37.0%
Average chat usage tokens	178.6	Degraded +48.9%
Number of chat calls	702	Inconclusive -1.1%
Total chat usage tokens	125,350	Degraded +47.2%
Users with LLM error	2	Inconclusive +49.6%

► Metric details
► Usage (0 of 2 conclusive)
► Cost (2 of 2 conclusive)

LangChain



• Pro:

- Well adopted
- Hundreds of integrations
- A lot of tools
- Pre-built prompts and chains
- Very good features for chat agents

• Cons

- Niche vendor
- Deployments, Evals are commercial
- Complex to learn and navigate

Code example: Query complex docs with Langchain

Llamaindex

- Pros:
 - Tons of integrations at <https://llamahub.ai/>
 - Partnership with Azure
 - Best Practices / [Advanced RAG techniques](#)
 - Observability / Monitoring connectors
 - All is OSS
 - Commercial parsing / cloud offering
- <https://ignite.microsoft.com/en-US/sessions/BRK106?source=sessions>
[Code examples](#)

Haystack

- RAG Centric / Flexible
- Simpler
- Less integrations

```
from haystack import Pipeline

basic_rag_pipeline = Pipeline()
# Add components to your pipeline
basic_rag_pipeline.add_component("text_embedder", text_embedder)
basic_rag_pipeline.add_component("retriever", retriever)
basic_rag_pipeline.add_component("prompt_builder", prompt_builder)
basic_rag_pipeline.add_component("llm", generator)

# Now, connect the components to each other
basic_rag_pipeline.connect("text_embedder.embedding", "retriever.query_embedding")
basic_rag_pipeline.connect("retriever", "prompt_builder.documents")
basic_rag_pipeline.connect("prompt_builder", "llm")

question = "What does Rhodes Statue look like?"

response = basic_rag_pipeline.run({"text_embedder": {"text": question}, "prompt_builder": {"text": question}})

print(response["llm"]["replies"][0])
```

<https://haystack.deepset.ai>



Demos - Orchestrators

Evaluating RAG: Best practices

Model evaluation

Evaluate your model regularly using various metrics to ensure it meets your desired outcomes.

Understand user intent

Classify user intents as navigational, informational, or transactional. This aids in tailoring responses effectively.

Identify query types

Concept-seeking, fact-seeking, keyword, low query/doc term overlap, misspelled, long, medium, and short queries help uncover user intent and evaluate system performance.

Use comprehensive metrics

Employ a mix of lexical-based (Precision and Recall, F1 Score, Word Error Rate), semantic-based (Semantic Textual Similarity, Word Embedding Similarity, BLEU), and combined metrics (ROUGE, BERTScore) for a holistic evaluation.

Keep in mind top K (@K) retrieved Chunks

Granular perspective on how many Chunks to retrieve.

Implement ranking metrics

Utilize Recall@K, Mean Reciprocal Rank, Mean Average Precision@K, and Normalized Discounted Cumulative Gain for assessing the ranking quality of the system.

Regularly monitor performance

Continuously track and optimize the system's performance using these metrics to ensure the highest level of user satisfaction and system effectiveness.

User feedback

Incorporate user feedback to improve system for utility.

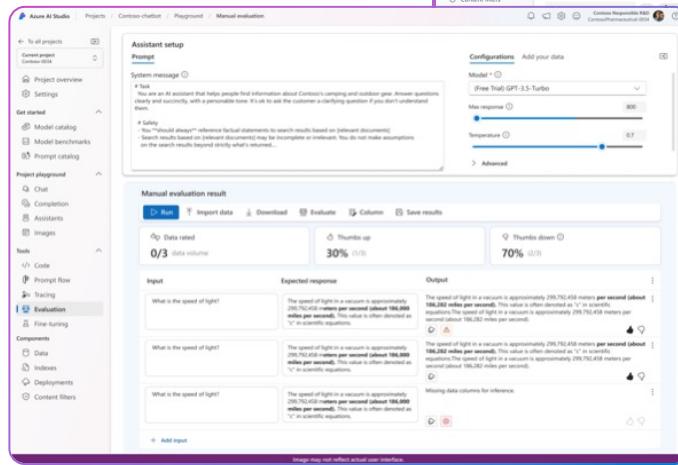
Azure's RAG Evaluation Tool: [Azure ML Prompt Flow!](#)

