Table of Contents

# Welcome to the Azure Cosmos DB OpenAI Python Dev Guide

## Pre-requisites

* [Azure account and subscription](https://azure.microsoft.com/free/) with Owner permissions
* [Python 3.11 or higher](https://www.python.org/?downloads)
* [Visual Studio Code](https://code.visualstudio.com/download)
* [Python extension for VS Code](https://marketplace.visualstudio.com/items?itemName=ms-python.python)
* [Jupyter Notebook extension for VS Code](https://marketplace.visualstudio.com/items?itemName=ms-toolsai.jupyter)
* [Docker Desktop](https://www.docker.com/products/docker-desktop/) with [WSL 2 backend (if on Windows)](https://learn.docker.com/desktop/wsl/)
* [Azure CLI](https://learn.microsoft.com/cli/azure/install-azure-cli)
* [Bicep CLI](https://learn.microsoft.com/azure/azure-resource-manager/bicep/install#install-manually)
* [Powershell](https://learn.microsoft.com/powershell/scripting/install/installing-powershell?view=powershell-7.3)

## Introduction

Lorem Ipsum

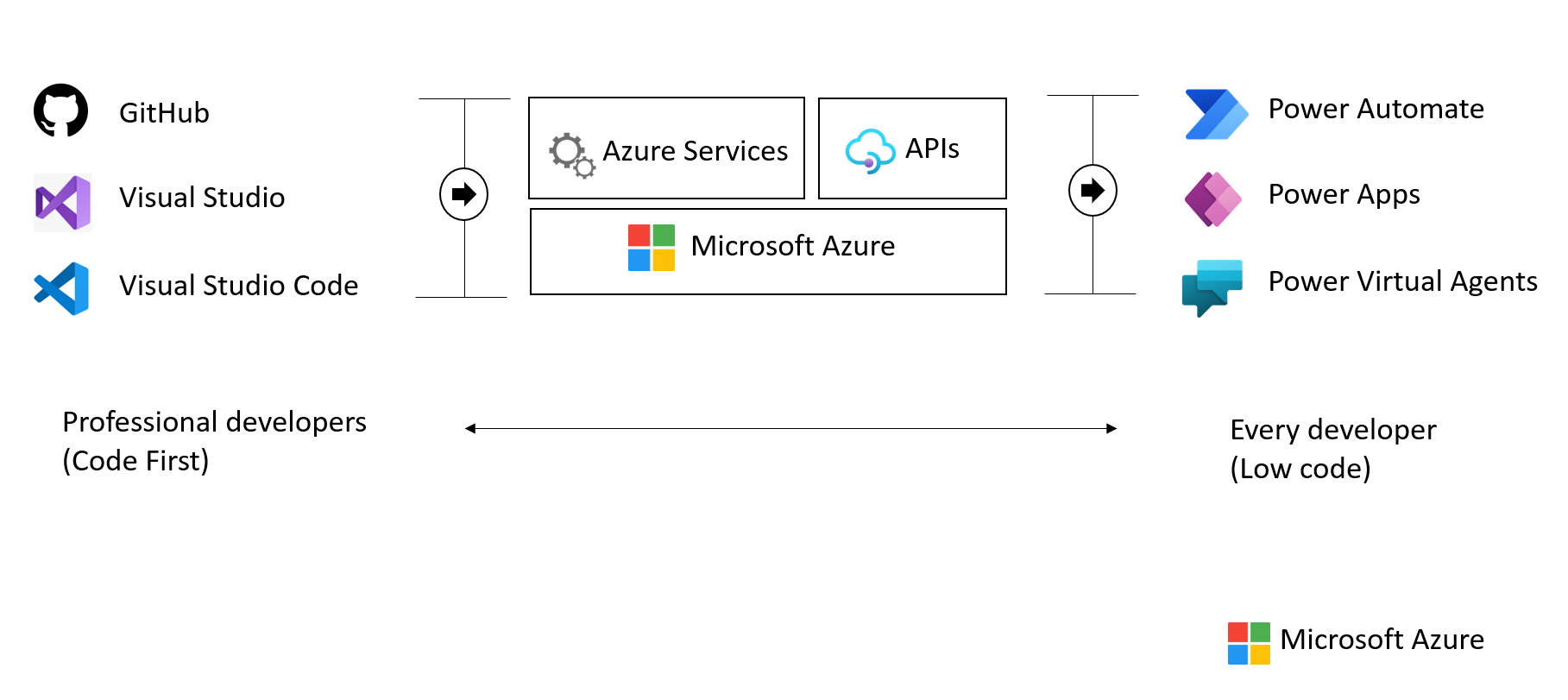
# Azure Overview

Millions of customers worldwide trust the Azure platform, and there are over 90,000 Cloud Solution Providers (CSPs) partnered with Microsoft to add extra benefits and services to the Azure platform. By leveraging Azure, organizations can easily modernize their applications, expedite application development, and adapt application requirements to meet the demands of their users.

## Advantages of choosing Azure

By offering solutions on Azure, ISVs can access one of the largest B2B markets in the world. Through the [Azure Partner Builder’s Program](https://partner.microsoft.com/marketing/azure-isv-technology-partners), Microsoft assists ISVs with the tools and platform to offer their solutions for customers to evaluate, purchase, and deploy with just a few clicks of the mouse.

Microsoft’s development suite includes such tools as the various [Visual Studio](https://visualstudio.microsoft.com/) products, [Azure DevOps](https://dev.azure.com/), [GitHub](https://github.com/), and low-code [Power Apps](https://powerapps.microsoft.com/). All of these contribute to Azure’s success and growth through their tight integrations with the Azure platform. Organizations that adopt modern tools are 65% more innovative, according to a [2020 McKinsey & Company report.](https://azure.microsoft.com/mediahandler/files/resourcefiles/developer-velocity-how-software-excellence-fuels-business-performance/Developer-Velocity-How-software-excellence-fuels-business-performance-v4.pdf)

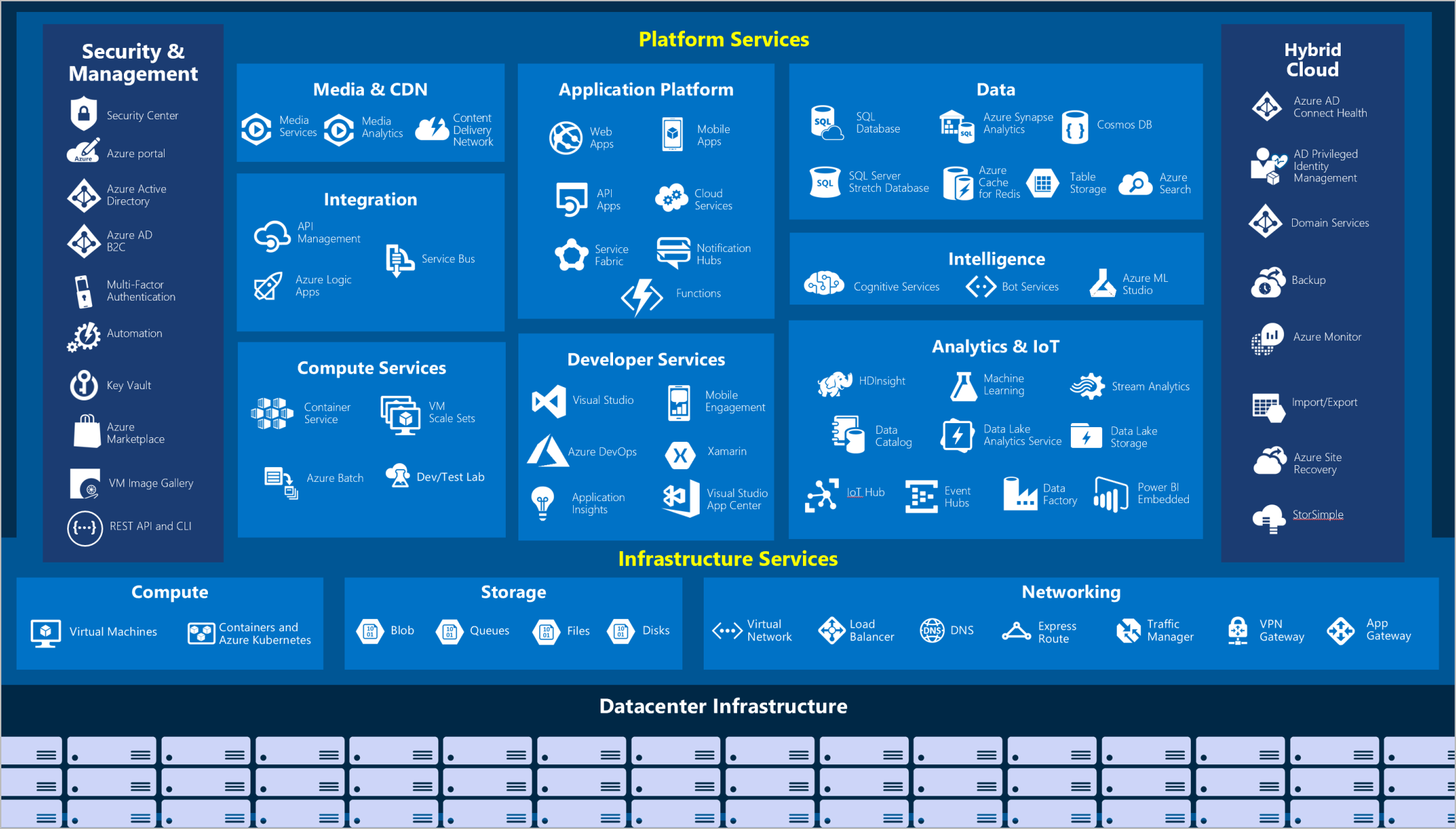


This image demonstrates common development tools on the Microsoft cloud platform to expedite application development.

To facilitate developers’ adoption of Azure, Microsoft offers a [free subscription](https://azure.microsoft.com/free/search/) with $200 credit, applicable for thirty days; year-long access to free quotas for popular services and access to always free Azure service tiers.

## Introduction to Azure resource management

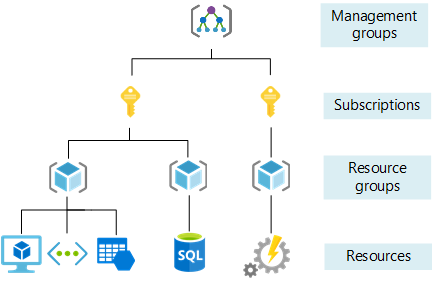
The [Azure Fundamentals Microsoft Learn Module](https://learn.microsoft.com/learn/modules/intro-to-azure-fundamentals/) demonstrates the different classifications of Azure Services. Moreover, Azure supports a variety of common tools, such as Visual Studio, PowerShell, and the Azure CLI, to manage Azure environments.



IaaS and PaaS Azure service classification and categories

### The Azure resource management hierarchy

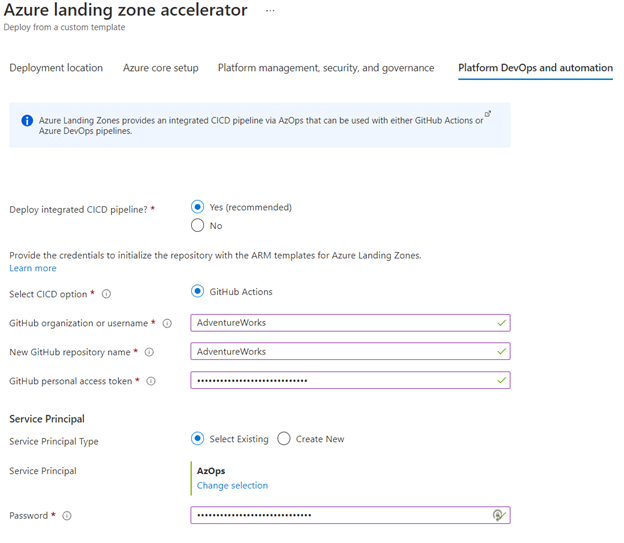
Azure provides a flexible resource hierarchy to simplify cost management and security. This hierarchy consists of four levels:

* [**Management groups**](https://learn.microsoft.com/azure/governance/management-groups/overview): Management groups consolidate multiple Azure subscriptions for compliance and security purposes.
* **Subscriptions**: Subscriptions govern cost control and access management. Azure users cannot provision Azure resources without a subscription.
* [**Resource groups**](https://learn.microsoft.com/azure/azure-resource-manager/management/manage-resource-groups-portal): Resource groups consolidate the individual Azure resources for a given deployment. All provisioned Azure resources belong to one resource group. In this guide, it will be required to provision a *resource group* in an *subscription* to hold the required resources.
  + Resource groups are placed in a geographic location that determines where metadata about that resource group is stored.
* **Resources**: An Azure resource is an instance of a service. An Azure resource belongs to one resource group located in one subscription.
  + Most Azure resources are provisioned in a particular region.
* 
* This image shows Azure resource scopes.

