



Azure Knowledge Retrieval & Conversation

Readme

Generative AI (Artificial Intelligence) – Microsoft Azure RAG (Retrieval Augmented Generation) - Read.me

Table of contents

Introduction	2
Problems with LLM (Large Language Models) and the Solution	3
Solution	3
Use Case of RAG.....	3
Admin user functions - via Microsoft Azure portal.....	5
End user functions	5
Microsoft Azure (RAG) Package Design	5
Package actions.....	5
Authentication	5
Create Blob Storage Account.....	6
Create Blob Container.....	8
Upload File(s) to Blob Container.....	10
Create Azure AI Search Service	13
Add Data Source & Ingest Content	15
Ask Questions.....	17
Setup / Important Points / limitations to be considered:	19
Microsoft Azure RAG Knowledge Documentation	19
Azure RAG Package support:	19

Introduction

This document explains the overview of Generative AI and use of RAG to augment the query response generation using LLM or foundation models. This also explains how our AI platform – Microsoft Azure helps implementing RAG concept and how the package is built using its APIs.

Retrieval Augmented Generation, or **RAG**, is an architectural approach that can improve the efficacy of large language model (LLM) applications by leveraging custom data. This is done by



retrieving data/documents relevant to a question or task and providing them as context for the LLM.

Problems with LLM (Large Language Models) and the Solution

Problems:

1. No source.
2. LLM models do not know your data.
3. Doesn't answer recent data, ChatGPT knowledge is limited to Sep'21 data.
4. Doesn't answer company specific data like how many employees joined last month?
5. Cost associated with any LLM.
6. Privacy and Security concerns.

Solution

An easy and popular way to use your own data is to provide it as part of the prompt with which you query the LLM model. This is called retrieval augmented generation (RAG), as you would retrieve the relevant data and use it as augmented context for the LLM. Instead of relying solely on knowledge derived from the training data, a RAG workflow pulls relevant information and connects static LLMs (Large Language Models) with real-time data retrieval.

With RAG architecture, organizations can deploy any LLM model and augment it to return relevant results for their organization by giving it a small amount of their data without the costs and time of fine-tuning or pretraining the model.

Use Case of RAG

There are many different use cases for RAG. Commonly used are:

1. **Ticket Submission and Initial Response:** Customers submit support tickets to Automation Anywhere, describing their issues. Their system processes the ticket and forwards the extracted message to AAI Enterprise Knowledge API. With a rich knowledge base from diverse sources (PDFs, Office, JSON, HTML, XML, Knowledge Portals, etc.), the API assesses the message and generates initial responses—acknowledgment, info, or detailed inquiries.
2. **Iterative Conversations:** During the conversation, customers share more details, queries, or clarifications. Using natural language processing and context, the API generates and emails responses directly to customers, efficiently resolving tickets without human intervention.
3. **Human-Agent Interaction:** The collaborative effort between the API and human agent ensures that complex issues are handled effectively, combining the efficiency of automation with no or the least amount of human touch.
4. **Data Analysis and Insights:** The company gathers valuable data from interactions, identifying common issues and customer satisfaction levels. These insights are

instrumental in enhancing customer support strategies and guiding product improvements.

5. **Continuous Iteration and Enhanced Experience:** The company continually refines the API's responses and workflows based on real-world usage, contributing to an enhanced customer experience. Customers receive timely, accurate, and helpful support, solidifying brand loyalty and customer satisfaction.

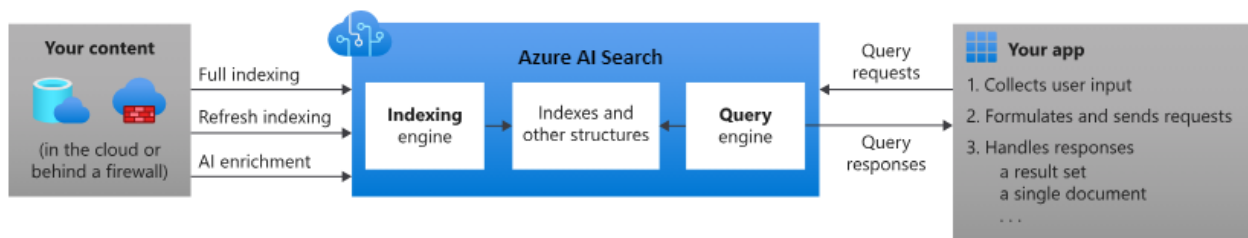
Why Microsoft's Cognitive Search (RAG) Service?