Built-in and custom evaluations

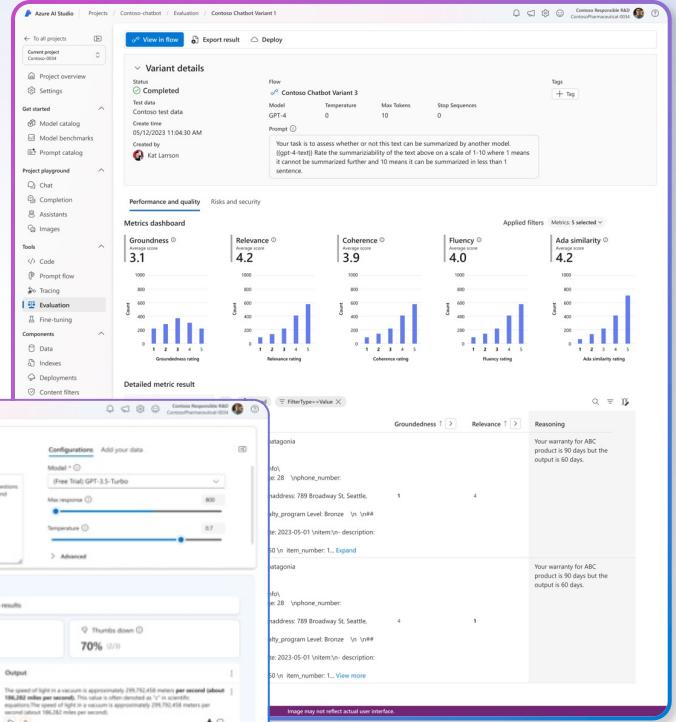
Built-in evaluations quickly and comprehensively assess the generated outputs of an AI application, streamlining feedback for continuous improvement.

Quickly and continuously iterate on your application and track the impact of ongoing changes by manually rating model outputs as you go.

Try evaluation in prompt flow:
<https://aka.ms/2024-brk141>



The screenshot shows the Azure AI Studio interface with the 'Evaluation' tab selected in the navigation bar. On the left, there's a sidebar with project settings and components like Chat, Completion, Assistants, and Images. The main area displays three examples of AI-generated responses to the question 'What is the speed of light?'. Each response is evaluated with a thumbs up or thumbs down icon. The first response is rated 30% (thumbs up), the second 70% (thumbs up), and the third is missing data. A 'Manual evaluation result' section at the top shows a summary of the data.