### Create landing zone

An [Azure landing zone](https://learn.microsoft.com/azure/cloud-adoption-framework/ready/landing-zone/) is the target environment defined as the final resting place of a cloud migration project. In most projects, the landing zone should be scripted via ARM templates for its initial setup. Finally, it should be customized with PowerShell or the Azure Portal to fit the workload’s needs. First-time Azure users will find creating and deploying to DEV and TEST environments easy.

To help organizations quickly move to Azure, Microsoft provides the Azure landing zone accelerator, which generates a landing zone ARM template according to an organization’s core needs, governance requirements, and automation setup. The landing zone accelerator is available in the Azure portal.

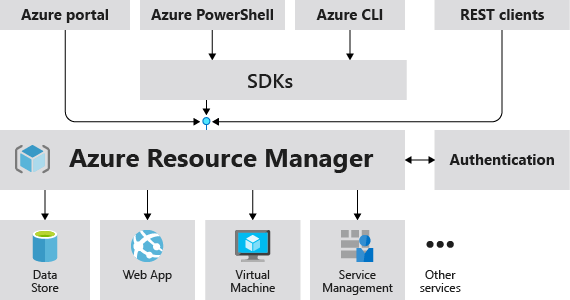


This image demonstrates the Azure landing zone accelerator in the Azure portal, and how organizations can optimize Azure for their needs and innovate.

### Automating and managing Azure services

When it comes to managing Azure resources, there are many potential options. [Azure Resource Manager](https://learn.microsoft.com/azure/azure-resource-manager/management/overview) is the deployment and management service for Azure. It provides a management layer that enables users to create, update, and delete resources in Azure subscriptions. Use management features, like access control, locks, and tags, to secure and organize resources after deployment.

All Azure management tools, including the [Azure CLI](https://learn.microsoft.com/cli/azure/what-is-azure-cli), [Azure PowerShell](https://learn.microsoft.com/powershell/azure/what-is-azure-powershell?view=azps-7.1.0) module, [Azure REST API](https://learn.microsoft.com/rest/api/azure/), and browser-based Portal, interact with the Azure Resource Manager layer and [Identity and access management (IAM)](https://learn.microsoft.com/azure/role-based-access-control/overview) security controls.

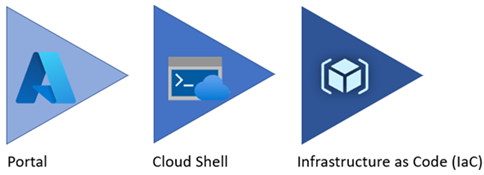


This image demonstrates how the Azure Resource Manager provides a robust, secure interface to Azure resources.

Access control to all Azure services is offered via the [Azure role-based access control (Azure RBAC)](https://learn.microsoft.com/azure/role-based-access-control/overview) natively built into the management platform. Azure RBAC is a system that provides fine-grained access management of Azure resources. Using Azure RBAC, it is possible to segregate duties within teams and grant only the amount of access to users that they need to perform their jobs.

### Azure management tools

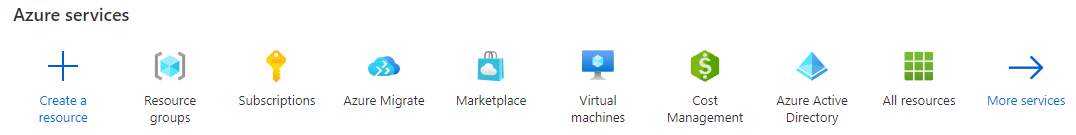
The flexibility and variety of Azure’s management tools make it intuitive for any user, irrespective of their skill level with specific technologies. As an individual’s skill level and administration needs mature, Azure has the right tools to match those needs.



Azure service management tool maturity progression.

#### Azure portal

As a new Azure user, the first resource a person will be exposed to is the Azure Portal. The **Azure Portal** gives developers and architects a view of the state of their Azure resources. It supports extensive user configuration and simplifies reporting. The [**Azure mobile app**](https://azure.microsoft.com/get-started/azure-portal/mobile-app/) provides similar features for users that are away from their main desktop or laptop.

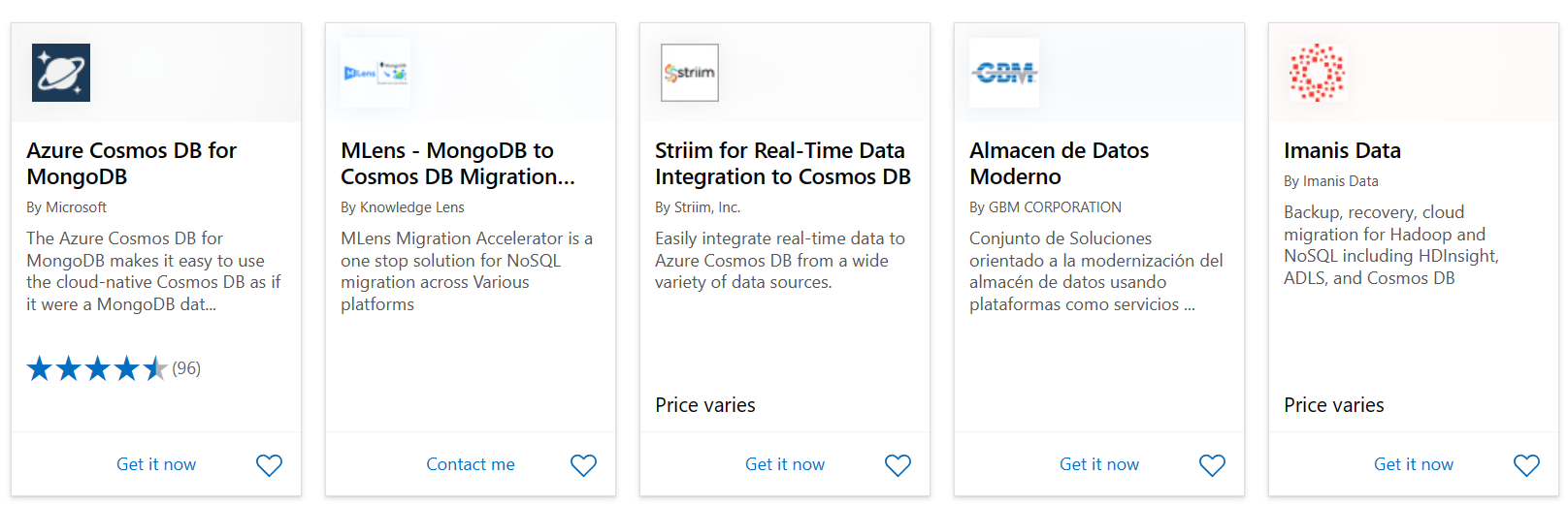


The picture shows the initial Azure service list.

Azure runs on a common framework of backend resource services, and every action taken in the Azure portal translates into a call to a set of backend APIs developed by the respective engineering team to read, create, modify, or delete resources.

##### Azure Marketplace

[Azure Marketplace](https://learn.microsoft.com/marketplace/azure-marketplace-overview) is an online store that contains thousands of IT software applications and services built by industry-leading technology companies. In Azure Marketplace, it is possible to find, try, buy, and deploy the software and services needed to build new solutions and manage the cloud infrastructure. The catalog includes solutions for different industries and technical areas, free trials, and consulting services from Microsoft partners.



The picture shows an example of Azure Marketplace search results.

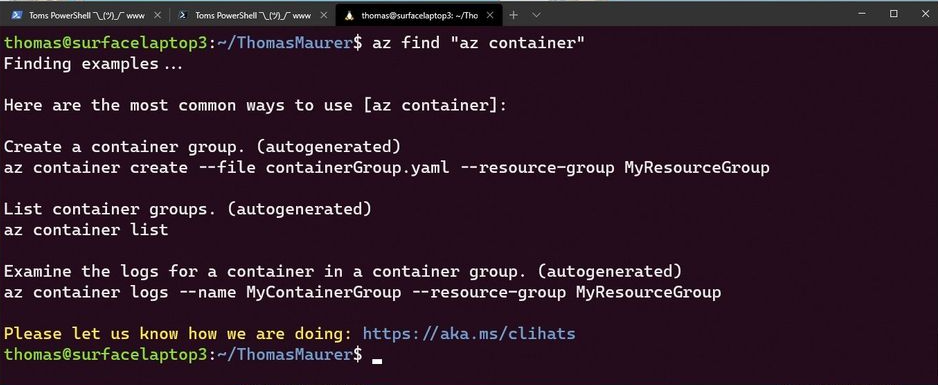
##### Evolving

Moving workloads to Azure alleviates some administrative burdens, but not all. Even though there is no need to worry about the data center, there is still a responsibility for service configuration and user access. Applications will need resource authorization.

Using the existing command-line tools and REST APIs, it is possible to build custom tools to automate and report resource configurations that do not meet organizational requirements.

#### Azure PowerShell and CLI

**Azure PowerShell** and the **Azure CLI** (for Bash shell users) are useful for automating tasks that cannot be performed in the Azure portal. Both tools follow an *imperative* approach, meaning that users must explicitly script the creation of resources in the correct order.



Shows an example of the Azure CLI.

There are subtle differences between how each of these tools operates and the actions that can be accomplished. Use the [Azure command-line tool guide](https://learn.microsoft.com/azure/developer/azure-cli/choose-the-right-azure-command-line-tool) to determine the right tool to meet the target goal.

#### Azure CLI

It is possible to run the Azure CLI and Azure PowerShell from the [Azure Cloud Shell](https://shell.azure.com), but it does have some limitations. It is also possible to run these tools locally.

To use the Azure CLI, [download the CLI tools from Microsoft.](https://learn.microsoft.com/cli/azure/install-azure-cli)

To use the Azure PowerShell cmdlets, install the Az module from the PowerShell Gallery, as described in the [installation document.](https://learn.microsoft.com/powershell/azure/install-az-ps?view=azps-6.6.0)

##### Azure Cloud Shell

The Azure Cloud Shell provides Bash and PowerShell environments for managing Azure resources imperatively. It also includes standard development tools, like Visual Studio Code, and files are persisted in an Azure Files share.

Launch the Cloud Shell in a browser at <https://shell.azure.com>.

#### PowerShell Module

The Azure portal and Windows PowerShell can be used for managing Azure Cosmos DB for Mongo DB API. To get started with Azure PowerShell, install the [Azure PowerShell cmdlets](https://learn.microsoft.com/en-us/powershell/module/az.cosmosdb/) for Cosmos DB with the following PowerShell command in an administrator-level PowerShell window:

Install-Module -Name Az.CosmosDB

#### Infrastructure as Code

[Infrastructure as Code (IaC)](https://learn.microsoft.com/devops/deliver/what-is-infrastructure-as-code) provides a way to describe or declare what infrastructure looks like using descriptive code. The infrastructure code is the desired state. The environment will be built when the code runs and completes. One of the main benefits of IaC is that it is human readable. Once the environment code is proven and tested, it can be versioned and saved into source code control. Developers can review the environment changes over time.

There are a few options of IaC tooling to choose from when provisioning and managing Azure resources. These include Azure-native tools from Microsoft, like ARM templates and Azure Bicep, as well as third-party tools popular within the industry like HashiCorp Terraform.

##### ARM templates

[ARM templates](https://learn.microsoft.com/azure/azure-resource-manager/templates/) can deploy Azure resources in a *declarative* manner. Azure Resource Manager can potentially create the resources in an ARM template in parallel. ARM templates can be used to create multiple identical environments, such as development, staging, and production environments.



The picture shows an example of an ARM template JSON export.