Azure AI Search (formerly known as "Azure Cognitive Search") provides secure information retrieval at scale over user-owned content in traditional and generative AI search applications.

Information retrieval is foundational to any app that surfaces text and vectors. Common scenarios include catalog or document search, data exploration, and increasingly chat-style apps over proprietary grounding data. When you create a search service, you work with the following capabilities:

- A search engine for [vector search](#) and [full text](#) and [hybrid search](#) over a search index
- Rich indexing with [integrated data chunking and vectorization \(preview\)](#), [lexical analysis](#) for text, and [optional applied AI](#) for content extraction and transformation
- Rich query syntax for [vector queries](#), text search, [hybrid queries](#), fuzzy search, autocomplete, geo-search and others
- Azure scale, security, and reach
- Azure integration at the data layer, machine learning layer, Azure AI services and Azure OpenAI

Architecturally, a search service sits between the external data stores that contain your un-indexed data, and your client app that sends query requests to a search index and handles the response.



In your client app, the search experience is defined using APIs from Azure AI Search, and can include relevance tuning, semantic ranking, autocomplete, synonym matching, fuzzy matching, pattern matching, filter, and sort.

Across the Azure platform, Azure AI Search can integrate with other Azure services in the form of *indexers* that automate data ingestion/retrieval from Azure data sources, and *skillsets* that incorporate consumable AI from Azure AI services, such as image and natural language processing, or custom AI that you create in Azure Machine Learning or wrap inside Azure Functions.

Admin user functions - via Microsoft Azure portal

An admin user can:

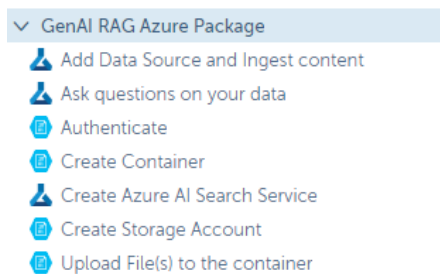
1. Implement RBAC (Role Based Access Control) mechanism using IAM (Identity Access Management) service. Admin can provide appropriate permissions / roles to the users in organization using IAM access management.
2. Create a new Storage Account and thus a new search service is associated with it.

End user functions

An end user can:

1. Add file(s) to storage container, subject to appropriate permission / role assigned via Admin interface. This will automatically synchronize the file contents in Vector database and Embeddings (indexed).
2. Can perform Q&A on the uploaded documents / files maintaining the query context using a query session.

Microsoft Azure (RAG) Package Design



Package actions

It is recommended to use the actions in the below logical order of creation:

Authentication

This action is used to generate authentication token which shall be used with the other actions to execute those steps.

GenAI RAG Azure Package: Authenticate

Create a storage account for RAG

Required bot agent version: 21.250 or above

Application (Client) Id

Credential Variable Insecure string

 (x)

Application (Client) Secret

Credential Variable Insecure string

 (x)

Directory (Tenant) Id

Credential Variable Insecure string

 (x)

Assign the output to variable

?? authentication (x)

Authorize token will be sent as an output

Sr, No.	Field name	Value	Input / Output	Description
1	Client ID	Azure Microsoft Client ID	Input - string	API key provided by the Admin.
2	Client Secret	Azure Microsoft Client Secret	Input - string	Secret key provided by the Admin.
3	Tenant ID	Azure Microsoft Tenant ID	Input - string	Tenant ID key provided by the Admin.
4	Auth token	Authorization token	Output - string	Generated auth token that can be used for subsequent steps.

***Note:** A dedicated App is created under AA Azure subscription with appropriate roles/permissions, please use above Client ID, Secret and Tenant ID to work with this package.

Create Blob Storage Account

This action is used to create Storage Account in Microsoft Azure that will shall be using in further down the steps to perform our RAG execution.