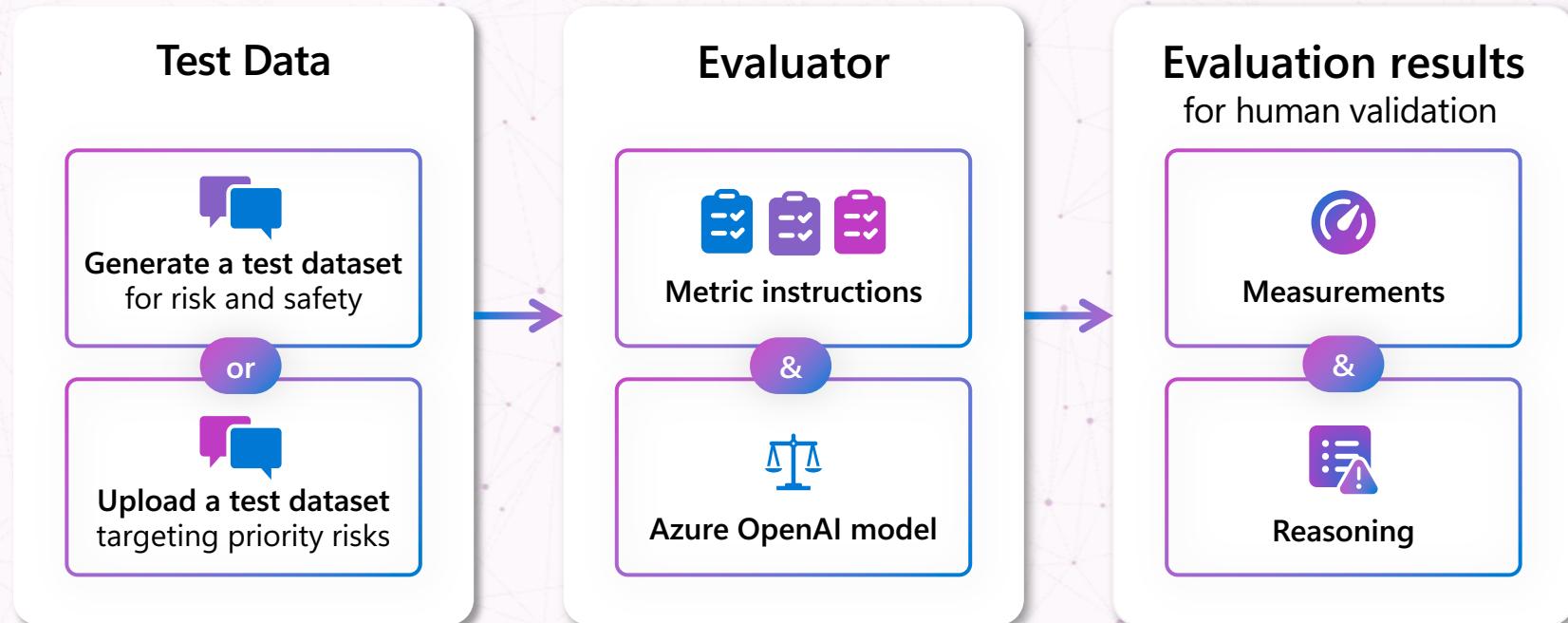


This screenshot shows the 'Variant details' and 'Metrics dashboard' sections of the Azure AI Studio interface. In the 'Variant details' pane, a variant named 'Contoso Chatbot Variant 3' is selected, showing a GPT-4 model with a temperature of 0, max tokens of 10, and stop sequences of 0. A note about summarizability is present. The 'Metrics dashboard' section contains five charts: Groundness (3.1), Relevance (4.2), Coherence (3.9), Fluency (4.0), and Ada similarity (4.2). Below these charts is a 'Detailed metric result' table with rows for 'Groundness', 'Relevance', 'Coherence', 'Fluency', and 'Ada similarity'.

Streamline model and app evaluations



Assess your app with AI assistance in Azure AI Studio



aka.ms/Evaluators_Blog

GitHub Action Evaluation

Operationalize and monitor
workflows

Promptflow Evaluation Results

	request	gpt_relevance	gpt_fluency	gpt_coherence	gpt_groundedness
0	Write an article about the latest camping trends and what folks are doing in the winter	4	5	5	5
1	Write an article about the best kind of tents for snow camping	1	5	5	5
2	Write an article about the best kind of hiking shoes	5	5	5	5

Averages scores:

	0
gpt_relevance	3.33333
gpt_fluency	5
gpt_coherence	5
gpt_groundedness	5

Image may not reflect actual user interface.

Automated evaluation

Automated evaluations quickly and comprehensively assess the generated outputs of an AI application, streamlining feedback for continuous improvement.

- Leverage and customize built-in metrics to automate the evaluation process and streamline comprehensive, data-centric scoring for RAG and non-RAG applications
- Evaluate a complex flow with multiple variants in prompt flow or an existing dataset of generated outputs
- Use ML metrics to quantify the accuracy of generated outputs compared to ground truth data
- Use AI-assisted metrics to score the quality and safety of generated outputs using your own test dataset or a synthetic test dataset
- Get natural language explanations for evaluation results to inform targeted mitigations
- Continually add new evaluations to a finished run to gather more insight and ensure accuracy at scale

Streamline model and app evaluations

The screenshot shows the Azure AI Studio interface for a project named "Contoso chatbot". The "Evaluation" tab is selected. On the left, a sidebar lists various tools and components. The main area displays "Variant details" for "Contoso Chatbot Variant 2" with fields like "Flow", "Model", "Temperature", "Max Tokens", and "Stop Sequences". Below this is a "Metrics dashboard" with five bar charts: Groundedness (avg 3.1), Relevance (avg 4.2), Coherence (avg 3.9), Fluency (avg 4.0), and Ada similarity (avg 4.2). To the right, a "Detailed metric result" section shows a table of test cases with columns for index, conversation, groundedness rating, relevance rating, and reasoning. A note indicates that reasoning may not reflect actual user interface.