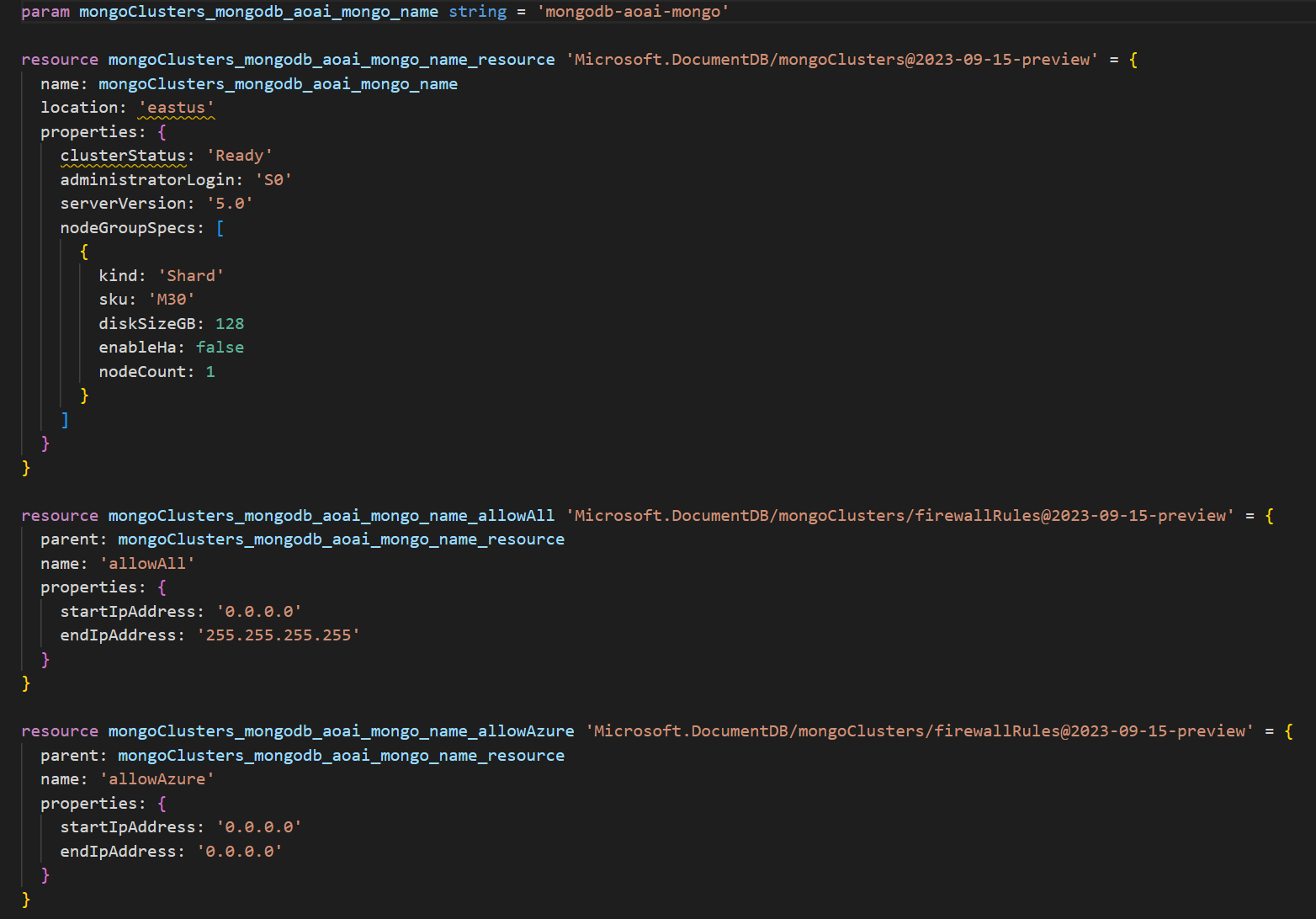
##### Bicep

Reading, updating, and managing the ARM template JSON code can be difficult for a reasonably sized environment. What if there was a tool that translates simple declarative statements into ARM templates? Better yet, what if there was a tool that took existing ARM templates and translated them into a simple configuration? [Bicep](https://learn.microsoft.com/azure/azure-resource-manager/bicep/overview) is a domain-specific language (DSL) that uses a declarative syntax to deploy Azure resources. Bicep files define the infrastructure to deploy to Azure and then use that file throughout the development lifecycle to repeatedly deploy infrastructure changes. Resources are deployed consistently.

By using the Azure CLI it is possible to decompile ARM templates to Bicep using the following:

az bicep decompile --file template.json

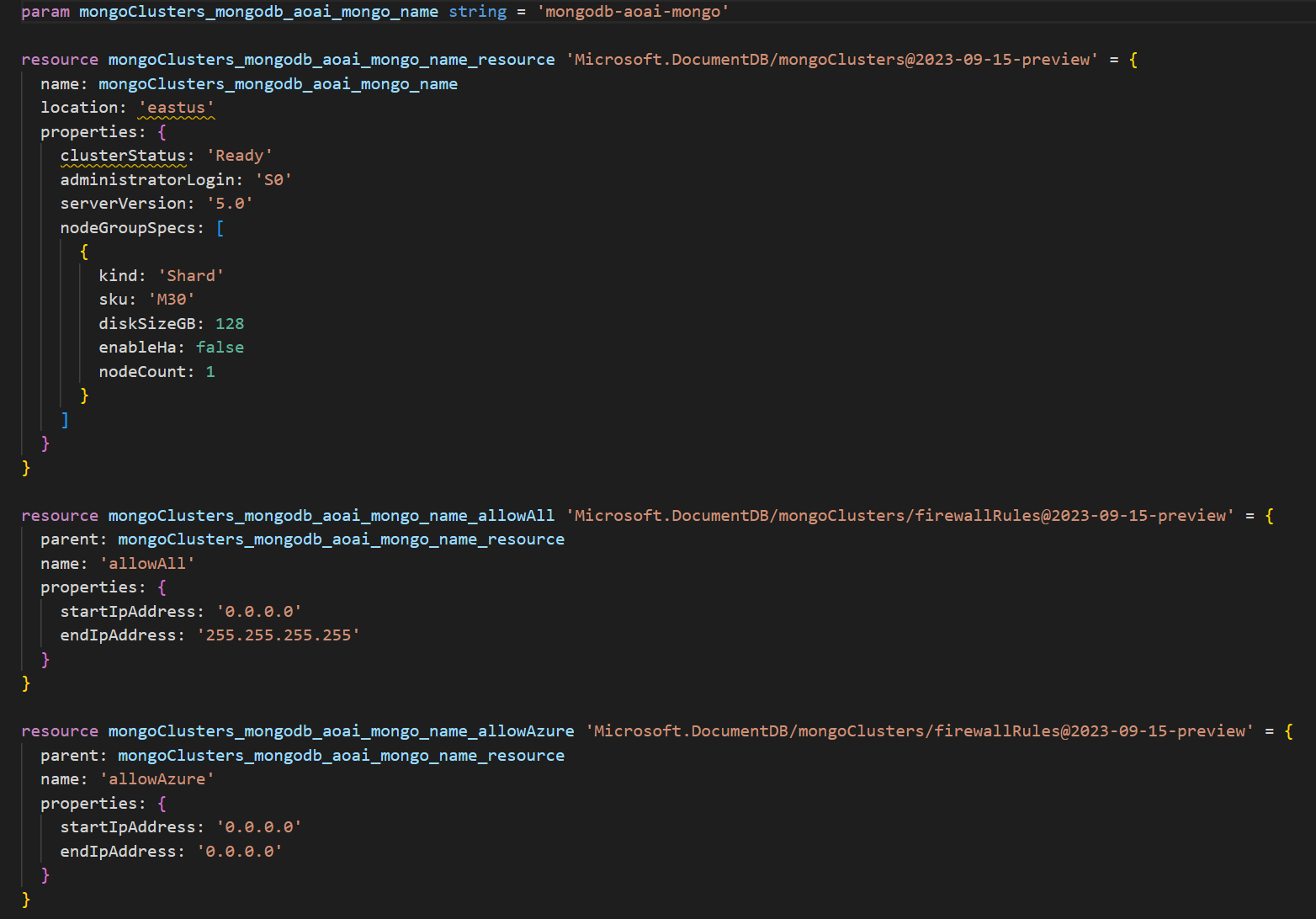
Additionally, the [Bicep playground](https://aka.ms/bicepdemo) tool can perform similar decompilation of ARM templates.



Sample Bicep code that deploys Azure Cosmos DB for MongoDB

##### Terraform

[Hashicorp Terraform](https://www.terraform.io/) is an open-source tool for provisioning and managing cloud infrastructure resources. [Terraform](https://learn.microsoft.com/azure/developer/terraform/overview) simplifies the deployment of Azure services, including Azure Kubernetes Service, Azure Cosmos DB, and Azure AI, through infrastructure-as-code to automate provisioning and management of Azure services. Terraform is also adept at deploying infrastructure across multiple cloud providers. It enables developers to use consistent tooling to manage each infrastructure definition.



Sample Terraform code that deploys Azure Cosmos DB for MongoDB

#### Other tips

Azure administrators should consult with cloud architects and financial and security personnel to develop an effective organizational hierarchy of resources.

Here are some best practices to follow for Azure deployments.

* **Utilize Management Groups** Create at least three levels of management groups.
* **Adopt a naming convention:** Azure resource names should include business details, such as the organization department, and operational details for IT personnel, like the workload. Defining an [Azure resource naming convention](https://learn.microsoft.com/azure/cloud-adoption-framework/ready/azure-best-practices/resource-naming) will help the organization standardize on a common naming convention that will help better identify resources once created.
* **Adopt other Azure governance tools:** Azure provides mechanisms such as [resource tags](https://learn.microsoft.com/azure/azure-resource-manager/management/tag-resources?tabs=json) and [resource locks](https://learn.microsoft.com/azure/azure-resource-manager/management/lock-resources?tabs=json) to facilitate compliance, cost management, and security.

# Overview of Azure Cosmos DB

[Azure Cosmos DB](https://learn.microsoft.com/en-us/azure/cosmos-db/introduction) is a globally distributed, multi-model database service that enables you to query and store data using NoSQL models using one of five APIs: SQL (document database), Cassandra (column-family), MongoDB (document database), Azure Table, and Gremlin (graph database). It provides turnkey global distribution, elastic scaling of throughput and storage worldwide, single-digit millisecond latencies at the 99th percentile, and guaranteed high availability with multi-homing capabilities. Azure Cosmos DB provides comprehensive service level agreements (SLAs) for throughput, latency, availability, and consistency guarantees, something not found in any other database service.

## Azure Cosmos DB for Mongo DB

[Azure Cosmos DB for MongoDB](https://learn.microsoft.com/en-us/azure/cosmos-db/mongodb/introduction) simplifies utilizing Azure Cosmos DB as a MongoDB database. You can leverage your current MongoDB expertise and still use your preferred MongoDB drivers, SDKs, and tools simply by directing your application to the connection string for your account.

### Azure Cosmos DB for Mongo DB API Architectures

The [RU architecture](https://learn.microsoft.com/en-us/azure/cosmos-db/mongodb/ru/introduction) for Azure Cosmos DB for MongoDB offers instantaneous scalability with zero warmup period, automatic and transparent sharding, and 99.999% availability. It supports active-active databases across multiple regions, cost-efficient, granular, unlimited scalability, real-time analytics, and serverless deployments where you pay only per operation.

Azure Cosmos DB for MongoDB [vCore architecture](https://learn.microsoft.com/en-us/azure/cosmos-db/mongodb/vcore/introduction) integrates AI-based applications with your data, with text indexing for easy querying. Simplify your development process with high-capacity vertical scaling and free 35-day backups with a point-in-time restore (PITR).

The [choice between vCore and Request Units (RU)](https://learn.microsoft.com/en-us/azure/cosmos-db/mongodb/choose-model) in Azure Cosmos DB for MongoDB API depends on the workload. A list of [compatibility and feature support between RU and vCore](https://learn.microsoft.com/en-us/azure/cosmos-db/mongodb/vcore/compatibility) is available.

vCore provides predictable performance and cost and is ideal for running high-performance, mission-critical workloads with low latency and high throughput. With vCore, the number of vCPUs and the memory the database needs is configurable and can be scaled up or down as needed.

Conversely, RU is a consumption-based model that charges based on the number of operations the database performs, including reads, writes, and queries. RU is ideal for scenarios where the workload has unpredictable traffic patterns or a need to optimize cost for bursty workloads.

A steady-state workload with predictable traffic patterns is best suited for vCore since it provides more predictable performance and cost. However, RU may be a better choice if the workload has unpredictable traffic patterns or requires bursty performance since it allows you to pay only for the resources used.

**NOTE**: AI-supporting workloads, such as vector search, must use vCore, as vector search is not supported with RU accounts.

# Overview of Azure OpenAI

Azure OpenAI is a collaboration between Microsoft Azure and OpenAI, a leading research organization in artificial intelligence. It is a cloud-based platform that enables developers and data scientists to build and deploy AI models quickly and easily. With Azure OpenAI, users can access a wide range of AI tools and technologies to create intelligent applications, including natural language processing, computer vision, and deep learning.

Azure OpenAI is designed to accelerate the development of AI applications, allowing users to focus on creating innovative solutions that deliver value to their organizations and customers.