GenAI RAG Azure Package: Create Storage Account

Create a storage account for RAG

Required bot agent version: 21.250 or above

Access Token

Credential Variable Insecure string

” \$authentication\$

Authorization token generated from Authorize action.

Azure Subscription ID

Subscription ID in which you need to perform the operations.

Region

Resource Groups Name

Resource Group name in which you need to perform the operations.

Storage Account Name

Azure Blob storage resource that will has the container with the files you would like to use for data grounding.

Performance

☒ Standard

☐ Premium

Redundancy

Geo-redundant storage (GRS)

Tags (optional)

Dictionary Variable

Assign the output to variable

Dictionary of responses and other details

Multiple variables Dictionary

Sr, No.	Field name	Value	Input / Output	Description
1	Access Token	API key	Input - string	Token generated in Step 1.
2	Subscription ID	Azure subscription, click on dropdown to select it	Input – dropdown select	Azure subscription ID. We use this value for testing/POC purposes.
3	Region	Useast-2	Input – dropdown select	Region in which the storage account would be created.
4	Resource Group Name	Azure resource group name, click on dropdown to select it	Input – dropdown select	Resource group under which this storage would be created
5	Storage Account Name	Storage Name	Input - string	Any meaningful name.
6	Performance	Standard/Premium	Choose	https://learn.microsoft.com/en-us/answers/questions/864000/difference-between-premium-vs-standard-general-purpose
7	Redundancy	LRS, GRS, ZRS, GZRS	Choose	https://learn.microsoft.com/en-us/azure/storage/common/storage-redundancy
8	Tags	Any string value	Input – Dictionary	To attach any metadata like name: value.
9	Output	Keys: id, name, location	Output - Dictionary	Returns a dictionary with keys: id, name, location

Create Blob Container

This action is used to create a container within a storage account so that to upload file(s) in this container.

GenAI RAG Azure Package: Create Container

Create a blob container for RAG

Required bot agent version: 21.250 or above

Access Token

Credential Variable Insecure string

” \$authentication\$

Authorization token generated from Authorize action.

Azure Subscription ID

Subscription ID in which you need to perform the operations.

Region

Resource Groups Name

Resource Group name in which you need to perform the operations.

Storage Account Name

Select Azure Blob storage resource that has the container with the files you would like to use for data grounding.

Storage Container Name

Select storage container that contains the data to be used in creating a search index for grounding.

Metadata (optional)

Dictionary Variable

This dictionary is empty

Add

Assign the output to variable

Dictionary of responses and other details

Multiple variables Dictionary

containerStatus

Sr, No.	Field name	Value	Input / Output	Description
1	Access Token	API key	Input - string	Token generated in Step 1.
2	Subscription ID	Azure subscription, click on dropdown to select it	Input – dropdown select	Azure subscription ID. We use this value for testing/POC purposes.
3	Region	Useast-2	Input – dropdown select	Region in which the storage account would be created.
4	Resource Group Name	Azure resource group name, click on dropdown to select it	Input – dropdown select	Resource group under which this storage would be created.
5	Storage Account Name	Storage Name	Input – dropdown select	Select the storage account name under which this container is to be created.
6	Metadata	Any string value	Input – Dictionary	To attach any metadata like name: value.
7	Output	Keys: id, name, location	Output - Dictionary	Returns a dictionary with keys: id, name, location

Upload File(s) to Blob Container

This action is used to upload file(s) in the Storage Container created in previous step. This will not take files from the sub-folders within the provided folders.

GenAI RAG Azure Package: Upload File(s) to the container

Upload file(s) to azure for RAG

Required bot agent version: 21.250 or above

Access Token

Credential Variable Insecure string

” \$authentication\$

Authorization token generated from Authorize action.

Azure Subscription ID

Subscription ID in which you need to perform the operations.

Region

Resource Groups Name

Resource Group name in which you need to perform the operations.

Storage Account Name

Select Azure Blob storage resource that has the container with the files you would like to use for data grounding.

Storage Container Name

Select storage container that contains the data to be used in creating a search index for grounding.

Key to sign

Upload to folder (optional)

??

folder path in the container e.g. folder1 or folder1/folder2

List of local folder paths

List Variable

↑

↓

Type

?? String

Value at 0

...