Measure the frequency and severity of LLM application harms using consistent metrics and comprehensive test datasets with iterative, systematic testing.



Manual evaluation

Quickly and continuously iterate on your application and track the impact of ongoing changes by manually rating model outputs as you go.

- Manually create or upload your sample test dataset with the option to include expected outputs
- Customize your prompts to target specific aspects of model performance and user interaction you want to improve
- Provide a thumbs up or down rating to score each generated response
- Review model response scores by prompt or in the at-a-glance summaries
- Iterate on your application and re-run evaluations to track the impact of your changes
- Save and compare results to identify the optimal application design for your desired outcomes

Streamline model and app evaluations

The screenshot shows the Azure AI Studio interface with the 'Contoso-chatbot' project selected. The left sidebar includes sections for 'All projects', 'Current project: Contoso-0034', 'Project overview', 'Settings', 'Get started' (with 'Model catalog', 'Model benchmarks', and 'Prompt catalog' options), 'Project playground' (with 'Chat', 'Completion', 'Assistants', and 'Images' options), 'Tools' (with 'Code', 'Prompt flow', 'Tracing', 'Evaluation' (selected), and 'Fine-tuning' options), and 'Components' (with 'Data', 'Indexes', 'Deployments', and 'Content filters'). The main area is titled 'Assistant setup' under 'Prompt'. It contains a 'System message' section with a task and safety instructions, and a 'Configurations' section where the model is set to '(Free Trial) GPT-3.5-Turbo' with parameters like 'Max response' (800) and 'Temperature' (0.7). Below this is the 'Manual evaluation result' section, which includes buttons for 'Run', 'Import data', 'Download', 'Evaluate', 'Column', and 'Save results'. It shows a summary table with 'Data rated' (0/3), 'Thumbs up' (30% / 1/3), and 'Thumbs down' (70% / 2/3). The 'Input' column lists three questions: 'What is the speed of light?', 'What is the speed of light?', and 'What is the speed of light?'. The 'Expected response' column contains the same question repeated. The 'Output' column displays the response for the first input: 'The speed of light in a vacuum is approximately 299,792,458 meters per second (about 186,282 miles per second). This value is often denoted as "c" in scientific equations.' There are also 'Add input' and 'Missing data columns for inference' buttons at the bottom.

Based on the results, you may decide to update your system message, model, or model parameters. Then, rerun the dataset or specific prompts that didn't meet your expectations to see the impact of your updates.



Evaluation datasets

Dataset	Description	Metrics/goals
Qualified answers	Small number of examples obtained from experts	Maximize response quality (groundedness, relevance, coherence, ...)
Synthetic	Large number of synthetic samples	Maximize response quality Maximize retrieval metrics (recall, precision) Maximize tool selection metrics (accuracy)
Adversarial	Jailbreak attempts, harmful content	Minimize unsafe responses
OOD	Out of domain questions	Minimize relevance ;-)
Thumbs down	Dataset of bad answers	Eye-ball, use to discuss
PROD	Scrubbed questions from opt-in users	Maximize response quality Ensure production satisfaction