Here are ways that Azure OpenAI can help developers:

* **Simplified integration** - Simple and easy-to-use APIs for tasks such as text generation, summarization, sentiment analysis, language translation, and more.
* **Pre-trained models** - AI models that are already fine-tuned on vast amounts of data making it easier for developers to leverage the power of AI without having to train their own models from scratch.
* **Customization** - Developers can also fine-tune the included pre-trained models with their own data with minimal coding, providing an opportunity to create more personalized and specialized AI applications.
* **Documentation and resources** - Azure OpenAI provides comprehensive documentation and resources to help developers get started quickly.
* **Scalability and reliability** - Hosted on Microsoft Azure, the OpenAI service provides robust scalability and reliability that developers can leverage to deploy their applications.
* **Responsible AI** - Azure OpenAI promotes responsible AI by adhering to ethical principles, providing explainability tools, governance features, diversity and inclusion support, and collaboration opportunities. These measures help ensure that AI models are unbiased, explainable, trustworthy, and used in a responsible and compliance manner.
* **Community support** - With an active developer community developers can seek help via forums and other community support channels.

## Comparison of Azure OpenAI and OpenAI

Azure OpenAI Service gives customers advanced language AI with OpenAI GPT-4, GPT-3, Codex, DALL-E, and Whisper models with the security and enterprise promise of Azure. Azure OpenAI co-develops the APIs with OpenAI, ensuring compatibility and a smooth transition from one to the other.

With Azure OpenAI, customers get the security capabilities of Microsoft Azure while running the same models as OpenAI. Azure OpenAI offers private networking, regional availability, and responsible AI content filtering.

## Azure OpenAI Data Privacy and Security

Azure OpenAI stores and processes data to provide the service and to monitor for uses that violate the applicable product terms. Azure OpenAI is fully controlled by Microsoft. Microsoft hosts the OpenAI models in Microsoft Azure for your usage of Azure OpenAI, and does not interact with any services operated by OpenAI.

Here are a few important things to know in regards to the security and privacy of your prompts (inputs) and completions (outputs), your embeddings, and your training data when using Azure OpenAI:

* are NOT available to other customers.
* are NOT available to OpenAI.
* are NOT used to improve OpenAI models.
* are NOT used to improve any Microsoft or 3rd party products or services.
* are NOT used for automatically improving Azure OpenAI models for your use in your resource (The models are stateless, unless you explicitly fine-tune models with your training data).
* Your fine-tuned Azure OpenAI models are available exclusively for your use.

## Azure AI Platform

Developers can use the power of AI, cloud-scale data, and cloud-native app development to create highly differentiated digital experiences and establish leadership among competitors. Build or modernize intelligent applications that take advantage of industry-leading AI technology and leverage real-time data and analytics to deliver adaptive, responsive, and personalized experiences.

The Azure platform of managed AI, containers, and database services, along with offerings developed by or in partnership with key software vendors, enables developers to build, deploy, and scale applications with speed, flexibility, and enterprise-grade security. This platform has been used by market leaders like The NBA, H&R Block, Real Madrid Football Club, Bosch, and Nuance to develop their own intelligent apps.

Developers can use Azure AI Services, along with other Azure services, to build and modernize intelligent apps on Azure. Examples of this could be:

* Build new with Azure Kubernetes Service or Azure Container Apps, Azure Cosmos DB, and Azure AI Services
* Modernize with Azure Kubernetes Service, Azure SQL or Azure Database for PostgresSQL, and Azure AI Services

### Azure AI Services

While this guide focuses on building intelligent apps using Azure OpenAI combined with vCore-based Azure Cosmos DB for MongoDB, the Azure AI Platform consists of many additional AI services. Each AI service is built to fit a specific AI and/or Machine Learning (ML) need.

Here’s a list of the AI services within the [Azure AI platform](https://learn.microsoft.com/azure/ai-services/what-are-ai-services):

| Service | Description |
| --- | --- |
| Azure AI Search | Bring AI-powered cloud search to your mobile and web apps |
| Azure OpenAI | Perform a wide variety of natural language tasks |
| Bot Service | Create bots and connect them across channels |
| Content Safety | An AI service that detects unwanted contents |
| Custom Vision | Customize image recognition to fit your business |
| Document Intelligence | Turn documents into usable data at a fraction of the time and cost |
| Face | Detect and identify people and emotions in images |
| Immersive Reader | Help users read and comprehend text |
| Language | Build apps with industry-leading natural language understanding capabilities |
| Machine Learning | ML professionals, data scientists, and engineers can use Azure Machine Learning in their day-to-day workflows to train and deploy models, such as those built from an open-source platform, such as PyTorch, TensorFlow, or scikit-learn |
| Speech | Speech to text, text to speech, translation and speaker recognition |
| Translator | Translate more than 100 languages and dialects |
| Video Indexer | Extract actionable insights from your videos |
| Vision | Analyze content in images and videos |

**Note:** Follow this link for additional tips to help in determining the which Azure AI service is most appropriate for a specific project requirement: <https://azure.microsoft.com/products/category/ai>

The tools that you will use to customize and configure models are different from those that you’ll use to call the Azure AI services. Out of the box, most Azure AI services allow you to send data and receive insights without any customization.

For example:

* You can send an image to the Azure AI Vision service to detect words and phrases or count the number of people in the frame
* You can send an audio file to the Speech service and get transcriptions and translate the speech to text at the same time

Azure offers a wide range of tools that are designed for different types of users, many of which can be used with Azure AI services. Designer-driven tools are the easiest to use, and are quick to set up and automate, but might have limitations when it comes to customization. The REST APIs and client libraries provide users with more control and flexibility, but require more effort, time, and expertise to build a solution. When using REST APIs and client libraries, there is an expectation that the developer is comfortable working with modern programming languages like C#, Java, Python, JavaScript, or another popular programming language.

### Azure Machine Learning

[Azure Machine Learning](https://learn.microsoft.com/azure/machine-learning/overview-what-is-azure-machine-learning?view=azureml-api-2) is a cloud service for accelerating and managing the machine learning (ML) project lifecycle. ML professionals, data scientists, and engineers can use it in their day-to-day workflows to train and deploy models and manage machine learning operations (MLOps).

Azure Machine Learning can be used to create a model or use a model built from an open-source platform, such as PyTorch, TensorFlow, or scikit-learn. Additionally, MLOps tools help you monitor, retrain, and redeploy models.

ML projects often require a team with a varied skill set to build and maintain. Azure Machine Learning has tools that help enable you to:

* Collaborate with your team via shared notebooks, compute resources, serverless compute, data, and environments
* Develop models for fairness and explainability, tracking and auditability to fulfill lineage and audit compliance requirements
* Deploy ML models quickly and easily at scale, and manage and govern them efficiently with MLOps
* Run machine learning workloads anywhere with built-in governance, security, and compliance

Enterprises working in the Microsoft Azure cloud can use familiar security and role-based access control for infrastructure. You can set up a project to deny access to protected data and select operations.

#### Azure Machine Learning vs Azure Open AI

Many of the Azure AI services are suited to a very specific AI / ML need. The Azure Machine Learning and Azure OpenAI services offer more flexible usage based on the solution requirements.

Here are a couple differentiators to help determine which of these to services to use when comparing the two:

* Azure Machine Learning service is appropriate for solutions where a custom model needs to be trained specifically on your own data.
* Azure OpenAI service is appropriate for solutions that require pre-trained models that provide natural language processing or vision services, such as the GPT-4 or DALL-E models from OpenAI.

If the solution requires other more task specific AI features, then one of the other Azure AI services should be considered.

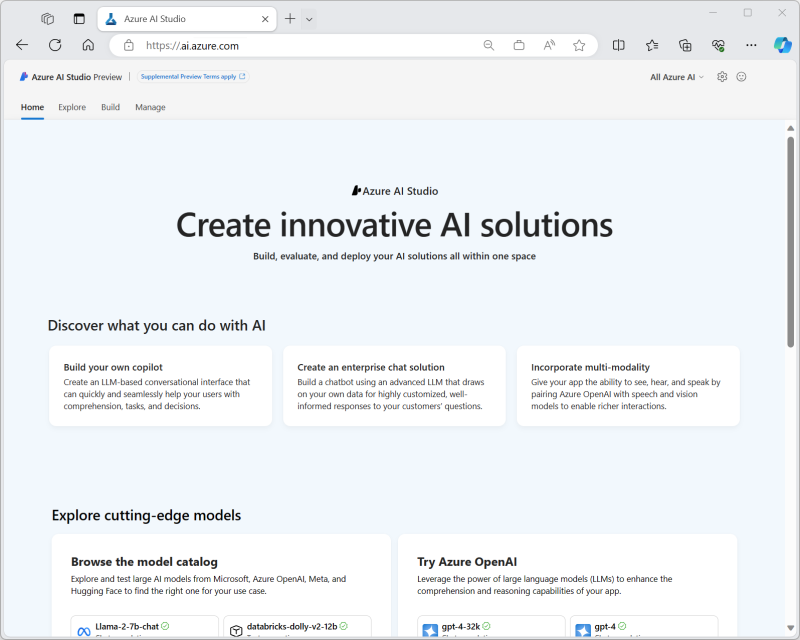
### Azure AI Studio

Azure AI Studio is a web portal that brings together multiple Azure AI-related services into a single, unified development environment.

Specifically, Azure AI Studio combines:

* The model catalog and prompt flow development capabilities of Azure Machine Learning service.
* The generative AI model deployment, testing, and custom data integration capabilities of Azure OpenAI service.
* Integration with Azure AI Services for speech, vision, language, document intelligence, and content safety.

Azure AI Studio enables teams to collaborate efficiently and effectively on AI projects, such as developing custom copilot applications that use large language models (LLMs).



Azure AI Studio screenshot

Tasks you can accomplish with Azure AI Studio include:

* Deploying models from the model catalog to real-time inferencing endpoints for client applications to consume.
* Deploying and testing generative AI models in an Azure OpenAI service.
* Integrating data from custom data sources to support a retrieval augmented generation (RAG) approach to prompt engineering for generative AI models.
* Using prompt flow to define workflows that integrate models, prompts, and custom processing.
* Integrating content safety filters into a generative AI solution to mitigate potential harms.
* Extending a generative AI solution with multiple AI capabilities using Azure AI services.

# Overview of AI Concepts

## Large Language Models (LLM)

A Large Language Models (LLM) is a type of AI that can process and produce natural language text. LLMs are “general purpose” AI models trained using massive amounts of data gathered from various sources; like books, articles, webpages, and images to discover patterns and rules of language.

LLMs are complex and built using a neural network architecture. They are trained using large amounts of information, and calculate millions of parameters. From a developer perspective, the APIs expose by Azure OpenAI Service enable the LLMs to be easily integrated into enterprise solutions without requiring knowledge of how to build to train the models.

Understanding the capabilities of what an LLM can do is important when deciding to use it for a solution:

* **Understand language** - An LLM is a predictive engine that pulls patterns together based on pre-existing text to produce more text. It doesn’t understand language or math.
* **Understand facts** - An LLM doesn’t have separate modes for information retrieval and creative writing; it simply predicts the next most probable token.
* **Understand manners, emotion, or ethics** - An LLM can’t exhibit anthropomorphism or understand ethics. The output of a foundational model is a combination of training data and prompts.

### Foundational Models

Foundational Models are specific instances or versions of an LLM. Examples of these would be GPT-3, GPT-4, or Codex. Foundational models are trained and fine-tuned on a large corpus of text, or code in the case of a Codex model instance.

A foundational model takes in training data in all different formats and uses a transformer architecture to build a general model. Adaptions and specializations can be created to achieve certain tasks via prompts or fine-tuning.

### Difference between LLM and traditional Natural Language Processing (NLP)

LLMs and Natural Language Processing (NLP) differs in their approach to understanding and processing language.

Here are a few things that separate NLPs from LLMs:

| Traditional NLP | Large Language Models |
| --- | --- |
| One model per capability is needed. | A single model is used for many natural language use cases. |
| Provides a set of labeled data to train the ML model. | Uses many terabytes of unlabeled data in the foundation model. |
| Describes in natural language what you want the model to do. | Highly optimized for specific use cases. |

## Prompting and Prompt Engineering

### What is a prompt?

A prompt is an input or instruction provided to an Artificial Intelligence (AI) model to direct its behavior and produce the desired results. The quality and specificity of the prompt are crucial in obtaining precise and relevant outputs. A well-designed prompt can ensure that the AI model generates the desired information or completes the intended task effectively. Some typical prompts include summarization, question answering, text classification, and code generation.

While there’s techniques and patterns used when building an Azure OpenAI solution and writing prompts, the following is a couple simple prompt examples:

* List the most popular products in the last quarter.
* How many customers are located in the state of California?