(x)

Add

Local folder paths e.g. C:\folder1 or C:\folder1\folder2.

Metadata (optional)

Dictionary Variable

A

Assign the output to variable (optional)

Dictionary of responses and other details

Multiple variables Dictionary

A uploadDocumentStatus

Sr. No.	Field name	Value	Input / Output	Description
1	Access Token	API key	Input - string	Token generated in Step 1.
2	Subscription ID	Azure subscription, click on dropdown to select it	Input – dropdown select	Azure subscription ID. We use this value for testing/POC purposes.
3	Region	Useast-2	Input – dropdown select	Region in which the storage account would be created.

4	Resource Group Name	Azure resource group name, click on dropdown to select it	Input – dropdown select	Resource group under which this storage would be created.
5	Storage Account Name	Storage Name	Input – dropdown select	Select storage account name under which this container is to be create.
6	Storage Container Name	Container name	Input – dropdown select	Select container name to which files are to be uploaded.
6	Key to sign	Private key to sign to certificate	Input – String	Generated as output from previous step.
7	Upload to folder	Container folder path	Input – String	Container folder path in the format Folder1 or Folder1/Folder2 etc.
8	List of local folders path	Local folders path	List – String	Provide the list of local folder paths. All the files within the folder(s) would get uploaded to Container.
9	Output	List of Dictionaries with Keys: id, name, location	Output - Dictionary	Returns a List of Dictionaries with Keys: id, name, location.

Create Azure AI Search Service

This action is used to create Azure AI Search Service that will help retrieve information from the file(s) we have uploaded for our respective queries.

Azure AI Search ([formerly known as "Azure Cognitive Search"](#)) provides secure information retrieval at scale over user-owned content in traditional and generative AI search applications.

GenAI RAG Azure Package: Create Azure AI Search Service

Create an Azure AI Search service resource.

Required bot agent version: 21.250 or above

Access Token

Credential Variable Insecure string

” \$authentication\$

Authorization token generated from Authorize action.

Azure Subscription ID

Subscription ID in which you need to perform the operations.

Region

Resource Group Name

Resource Group name in which you need to perform the operations.

Azure AI Search Service name

Provide the service name. It must only contain lowercase letters, digits or dashes, cannot use dash as the first two characters, cannot contain consecutive dashes, and is limited between 2 and 60 characters in length.

Dictionary of search service admin keys: primaryKey, secondaryKey and error. (optional)

Dictionary of search service admin keys.

Multiple variables Dictionary

aiServiceStatus

Sr, No.	Field name	Value	Input / Output	Description
1	Access Token	API key	Input - string	Token generated in Step 1.
2	Azure AI Search Service Name	Name	Input - string	Azure AI Search Service Name to be used for retrieval operations.

3	Subscription ID	Azure subscription, click on dropdown to select it	Input – dropdown select	Azure subscription ID. We use this value for testing/POC purposes.
4	Region	Useast-2	Input – dropdown select	Region in which the storage account would be created.
5	Resource Group Name	Azure resource group name, click on dropdown to select it	Input – dropdown select	Resource group under which this storage would be created.
6	Azure AI Search Service Name	Azure AI service Name	Input- String	Any meaningful name
7	Output	List of Dictionaries with Keys: primaryKey, secondaryKey, error	Output - Dictionary	Returns a List of Dictionaries with Keys: primaryKey, secondaryKey, error.

Add Data Source & Ingest Content

This action is used to add data source to our AI search service & then add content so it can retrieve data correctly.

Data ingestion involves loading data into a table in your cluster. Azure Data Explorer ensures data validity, converts formats as needed, and performs manipulations like schema matching, organization, indexing, encoding, and compression. Once ingested, data is available for query.

GenAI RAG Azure Package: Add Data Source and Ingest content

Indexes the given Data source and Ingest content for Search.

Required bot agent version: 21.250 or above

Access Token

Credential Variable Insecure string

” \$authentication\$

Authorization token generated from Authorize action.

Azure Subscription ID

Subscription ID in which you need to perform the operations.

Region

Resource Group Name

Resource Group name in which you need to perform the operations.