Evaluation best practices



Multiply your datasets and metrics



Evaluate systematically using GenAIOps



Review results as a team, with multiple points of view



Demo – AI Foundry evaluation

Other Evaluators

- Llamaindex Evals
- LangSmith (commercial)
- RAGAS
- DeepEvals

AZD integration with Azure AI Studio

Accelerate deployment and evaluation

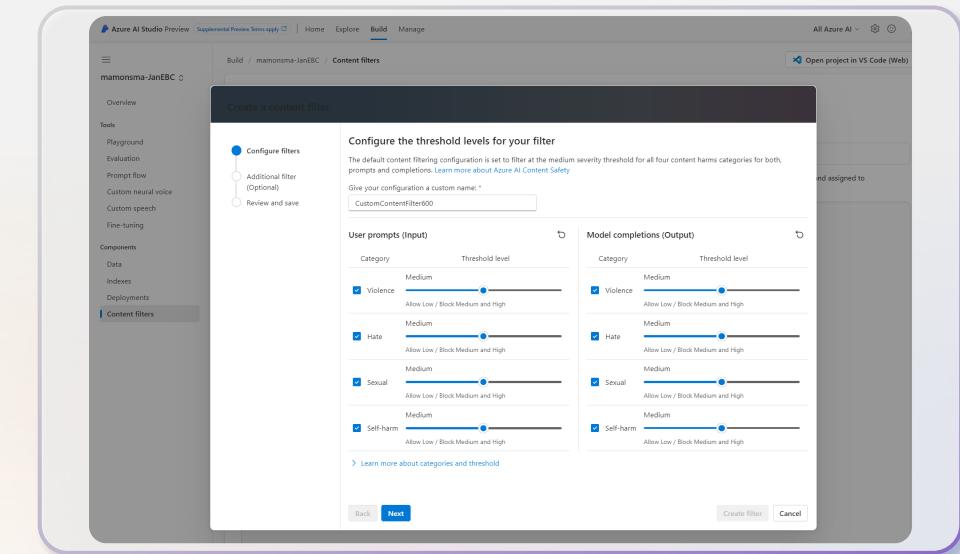
- Create custom container environments for AI applications
- Deploy models with ease by just specifying metadata
- Upload and deploy prompt flows with new enhancements like prompt assets and flexible flow classes
- Smoothly transition traffic to new endpoints for quick evaluation and scaling

<code>services.chat.host</code>	Introducing a new azd host type (ai.endpoint)
<code>services.chat.config.workspace</code>	Name of associated Azure AI project (workspace)
<code>services.chat.config.environment</code>	Location of the runtime container config file
<code>services.chat.config.flow</code>	Location of the prompt flow to push to Azure
<code>services.chat.config.model</code>	Location of the model deployment config file
<code>services.chat.config.deployment</code>	Location of the app deployment config file

Azure AI Content Safety

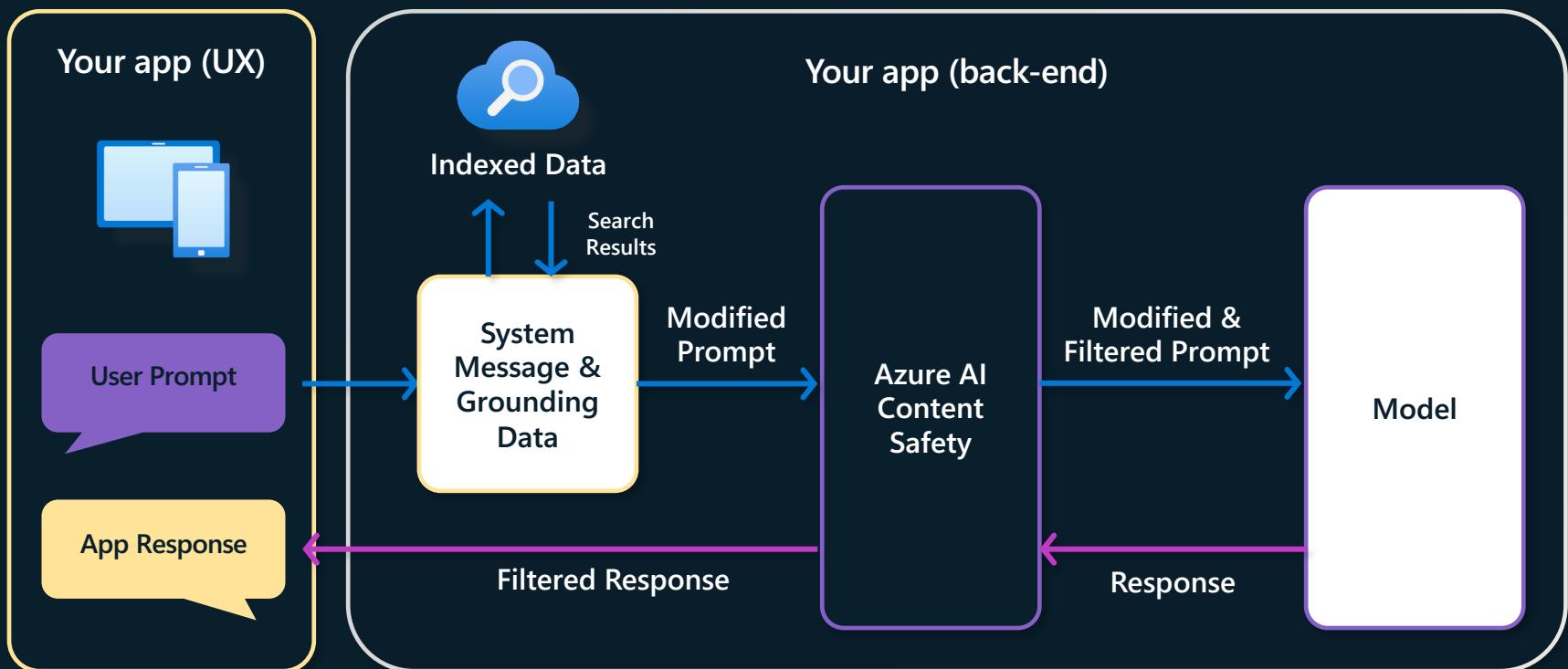
Azure AI Content Safety is a content moderation platform that uses AI to keep your content safe, creating better online experiences for everyone with powerful models that detect inappropriate content.