#### Guidelines for creating robust prompts

While it can be quick to write basic prompts, it can also be difficult to write more complex prompts to ge the AI to generate the responses necessary. When writing prompts, there are three basic guidelines to follow for creating useful prompts:

* **Show and tell** - Make it clear what you want either through instructions, examples, or a combination of the two. If you want the model to rank a list of items in alphabetical order or to classify a paragraph by sentiment, include these details in your prompt to show the model.
* **Provide quality data** - If you’re trying to build a classifier or get the model to follow a pattern, make sure there are enough examples. Be sure to proofread your examples. The model is smart enough to resolve basic spelling mistakes and give you a meaningful response. Conversely, the model might assume the mistakes are intentional, which can affect the response.
* **Check your settings** - Probability settings, such as Temperature and Top P, control how deterministic the model is in generating a response. If you’re asking for a response where there’s only one right answer, you should specify lower values for these settings. If you’re looking for a response that’s not obvious, you might want to use higher values. The most common mistake users make with these settings is assuming they control “cleverness” or “creativity” in the model response.

### What is prompt engineering

[Prompt engineering](https://learn.microsoft.com/en-us/azure/ai-services/openai/concepts/prompt-engineering) is the iterative process of designing, evaluating, and optimizing prompts to produce consistently accurate responses from language models for a particular problem domain. It involves designing and refining the prompts given to an AI model to achieve the desired outputs. Prompt engineers experiment with various prompts, test their effectiveness, and refine them to improve performance. Performance is measured using predefined metrics such as accuracy, relevance, and user satisfaction to assess the impact of prompt engineering.

### General anatomy of a prompt

context, input, output indicator There are a several components that are used in the anatomy of constructing AI prompts:

* Input
  + Instructions
  + Primary content
  + Supporting content
  + System message
* Output

#### Input

The Input in prompt engineering is the text or query that is provided to the LLM model. The model will use this prompt as the basis for the completion, or response, that is generated.

##### Instructions

Instructions are likely the most commonly used prompt component. Instructions are simply that: instructions to the model on what to do. While simple in concept, they can be complex in practice.

The following are examples of both simple and more complex instructions:

| Instruction Type | Prompt | Completion |
| --- | --- | --- |
| Simple | Write an introduction for a weekly newsletter. | Welcome to our weekly newsletter. This week we have a special focus on productivity. We’ll be sharing tips and tricks on how to be productive in all areas of your life. We hope you enjoy and please share with your friends! |
| Complex | Write an introduction for a weekly newsletter to my company named Contoso. It should mention the great all hands meeting we had last week and thank the team for all their hard work over the tough past few months. | Welcome to Contoso’s weekly newsletter! We had a great all hands meeting last week and I want to thank the team for all their hard work over the tough past few months. We’re looking forward to a bright future and I’m confident that with our team’s dedication and commitment, we will achieve great things. Thank you for being a part of our team and I look forward to working with you all in the coming weeks and months. Sincerely, [Your name]. |
| Complex | Write an introduction for a weekly newsletter, including the following: - My company is named contoso - An exuberant greeting - Thanking the team for their hard work over a couple tough months - Positive outlook for the coming quarter - Signed by the SLT | Hello Contoso team! I wanted to take a moment to thank you for all your hard work over the past few months. It’s been a tough couple of months, but you’ve all pulled together and done an amazing job. I’m really proud of you all. Looking ahead to the next quarter, I’m feeling really positive. I think we’re going to achieve some great things. Thanks again, The Senior Leadership Team. |

##### Primary Content

Primary content refers to some sort of text that is being processed or transformed by the model. Primary content is typically used in conjunction with instructions. A simple example would be language translation.

| Prompt | Completion |
| --- | --- |
| Can you please tell me how to get to the museum? Translate to French: | Pouvez-vous s'il vous plaît me dire comment aller au musée? |

Primary content can also be much longer. For example, the primary content could pass the introduction section of text content that could be hundreds of words long. Additionally, the primary content could be int he form of structured data as well, such as in JSON or TSV format.

##### Supporting Content

Supporting content is information that the model can utilize to influence the output in some way. It differs from primary content in that it is not the main target of task, but it is typically used along with primary content. Common examples include contextual information such as the current date, the name of the user, user preferences, and so on.

The following are example prompts and possible completions to show what a prompt with and without supporting content might look like:

| Supporting Content | Prompt | Completion |
| --- | --- | --- |
| Without | Workshop list: |  |
| Title Description Planning Workshop Kickoff Please join us to kick off our workshop! Hear from our leaders about cross group priorities and learnings for the year ahead. The invitation to this event has already been sent to everyone in […] Summarize the workshops above, grouped by topic: | Workshop list: Planning Workshop Kickoff - Planning Workshop Wrap-up - Core MeetUp - Customer Driven Perspectives - Impact and Key Focus Areas - Best practices - Accelerating app adoption […] |  |
| With | Workshop list: Title Description Planning Workshop Kickoff Please join us to kick off our workshop! Hear from our leaders about cross group priorities and learnings for the year ahead. The invitation to this event has already been sent to everyone in […] My Important Topics: prompt engineering, search, GPT models Summarize the workshops above, grouped by my important topics: | 1. Prompt Engineering: - New prompt engineering tips - The art of prompt engineering 2. Search: - Intro to vector search with embedding 3. GPT Models: - Intro to GPT-4 - GPT-35-Turbo in-depth. |

The above prompt examples are truncated with […] for brevity. Writing similar prompts would contain more text.

##### System message

System message, also called a system prompt, is used by developers of a system to restrict the LLM to a specific set of constraints. This is useful when building enterprise solutions that integrate Azure OpenAI so the AI completion will be restricted to the focus of the enterprise data it’s integrated with.

The following is an example system message that could be used to constrain the LLM in an enterprise solution:

You are a helpful, fun and friendly sales assistant for Cosmic Works, a bicycle and bicycle accessories store.  
  
Your name is Cosmo.  
  
You are designed to answer questions about the products that Cosmic Works sells, the customers that buy them, and the sales orders that are placed by customers.  
  
If you don't know the answer to a question, respond with "I don't know."  
  
Only answer questions related to Cosmic Works products, customers, and sales orders.  
  
If a question is not related to Cosmic Works products, customers, or sales orders, respond with "I only answer questions about Cosmic Works

#### Output

The Output is the completion, or response, from the LLM returned as a result to the input prompt given. When an input prompt is given, the language model will process the information and generate an output in the form of text. The text response is the output.

## Standard Patterns

### Retrieval Augmentation Generation (RAG)

[Retrieval Augmentation Generation (RAG)](https://learn.microsoft.com/en-us/azure/search/retrieval-augmented-generation-overview) is an architecture that augments the capabilities of a Large Language Model (LLM) like ChatGPT by adding an information retrieval system that provides grounding data. Adding an information retrieval system gives you control over grounding data used by an LLM when it formulates a response. For an enterprise solution, RAG architecture means that you can constrain generative AI to your enterprise content sourced from vectorized documents, images, audio, and video.

GPT language models can be fine-tuned to achieve several common tasks such as sentiment analysis and named entity recognition. These tasks generally don’t require additional background knowledge.

The RAG pattern facilitates bringing private proprietary knowledge to the model so that it can perform Question Answering over this content. Remember that Large Language Models are indexed only on public information. For an enterprise solution, RAG architecture means you can constrain the generative AI to your enterprise content sourced from vectorized documents, images, audio, and video.

Because the RAG technique accesses external knowledge sources to complete tasks, it enables more factual consistency, improves the reliability of the generated responses, and helps to mitigate the problem of “*hallucination*”.

In some cases, the RAG process involves a technique called vectorization on the proprietary data. The user prompt is compared to the vector store and only the most relevant/matching pieces of information are returned and stuffed into prompt for the LLM to reason over and provide an answer. The next set of demos will go into this further.

### Chain of Thought (CoT)

Instead of splitting a task into smaller steps, with [Chain of Thought (CoT)](https://learn.microsoft.com/en-us/azure/ai-services/openai/concepts/advanced-prompt-engineering?pivots=programming-language-chat-completions#chain-of-thought-prompting) the model response is instructed to proceed step-by-step and present all the steps involved. Doing so reduces the possibility of inaccuracy of outcomes and makes assessing the model response easier.

| System message | Prompt | Response |
| --- | --- | --- |
| You are an AI assistant that helps people find information. | Who was the most decorated (maximum medals) individual athlete in the Olympic games that were held at Sydney? Take a step-by-step approach in your response, cite sources and give reasoning before sharing final answer in the below format: ANSWER is: <name> | Step 1: Determine the year of the Sydney Olympic games. The Sydney Olympic games were held in the year 2000. Step 2: Find a list of individual athletes who won multiple medals in the 2000 Sydney Olympics. According to the official website of the Olympic Games, there were several individual athletes who won multiple medals in the 2000 Sydney Olympics. Some of the notable names include: - Ian Thorpe (Australia) - 3 gold, 2 silver, 0 bronze - Pieter van den Hoogenband (Netherlands) - 2 gold, 0 silver, 2 bronze - Lenny Krayzelburg (USA) - 3 gold, 0 silver, 0 bronze - Inge de Bruijn (Netherlands) - 3 gold, 1 silver, 0 bronze Step 3: Determine the athlete with the most medals. After reviewing the list of individual athletes who won multiple medals in the 2000 Sydney Olympics, we can see that the athlete with the most medals is Ian Thorpe from Australia, who won a total of 5 medals (3 gold and 2 silver). Therefore, the ANSWER is: Ian Thorpe. |

### ReAct

Reasoning and Acting (ReAct) leverages LLMs to generate both reasoning traces and task-specific actions. ReAct combines these to allow for greater synergy between the two. Reasoning traces help the model induce, track, and update action plans as well as handle exceptions, while actions allow it to interface with external data sources to gather additional information. When using LLMs to answer questions and verify facts, ReAct is used to overcome issues of hallucination and error propagation by interacting with the additional usage of external data sources.

### Others

#### Zero-shot prompting

Zero-shot prompting is what we would consider the “default”. This is when we provide no examples of inputs/expected outputs to the model to work with. We’re leaving it up to the model to decipher what is needed and how to output it from the instructions.

Example:

| Prompt | Completion |
| --- | --- |
| Headline: Coach confident injury won’t derail Warriors Topic: | The coach is confident that the injury won’t derail the Warriors’ season. The team is still focused on their goals and that they will continue to work hard to achieve them. |

#### Few-shot prompting

Few-shot prompting is a technique that enables in-context learning for the LLM by providing examples or further information within the prompt to steer the model to generate a better response. Providing additional information in the prompt helps particularly in areas where the data used to train the model isn’t enough to generate the desired output from the model.

Example:

| Prompt | Completion |
| --- | --- |
| Headline: Twins’ Correa to use opt-out, test free agency Topic: Baseball Headline: Qatar World Cup to have zones for sobering up Topic: Soccer Headline: Yates: Fantasy football intel for Week 6 Topic: Football Headline: Coach confident injury won’t derail Warriors Topic: | Basketball |

# Explore the Azure OpenAI models and endpoints (console app)

## Azure OpenAI Models

Azure OpenAI is powered by a diverse set of models with different capabilities.

| Model | Description |
| --- | --- |
| GPT-4 | A set of models that improve on GPT-3.5 and can understand and generate natural language and code. |
| GPT-3.5 | A set of models that improve on GPT-3 and can understand and generate natural language and code. |
| Embeddings | A set of models that can convert text into numerical vector form to facilitate text similarity. |
| DALL-E | A series of models that can generate original images from natural language. |
| Whisper | A series of models that can transcribe and translate speech to text. |

### GPT-4 and GPT-3.5 Models

GPT-4 can solve difficult problems with greater accuracy than any of OpenAI’s previous models. Like GPT-3.5 Turbo, GPT-4 is optimized for chat and works well for traditional completions tasks.

The GPT-35-Turbo and GPT-4 models are language models that are optimized for conversational interfaces. The models behave differently than the older GPT-3 models. Previous models were text-in and text-out, meaning they accepted a prompt string and returned a completion to append to the prompt. However, the GPT-35-Turbo and GPT-4 models are conversation-in and message-out. The models expect input formatted in a specific chat-like transcript format, and return a completion that represents a model-written message in the chat. While this format was designed specifically for multi-turn conversations, you’ll find it can also work well for non-chat scenarios too.

### Embeddings

Embeddings, such as the text-embedding-ada-002 model, measure the relatedness of text strings.

Embeddings are commonly used for the following:

* **Search** - results are ranked by relevance to a query string
* **Clustering** - text strings are grouped by similarity
* **Recommendations** - items with related text strings are recommended
* **Anomaly detection** - outliers with little relatedness are identified
* **Diversity measurement** - similarity distributions are analyzed
* **Classification** - text strings are classified by their most similar label

### DALL-E

DALL-E is a model that can generate an original images from a natural language text description given as input.

### Whisper

Whisper is a speech recognition model, designed for general-purpose applications. Trained on an extensive dataset encompassing diverse audio inputs, and operates as a multi-tasking model capable of executing tasks like multilingual speech recognition, speech translation, and language identification.