Storage Account Name

Select Azure Blob storage resource that has the container with the files you would like to use for data grounding.

Storage Container Name

Select storage container that contains the data to be used in creating a search index for grounding.

Sub folder name (optional)

Specify sub-folder name in the Container from which documents would be indexed.

Azure AI Search Service name

Select the Azure AI Search resource where the index used for grounding will be created.

Index Name

” azurepackageindex

Enter the index name that will be used to reference this data source. A new cognitive search index with the provided will be generated after data ingestion is complete. Name must start and end with alphanumeric characters and lowercase letters, digits or dashes.

Assigned to (optional)

” ingestStatus

Sr, No.	Field name	Value	Input / Output	Description
1	Access Token	API key	Input - string	Token generated in Step 1.
2	Azure AI Search Service Name	Name	Input - string	Azure AI Search Service Name to be used for retrieval operations.
3	Subscription ID	Azure subscription, click on	Input – dropdown select	Azure subscription ID. We use this value for testing/POC purposes.

		dropdown to select it		
4	Region	Useast-2	Input – dropdown select	Region in which the storage account would be created.
5	Resource Group Name	Azure resource group name, click on dropdown to select it	Input – dropdown select	Resource group under which this storage would be created.
6	Storage Account Name	Storage Name	Input – dropdown select	Select storage account name under which this container is to be create.
7	Storage Container Name	Container name	Input – dropdown select	Select container name to which files are to be uploaded.
8	Azure AI Search Service Name	Azure AI service Name	Input- String	Generated in previous step
9	Index Name	Azure Index Name	Input- String	Give a meaningful name.
10	Output	Output variable	Output - String	Returns a string output

Ask Questions

This action is used to query documents uploaded to the Microsoft Azure Blob Container.

GenAI RAG Azure Package: Ask questions on your data

Ask question on your data.

Required bot agent version: 21.250 or above

Access Token

Credential Variable Insecure string

” \$authentication\$

Authorization token generated from Authorize action.

Azure Subscription ID

Subscription ID in which you need to perform the operations.

Azure AI Search Service name

Select the Azure AI Search resource where the index used for grounding will be created.

Azure AI Search Service API Key

Credential Variable Insecure string

” \$aiServiceStatus{primaryKey}\$

Azure AI Search Service API Key

Azure AI Search Index name

Select the Azure AI Search Index where it is used for searching.

Query Type

” simple

Query type, possible values are 'simple' and 'semantic'.

Ask questions about your own data


” \$userQuestion\$

User question(s) to be asked to AI search service.

Assigned to

Dictionary of responses and citations. Dictionary keys: output, error.

Multiple variables Dictionary

 queryResponse

Sr, No.	Field name	Value	Input / Output	Description
1	Access Token	API key	Input - string	Token generated in Step 1.
2	User Query	User Query	Input - string	User query to get response using Azure RAG.

Setup / Important Points / limitations to be considered:

- To get your **API key** from Microsoft Azure, follow these steps or contact your administrator or IT support:
 - Log in to the [Microsoft Azure Portal](#).
 - Click App registrations.
 - Click New registration.
 - Give your application a name. This name will be visible to your end users.
 - Set the audience for the app to Accounts in any organizational directory and personal Microsoft accounts.
 - Review Microsoft's Platform Policies, then click Register.
- The max. 1000 characters can be inputted as the User query text.
- Azure container **supports the following file formats**: Plain text (.txt), HyperText Markup Language (.html), Microsoft Word document (.doc/.docx), Comma-separated values (.csv), Microsoft Excel spreadsheet (.xls/.xlsx) and Portable Document Format (.pdf).
- Below should be the logical package **Actions order**, in which, should get executed:
 - Authentication
 - Create Storage Account
 - Create Container
 - Upload file(s) to Container
 - Create Azure AI Search Service
 - Add Data Source & Ingest Content
 - Ask Questions & retrieve response

Microsoft Azure RAG Knowledge Documentation

<https://learn.microsoft.com/en-us/azure/search/search-what-is-azure-search>

Azure RAG Package support:

Pinkesh.achhodwala@automationanywhere.com

Himanshu.manjarawala@automationanywhere.com

Aditya.singhanian@automationanywhere.com