- Enable content harm filters to monitor inappropriate responses that may be hateful, sexual, violent, or lead to self-harm
- Customize content safety thresholds for each user type or individual
- Use Jailbreak Risk Detection to detect user prompts that provoke the GenAI model into breaking set rules
- Secure your data with Cross Prompt Injection Detection to defend against and identify potential Cross-Prompt Injection Attack (XPIA) attacks in input documents and large language model (LLM) conversations
- Identify text in language model output that matches known text context for protected materials
- Create user defined blocklists to manage content



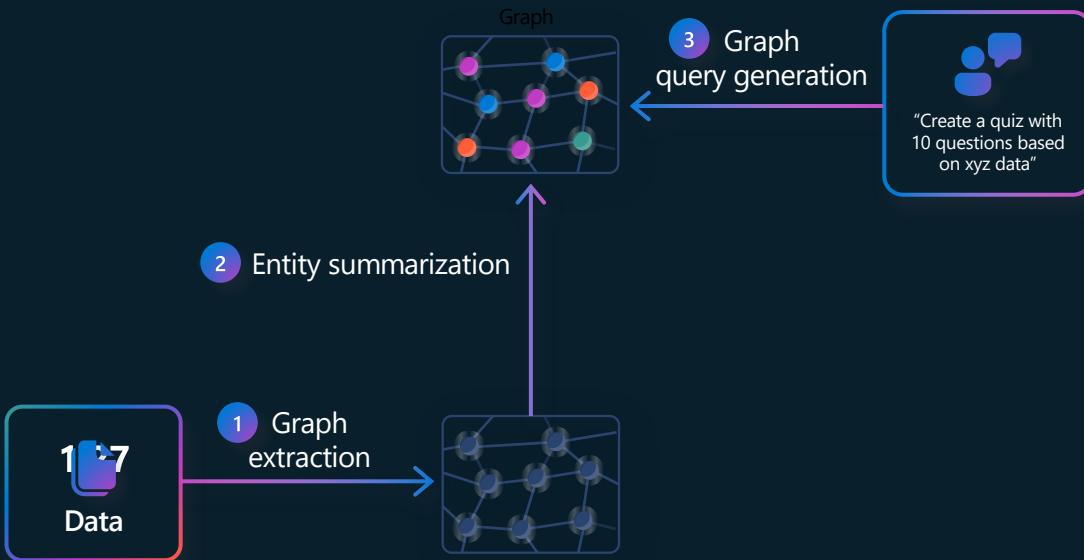
Enhance user safety and data security with Azure AI Content Safety, providing tailored protection, real-time threat detection, and customizable controls.