## Selecting an LLM

Before a Large Language Model (LLM) can be implemented into a solution, an LLM model must be chosen. For this the business use case and other aspects to the overall goal of the AI solution will need to be defined.

Once the business goals of the solution are known, there are a few key considerations to think about:

* **Business Use Case** - What are the specific tasks the business needs the AI solution to perform? Each LLM is designed for different goals, such as text generation, language translation, image generation, answering questions, code generation, etc.
* **Pricing** - For cases where there may be multiple LLMs to choose from, the [pricing](https://azure.microsoft.com/pricing/details/cognitive-services/openai-service/) of the LLM could be a factor to consider. For example, when choosing between GPT-3.5 or GPT-4, it may be worth to consider that the overall cost of GPT-4 may be higher than GPT-3.5 for the solution since GPT-4 requires more compute power behind the scenes than GPT-3.5
* **Accuracy** - For cases where there may be multiple LLMs to choose from, the comparison of accuracy between them may be a factor to consider. For example, GPT-4 offers improvements over GPT-3.5 and depending on the use case, GPT-4 may provide increased accuracy.
* **Quotas and limits** - The Azure OpenAI service does have [quotas and limits](https://learn.microsoft.com/azure/ai-services/openai/quotas-limits) on using the service. This may affect the performance and pricing of the AI solution. Additionally, some of quotas and limits may vary depending on the Azure Region that is used to host the Azure OpenAI service. The potential impact of these on the pricing and performance of the solution will want to be considered in the design phase of the solution.

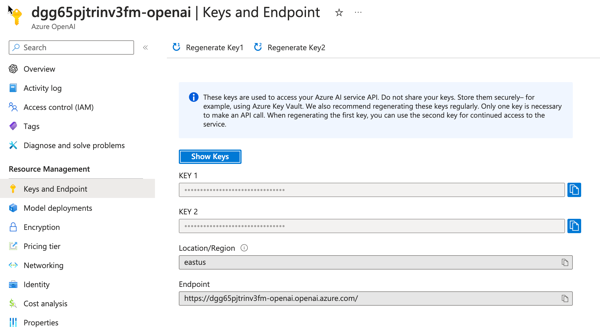
## Do I use an out-of-the-box model or a fine-tuned model?

A base model is a model that hasn’t been customized or fine-tuned for a specific use case. Fine-tuned models are customized versions of base models where a model’s weights are trained on a unique set of prompts. Fine-tuned models let you achieve better results on a wider number of tasks without needing to provide detailed examples for in-context learning as part of your completion prompt.

The [fine-tuning guide](https://learn.microsoft.com/azure/ai-services/openai/how-to/fine-tuning) can be referenced for more information.

## Explore and use Azure OpenAI models from code

The key and endpoint necessary to make API calls to Azure OpenAI can be located on **Azure OpenAI** blade in the Azure Portal on the **Keys and Endpoint** pane.



Azure OpenAI Keys and Endpoint pane in the Azure Portal

Please visit the lab repository to complete this lab.

# Provision Azure resources (Azure Cosmos DB workspace, Azure OpenAI, etc.)

**TBD once all other modules are complete.**

Bicep deployment is available in the [deploy folder of the lab repository](https://github.com/solliancenet/cosmos-db-openai-python-dev-guide-labs/tree/main/deploy). Currently deploying the following:

* Resource Group (this is done manually in the Azure Portal)
* Azure vCore-based Azure Cosmos DB for MongoDB account
* Azure OpenAI resource
  + Chat GPT-3.5 completions model
  + text-embedding-ada-002 model embeddings model
* Azure Container Registry to host Docker images
* Azure Container Apps Environment to host Docker containers
* Azure Container App to run Docker containers (initially with hello-world, but updated during the Backend API lab to run the backend Python API)
* Azure App Service to host Front-End SPA written in React

# Create your first Cosmos DB project

This section will cover how to create your first Cosmos DB project. We’ll use a notebook to demonstrate the basic CRUD operations. We’ll also cover using the Azure Cosmos DB Emulator to test your code locally.

## Emulator support

Azure Cosmos DB has an emulator that you can use to develop your code locally. The emulator supports the API for NoSQL and the API for MongoDB. The use of the emulator does not require an Azure subscription, nor does it incur any costs, so it is ideal for local development and testing. The Azure Cosmos DB emulator can also be utilized with unit tests in a [GitHub Actions CI workflow](https://learn.microsoft.com/en-us/azure/cosmos-db/how-to-develop-emulator?tabs=windows%2Cpython&pivots=api-mongodb#use-the-emulator-in-a-github-actions-ci-workflow).

There is not 100% feature parity between the emulator and the cloud service. Visit the [Azure Cosmos DB emulator](https://learn.microsoft.com/en-us/azure/cosmos-db/emulator) documentation for more details.

For Windows machines, the emulator can be installed via an installer. There is a Windows container using Docker available. However, it does not currently support the API for Mongo DB. A Docker image is also available for Linux that does support the API for Mongo DB.

Learn more about the pre-requisites and installation of the emulator [here](https://learn.microsoft.com/en-us/azure/cosmos-db/how-to-develop-emulator?tabs=windows%2Cpython&pivots=api-mongodb).

**NOTE**: When using the Azure CosmosDB emulator using the API for MongoDB it must be started with the [MongoDB endpoint options enabled](https://learn.microsoft.com/en-us/azure/cosmos-db/how-to-develop-emulator?tabs=windows%2Cpython&pivots=api-mongodb#start-the-emulator) at the command-line.

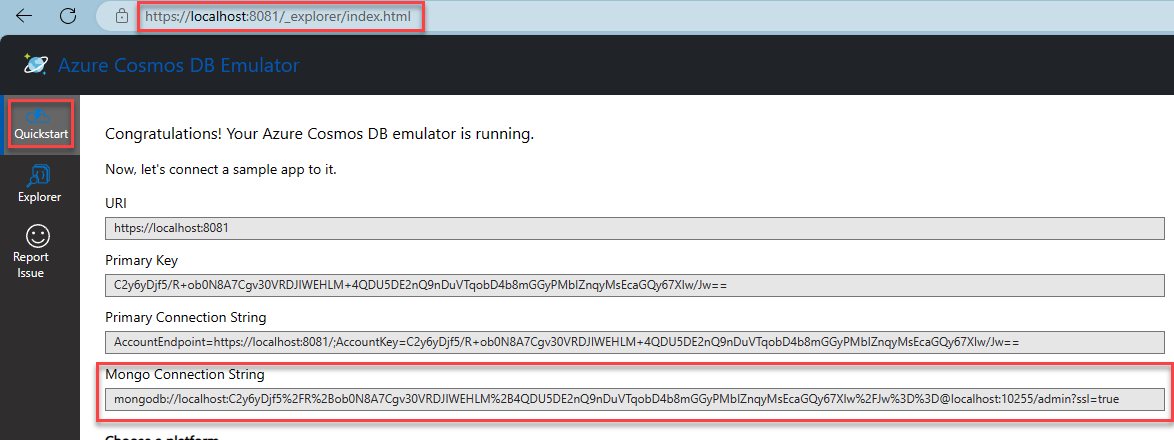
**The Azure Cosmos DB emulator does not support vector search. To complete the vector search and AI-related labs, you must use an vCore-based Azure Cosmos DB for MongoDB account in Azure.**

## Authentication

Authentication to Azure Cosmos DB API for Mongo DB uses a connection string. The connection string is a URL that contains the authentication information for your Azure Cosmos DB account or local emulator. The username and password used when provisioning the Azure Cosmos DB API for MongoDB service are used in the connection string when authenticating to Azure.

### Retrieving the connection string from the Cosmos DB Emulator

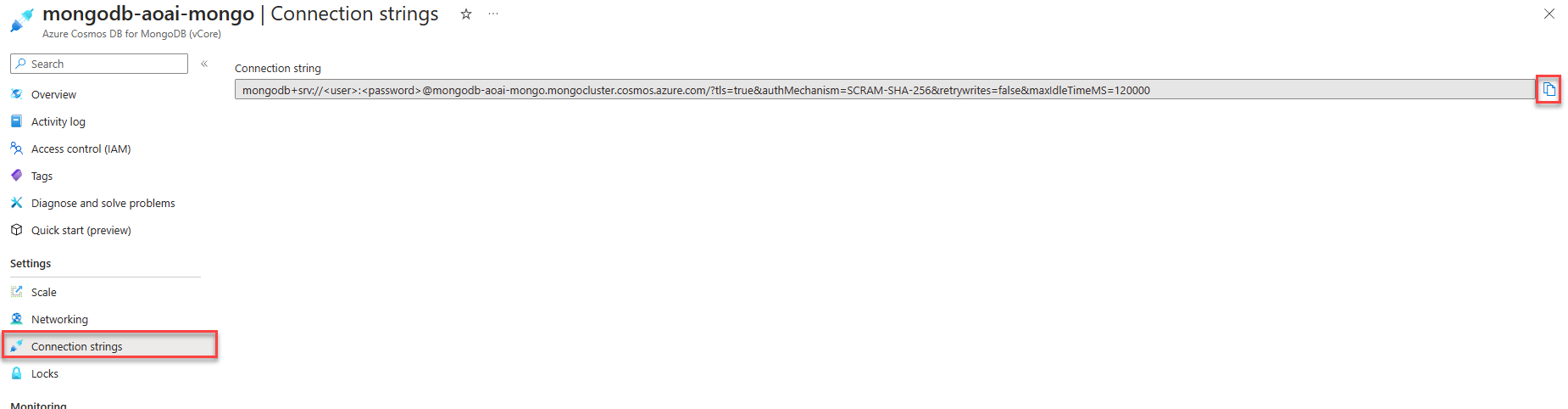
The splash screen or **Quickstart** section of the Cosmos DB Emulator will display the connection string. Access this screen through the following URL: https://localhost:8081/\_explorer/index.html.



The Azure Cosmos DB emulator screen displays with the local host url, the Quickstart tab, and the Mongo connection string highlighted.

### Retrieving the connection string from the Azure portal

Retrieve the connection string from the Azure portal by navigating to your Azure Cosmos DB account and selecting the **Connection String** menu item on the left-hand side of the screen. The connection string contains tokens for the username and password that must be replaced with the username and password used when provisioning the Azure Cosmos DB API for MongoDB service.



The Azure CosmosDb API for MongoDB Connection strings screen displays with the copy button next to the connection string highlighted.

## Lab 1 - Create your first Cosmos DB for the MongoDB application

Using a notebook, we’ll create a Cosmos DB for the MongoDB application in this lab. You can use the Azure Cosmos DB Emulator or an Azure Cosmos DB account in Azure. The following concepts will be covered in this lab:

1. Create a database
2. Create a collection
3. Create a document
4. Read a document
5. Update a document
6. Delete a document
7. Query documents

It is highly recommended that you use a [virtual environment](https://python.land/virtual-environments/virtualenv) for this lab

Please visit the lab repository to complete this lab.

# Load data into Azure Cosmos DB API for MongoDB

Lab 1 demonstrated how to add data to a collection individually. This lab will demonstrate how to load data using bulk operations into multiple collections. This data will be used in subsequent labs to explain further the capabilities of Azure Cosmos DB API for MongoDB about AI.

When loading data, bulk operations are preferred over adding each document individually. Bulk operations involve performing multiple database operations as a batch rather than executing them simultaneously. This approach is more efficient and provides several benefits:

1. Performance: By issuing load operations in bulk, the lab can significantly reduce the overhead of network round-trips and database operations. This results in faster data loading and improved overall performance.
2. Scalability: Bulk operations allow the lab to handle large volumes of data efficiently. They can quickly process and load a substantial amount of customer, product, and sales data, enabling them to scale their operations as needed.
3. Atomicity: Bulk operations ensure that all database changes within a batch are treated as a single transaction. The entire batch can be rolled back if any document fails to load, maintaining data integrity and consistency.
4. Simplified code logic: By using bulk operations, the lab can simplify its code logic and reduce the number of database queries. This results in cleaner, more manageable code and reduces the likelihood of errors or inconsistencies.

## Lab 2 - Load data into Azure Cosmos DB API for MongoDB collections

This lab will load the Cosmic Works Customer, Product, and Sales data into Azure Cosmos DB API for MongoDB collections using bulk operations.

Please complete Lab 2 in the lab repository.