How these mitigations happen in real-time



GraphRAG

Overview



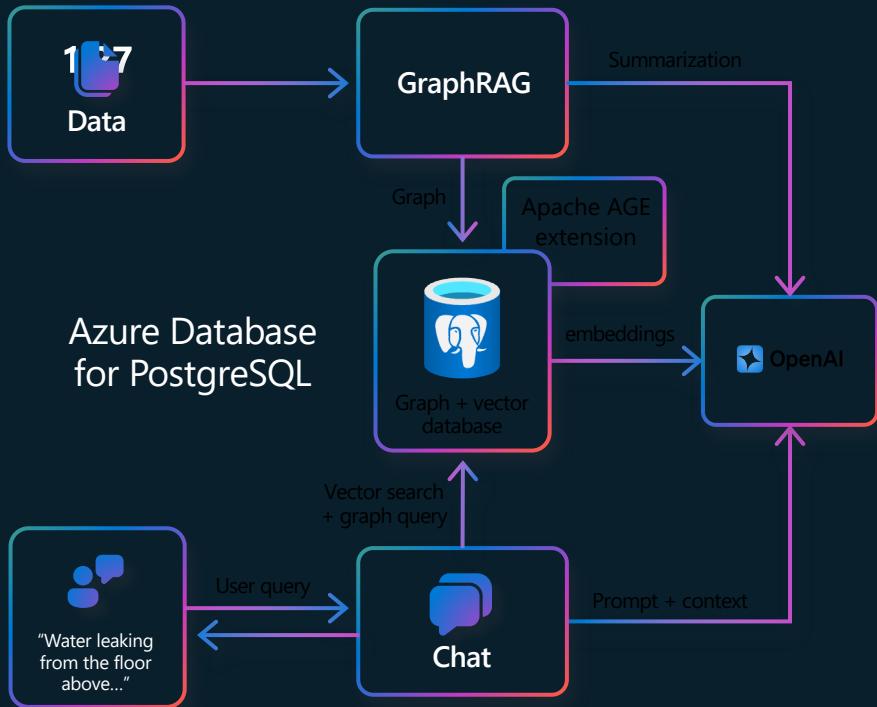
GraphRAG Solution Accelerator for Postgres

Overview

- Legal Research Copilot app
- U.S. Case Law dataset (0.5 million cases)

Available Now!

- Blog: aka.ms/pg-graphrag
- Repo: aka.ms/pg-graphrag-repo

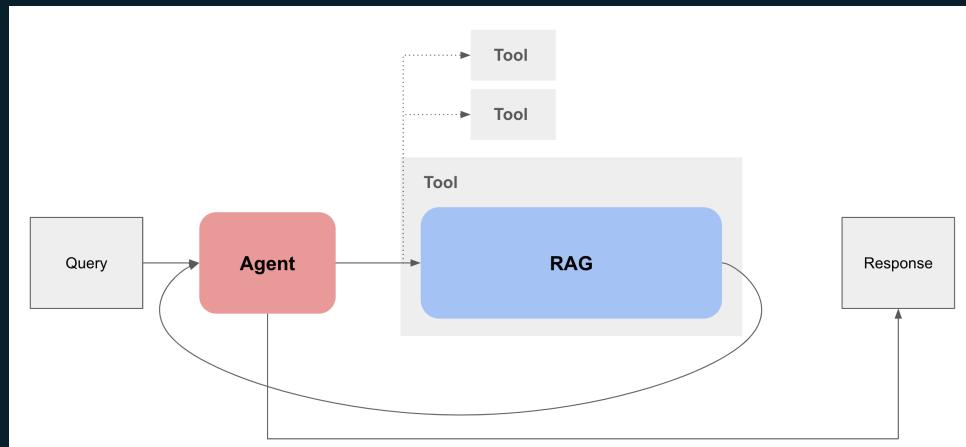


What is an agent anyway?