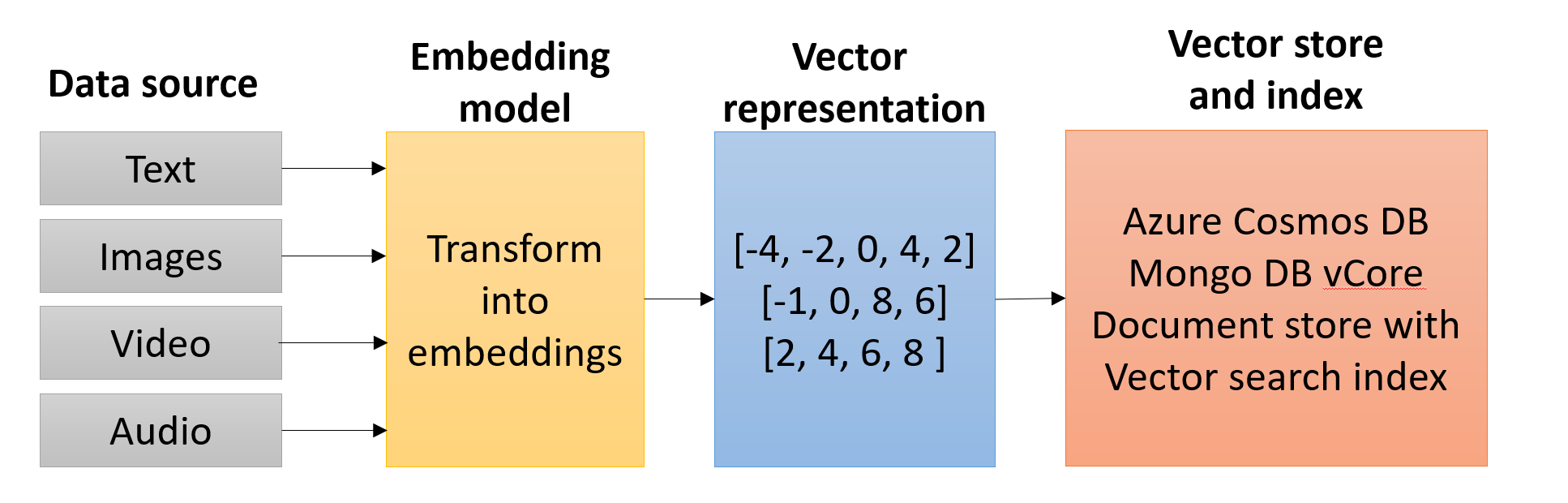
# Use vector search on embeddings in vCore-based Azure Cosmos DB for MongoDB

**NOTE**: vCore-based Azure Cosmos DB for MongoDB supports vector search on embeddings. This functionality is not supported on RUs-based accounts.

## Embeddings and vector search

Embedding is a way of serializing the semantic meaning of data into a vector representation. Because the generated vector embedding represents the semantic meaning, it means that when it is searched, it can find similar data based on the semantic meaning of the data rather than exact text. Data can come from many sources, including text, images, audio, and video. Because the data is represented as a vector, vector search can, therefore, find similar data across all different types of data.

Embeddings are created by sending data to an embedding model, where it is transformed into a vector, which then can be stored as a vector field within its source document in vCore-based Azure Cosmos DB for MongoDB. vCore-based Azure Cosmos DB for MongoDB supports the creation of vector search indexes on top of these vector fields. A vector search index is a collection of vectors in [latent space](https://idl.cs.washington.edu/papers/latent-space-cartography/) that enables a semantic similarity search across all data (vectors) contained within.



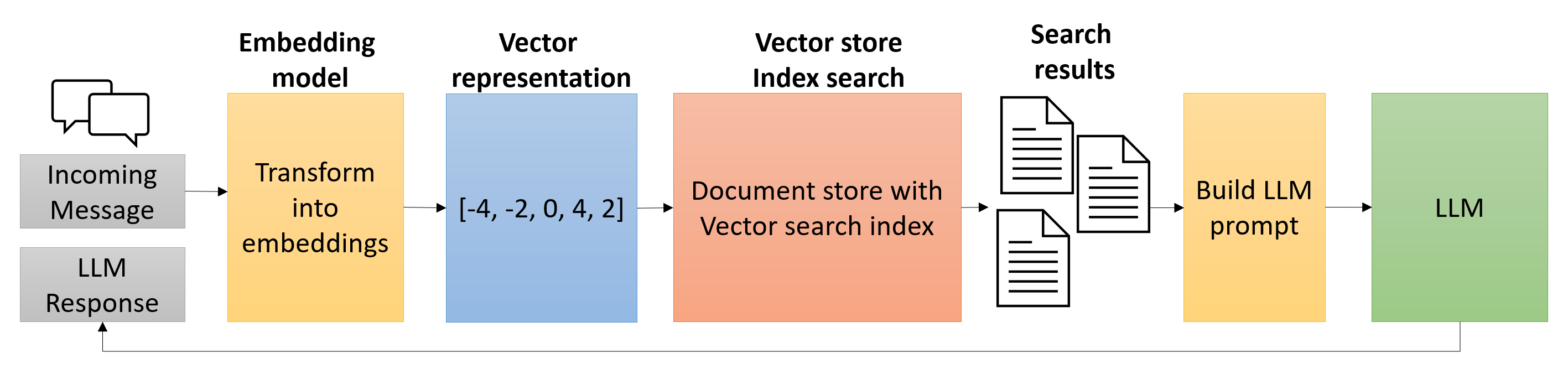
A typical embedding pipeline that demonstrates how source data is transformed into vectors using an embedding model then stored in a document in an Azure Cosmos DB vCore database and exposed via a vector search index.

## Why vector search?

Vector search is an important RAG (Retrieval Augmented Generation) pattern component. Large Language Model (LLM) data is trained on a snapshot of public data at a point in time. This data does not contain recent public information, nor does it collect private, corporate information. LLMs are also very broad in their knowledge, and including information from a RAG process can help it focus accurately on a specific domain.

A vector index search allows for a prompt pre-processing step where information can be semantically retrieved from an index and then used to generate a factually accurate prompt for the LLM to reason over. This provides the knowledge augmentation and focus (attention) to the LLM.

In this example, assume textual data is vectorized and stored within an vCore-based Azure Cosmos DB for MongoDB database. The text data and embeddings/vector field are stored in the same document. A vector search index has been created on the vector field. When a message is received from a chat application, this message is also vectorized using the same embedding model (ex., Azure OpenAI text-embedding-ada-002), which is then used as input to the vector search index. The vector search index returns a list of documents whose vector field is semantically similar to the incoming message. The unvectorized text stored within the same document is then used to augment the LLM prompt. The LLM receives the prompt and responds to the requestor based on the information it has been given.



A typical vector search request in a RAG scenario depicts an incoming message getting vectorized and used as input to a vector store index search. Multiple results of the vector search are used to build a prompt fed to the LLM. The LLM returns a response to the requestor.

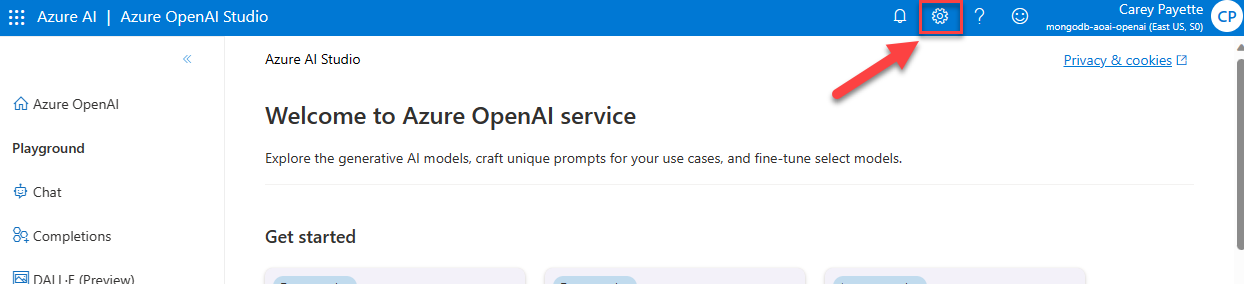
## Why use vCore-based Azure Cosmos DB for MongoDB as a vector store?

It is common practice to store vectorized data in a dedicated vector store as vector search indexing is not a common capability of most databases. However, this introduces additional complexity to the solution as the data must be stored in two different locations. vCore-based Azure Cosmos DB for MongoDB supports vector search indexing, which means that the vectorized data can be stored in the same document as the original data. This reduces the complexity of the solution and allows for a single database to be used for both the vector store and the original data.

## Lab 3 - Use vector search on embeddings in vCore-based Azure Cosmos DB for MongoDBvCore

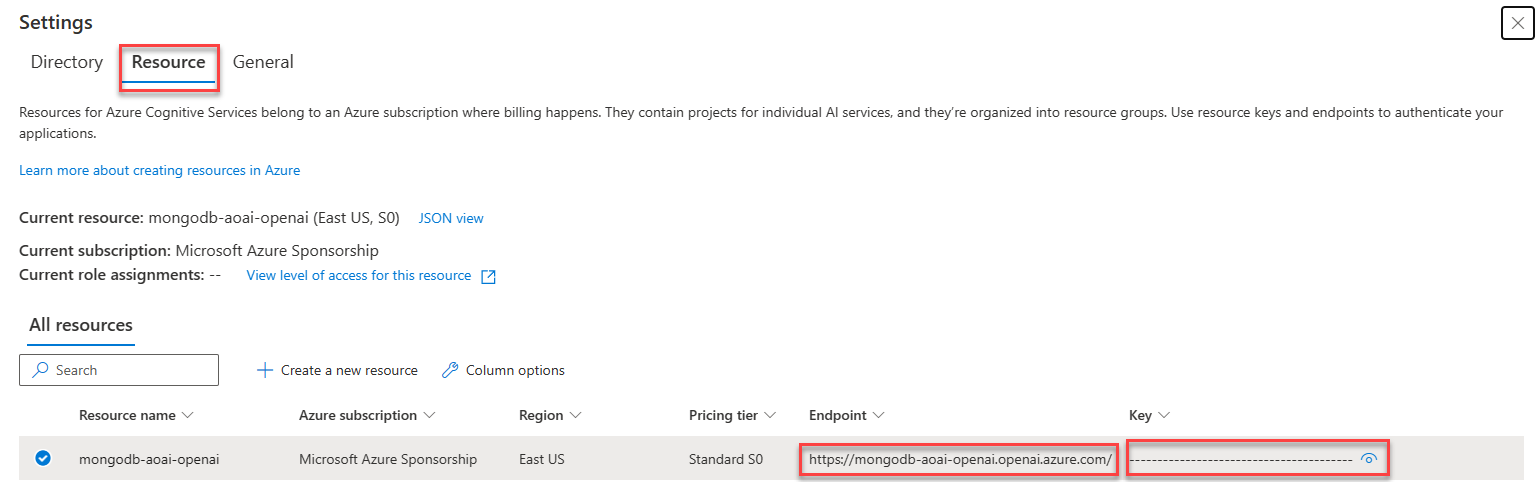
In this lab, a notebook demonstrates how to add an embedding field to a document, create a vector search index, and perform a vector search query. The notebook ends with a demonstration of utilizing vector search with an LLM in a RAG scenario.

Lab 3 requires the Azure OpenAI endpoint and access key to be added to the settings (.env) file. Access this information by opening [Azure OpenAI Studio](https://oai.azure.com/portal) and selecting the **Gear**/Settings icon located to the right in the top toolbar.



Azure OpenAI Studio displays with the Gear icon highlighted in the top toolbar.

On the **Settings** screen, select the **Resource** tab, then copy and record the **Endpoint** and **Key** values for use in the lab.



The Azure OpenAI resource settings screen displays with the endpoint and key values highlighted.

**NOTE**: This lab can only be completed using a deployed vCore-based Azure Cosmos DB for MongoDB account due to the use of vector search. The Azure Cosmos DB Emulator does not support vector search.

Please visit the lab repository to complete this lab.

# LangChain

[LangChain](https://www.langchain.com/) is an open-source framework designed to simplify the creation of applications that use large language models (LLMs). LangChain has a vibrant community of developers and contributors and is used by many companies and organizations. LangChain utilizes proven Prompt Engineering patterns and techniques to optimize LLMs, ensuring successful and accurate results through verified and tested best practices.

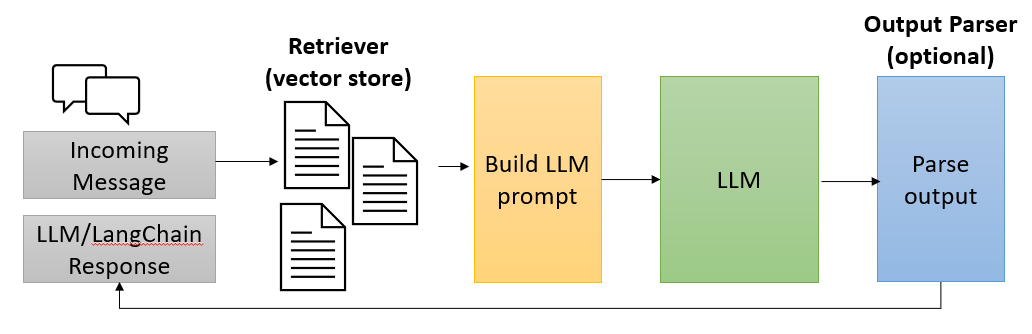
Part of the appeal of LangChain syntax is the capability of breaking down large complex interactions with LLMs into smaller, more manageable steps by composing a reusable [chain](https://python.langchain.com/docs/modules/chains/) process. LangChain provides a syntax for chains([LCEL](https://python.langchain.com/docs/modules/chains/#lcel)), the ability to integrate with external systems through [tools](https://python.langchain.com/docs/integrations/tools/), and end-to-end [agents](https://python.langchain.com/docs/modules/agents/) for common applications.

The concept of an agent is quite similar to that of a chain in LangChain but with one fundamental difference. A chain in LangChain is a hard-coded sequence of steps executed in a specific order. Conversely, an agent leverages the LLM to assess the incoming request with the current context to decide what steps or actions need to be executed and in what order.

LangChain agents can leverage tools and toolkits. A tool can be an integration into an external system, custom code, or even another chain. A toolkit is a collection of tools that can be used to solve a specific problem.

## LangChain RAG pattern

Earlier in this guide, the RAG (Retrieval Augmented Generation) pattern was introduced. In LangChain, the RAG pattern is implemented as part of a chain that combines a retriever and a Large Language Model (generator). The retriever is responsible for finding the most relevant documents for a given query, in this case, doing a vector search on vCore-based Azure Cosmos DB for MongoDB, and the LLM (generator) is responsible for reasoning over the incoming prompt and context.



LangChain RAG diagram shows the flow of an incoming message through a retriever, augmenting the prompt, parsing the output and returning the final message.

When an incoming message is received, the retriever will vectorize the message and perform a vector search to find the most relevant documents for the given query. The retriever returns a list of documents that are then used to augment the prompt. The augmented prompt is then passed to the LLM (generator) to reason over the prompt and context. The output from the LLM is then parsed and returned as the final message.

**Note**: A vector store retriever is only one type of retriever that can be used in the RAG pattern. Learn more about retrievers in the [LangChain documentation](https://python.langchain.com/docs/modules/data_connection/retrievers/).

## Lab 4 - Vector search and RAG using LangChain

In this lab, you will learn to use LangChain to re-implement the RAG pattern introduced in Lab 3. Take note of the readability of the code and how easy it is to compose a reusable RAG chain using LangChain that queries the products vector index in vCore-based Azure Cosmos DB for MongoDB. Lab 4 concludes with the creation of an agent with various tools for the LLM to leverage to fulfill the incoming request.

# Lab 5 - Backend API

In the previous lab, a LangChain agent was created armed with tools to do vector lookups and concrete document id lookups via function calling. In this lab, the agent functionality needs to be extracted into a backend api for the frontend application that will allow users to interact with the agent.

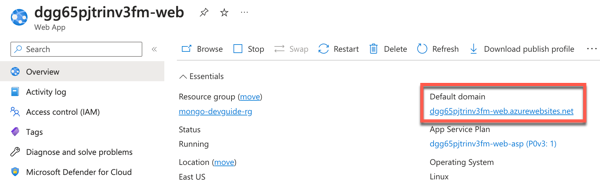
Please visit the lab repository to complete this lab.

# Connect the chat user interface with the chatbot API

In the previous lab, you configured and deployed the Backend API code that integrates vCore-based Azure Cosmos DB for MongoDB with Azure OpenAI. When the Azure resource template for this lab was run to deploy the necessary Azure resources, a front-end web application written as a SPA (single page application) in React was deployed.

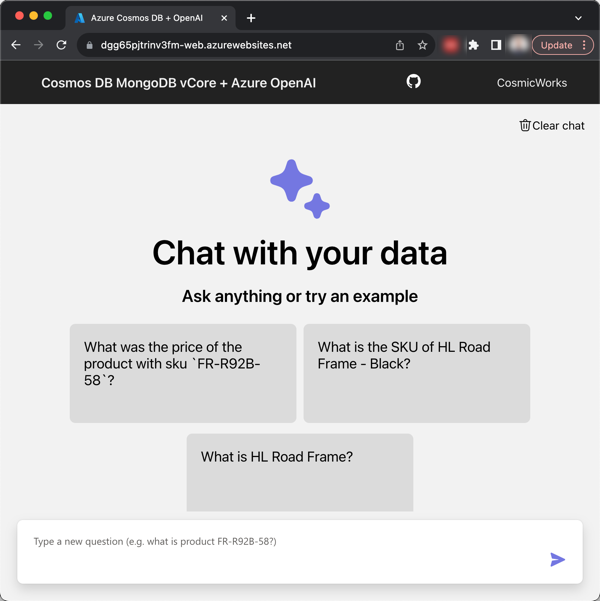
You can find the URL to access this front-end application within the Azure Portal on the **Web App** resource with the name that ends with **-web**.

The following screenshot shows where to find the front-end application URL:



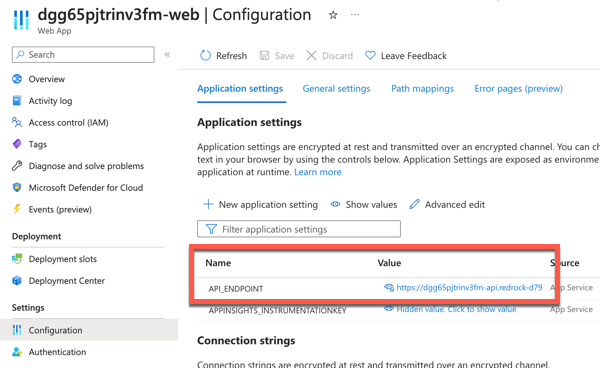
Web App resource for front-end application with Default domain highlighted

Navigating to this URL in the browser will allow you to access the front-end application. Through this front-end applications User Interface, you will be able to ask the Azure OpenAI model questions about the CosmicWorks company data, then it will generate responses accordingly.



Front-end Web Application User Interface

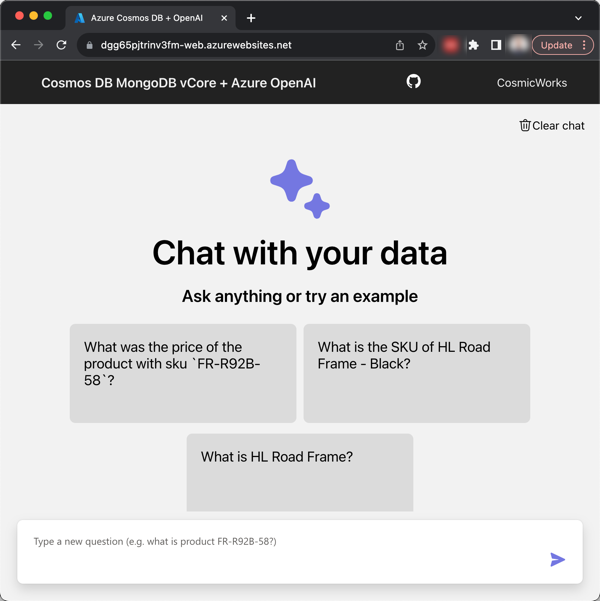
While the code for the SPA web application is outside the scope of this dev guide. It’s worth noting that the Web App is configured with the URL for the Backend API using the **Application setting** named API\_ENDPOINT. When the application was deployed as part of the Azure template deployment, it was automatically configured with this URL to connect the front-end SPA web application to the Backend API.



Web App resource showing the application settings with the API\_ENDPOINT setting highlighted

## Ask questions about your data and observe the responses

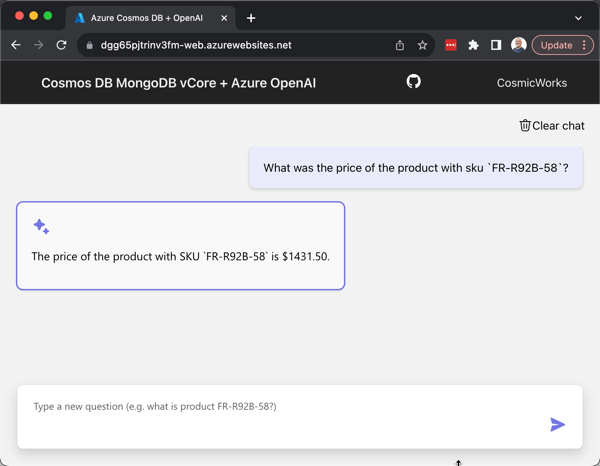
To ask the AI questions about the CosmicWorks company data, you can type your questions in to the front-end applications chat user interface. The web application includes tiles with a couple example questions to get you started. To use these, simply click on the question tile and it will generate an answer.



Front-end Web Application User Interface

These example questions are: - What was the price of the product with sku FR-R92B-58? - What is the SKU of HL Road Frame - Black? - What is HL Road Frame?

The chat user interface presents as a traditional chat application style interface when asking questions.



Chat user interface screenshot with question and generated answer displayed

Go ahead, ask the service a few questions about CosmicWorks and observice the responses.

## What do I do if the responses are incorrect?

It’s important to remember the model is pre-trained with data, given a system message to guide it, in addition to the company data it has access to via vCore-based Azure Cosmos DB for MongoDB. There are times when the Azure OpenAI model may generate an incorrect response to the prompt given that is either incomplete or even a hallucination (aka includes information that is not correct or accurate).

There are a few options of how this can be handled when the response is incorrect:

1. Provide a new prompt that includes more specific and structured information that can help the model generate a more accurate response.
2. Include more data in the library of company information the model has access to. The incorrect response may be a result of data or information about the company that is missing currently.
3. Use Prompt Engineering techniques to enhance the System message and/or Supporting information provided to guide the model.

While it may be simple to ask the model questions, there are times when Prompt Engineering skills may be necessary to get the most value and reliable responses from the AI model.

## What happens when I start exceeding my token limits?

A Token in Azure OpenAI is a basic unit of input and output that the service processes. Generally, the models understand and process text by breaking it down into tokens.

For example, the word hamburger gets broken up into the tokens ham, bur and ger, while a short and common word like pear is a single token. Many tokens start with a whitespace, for example hello and bye.

The total number of tokens processed in a given request depends on the length of your input, output and request parameters. The quantity of tokens being processed will also affect your response latency and throughput for the models.

**Note**: The [pricing of the Azure OpenAI](https://azure.microsoft.com/en-us/pricing/details/cognitive-services/openai-service/) service is primarily based on token usage.

### Exceeding Token Quota Limits

Azure OpenAI has **tokens per minute** quota limits on the service. This quota limit, is based on the OpenAI model being used and the Azure region it’s hosted in.

**Note**: The [Azure OpenAI Quotas and limits documentation](https://learn.microsoft.com/en-us/azure/ai-services/openai/quotas-limits) contains further information on the specific quotas per OpenAI model and Azure region.

If an applications usage of an Azure OpenAI model exceeds the token quota limits, then the service will respond with a **Rate Limit Error** (Error code 429).

When this error is encountered, there are a couple options available for handling it:

* **Wait a minute** - With the tokens quota limit being a rate limits of the maximum number of tokens allowed per minute, the application will be able to send more prompts to the model again after the quota resets each minute.
* **Request a quota increase** - It may be possible to get Microsoft to increase your token quota to a higher limit, but it’s not guaranteed you will be able to get an increase. This request can be made at <https://aka.ms/oai/quotaincrease>

### Tips to Minimize Token Rate Limit Errors

Here are a few tips that can help to minimize an applications token rate limit errors:

* **Retry Logic** - Implement retry logic in the application so it will retry the call to the Azure OpenAI model, rather than throwing an exception the first time. This is generally best practices when consuming external APIs from applications so they can gracefully handle any unexpected exceptions.
* **Scale Workload Gradually** - Avoid increasing the workload of the application too quickly. By gradually increasing the scale of the workload.
* **Asynchronous Load Patterns** - While there are time sensitive operations that will require a response immediately, there are also operations that are able to be run more asynchronously. Background processes or other similar operations could be build in a way to perform a combination of rate limiting the applications own usage of the model, or even delaying calls until periods of the day where the application is under less load.
* **Set max\_tokens** - Setting a lower max\_tokens when calling the service when a short response is expected will limit the maximum number of tokens allowed for the generated answer.
* **Set best\_of** - Setting a lower best\_of when calling the service enables the application to control the number of candidate completions generated and how many to return from the service.

### Exceeding Token Limit for System message

When configuring a System message to guide the generated responses, there is a limit on how long the System message can be. The token limit for the System message is 400 tokens.

If the System message provided is more than 400 tokens, the rest of the tokens beyond the first 400 will be ignored.