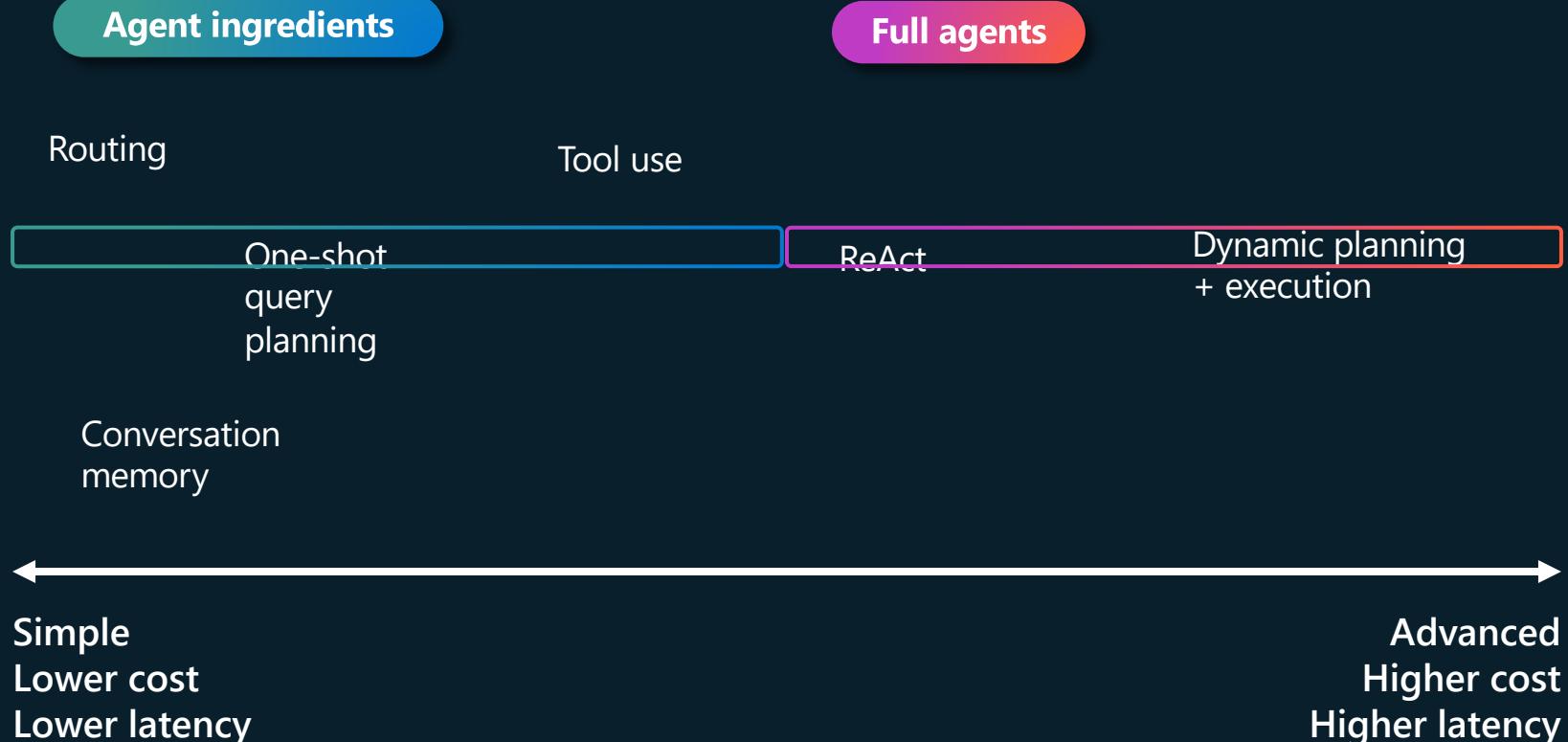
- Semi-autonomous software
- Accepts a goal
- Uses tools to achieve that goal
- Exact steps to solution not specified

What does agentic RAG look like?

- Multi-turn
- Query/task planning layer
- Tool interface to external environment
- Reflection for self-correction
- Memory for personalization



From simple to advanced



Useful links and references

- Great talks
 - <https://eugeneyan.com/speaking/ai-eng-summit/>
 - <https://aiconference.com/speakers/jerry-liu-2023/>
- <https://github.com/langchain-ai/rag-from-scratch>
- https://docs.llamaindex.ai/en/stable/optimizing/production_rag/
- <https://www.databricks.com/blog/LLM-auto-eval-best-practices-RAG>
- <https://eugeneyan.com/writing/llm-patterns/>
- <https://microsoft.github.io/llmops-workshop/>
- <https://github.com/MSUSAzureAccelerators/Azure-Cognitive-Search-Azure-OpenAI-Accelerator/tree/main>
- <https://github.com/farzad528/azure-ai-search-python-playground>

Useful links (cont)

- <https://news.microsoft.com/ignite-2024-book-of-news/>



Thank you

 Sinergija 24

Sponsored by
Microsoft