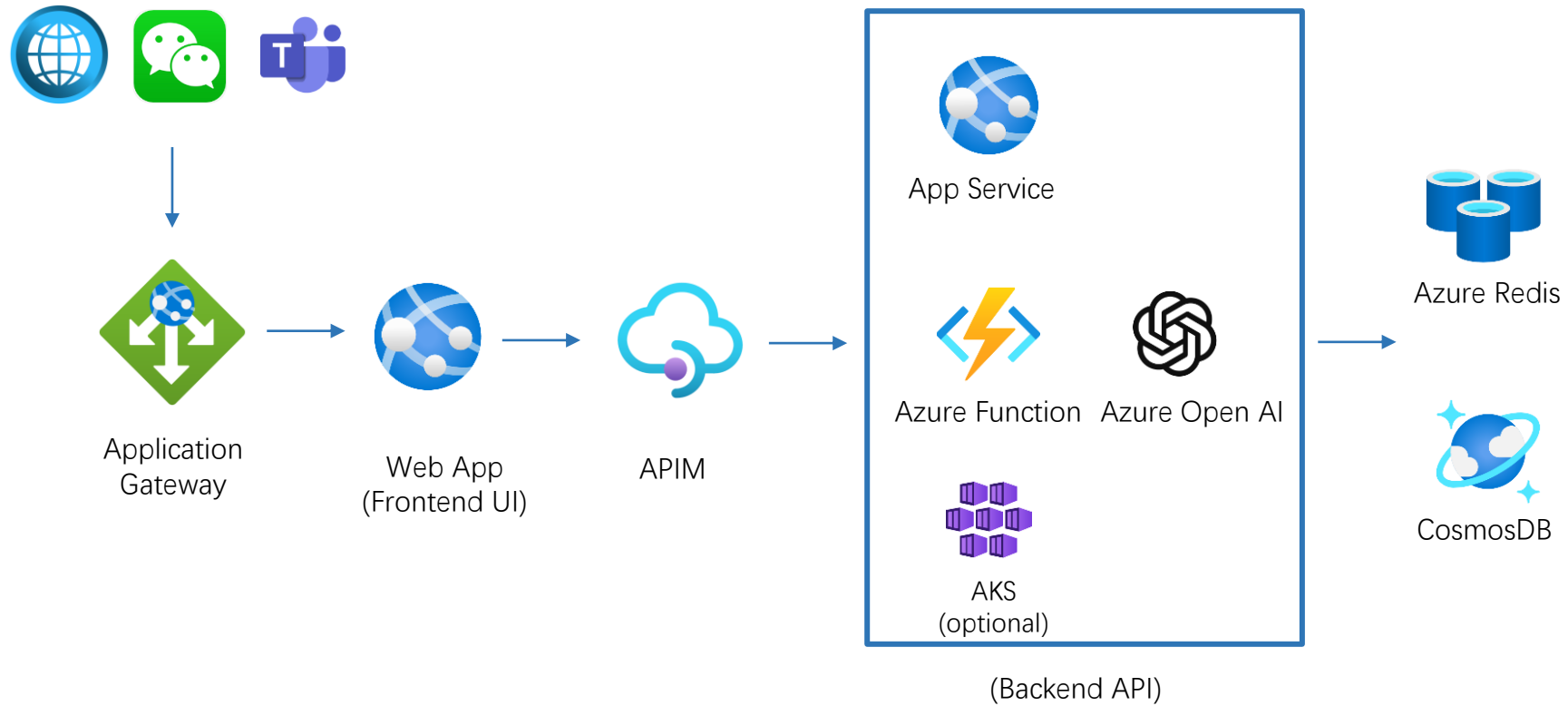
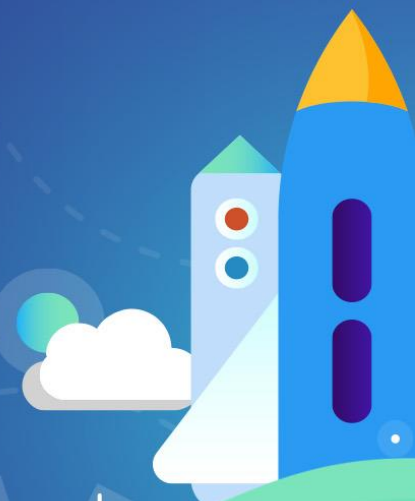
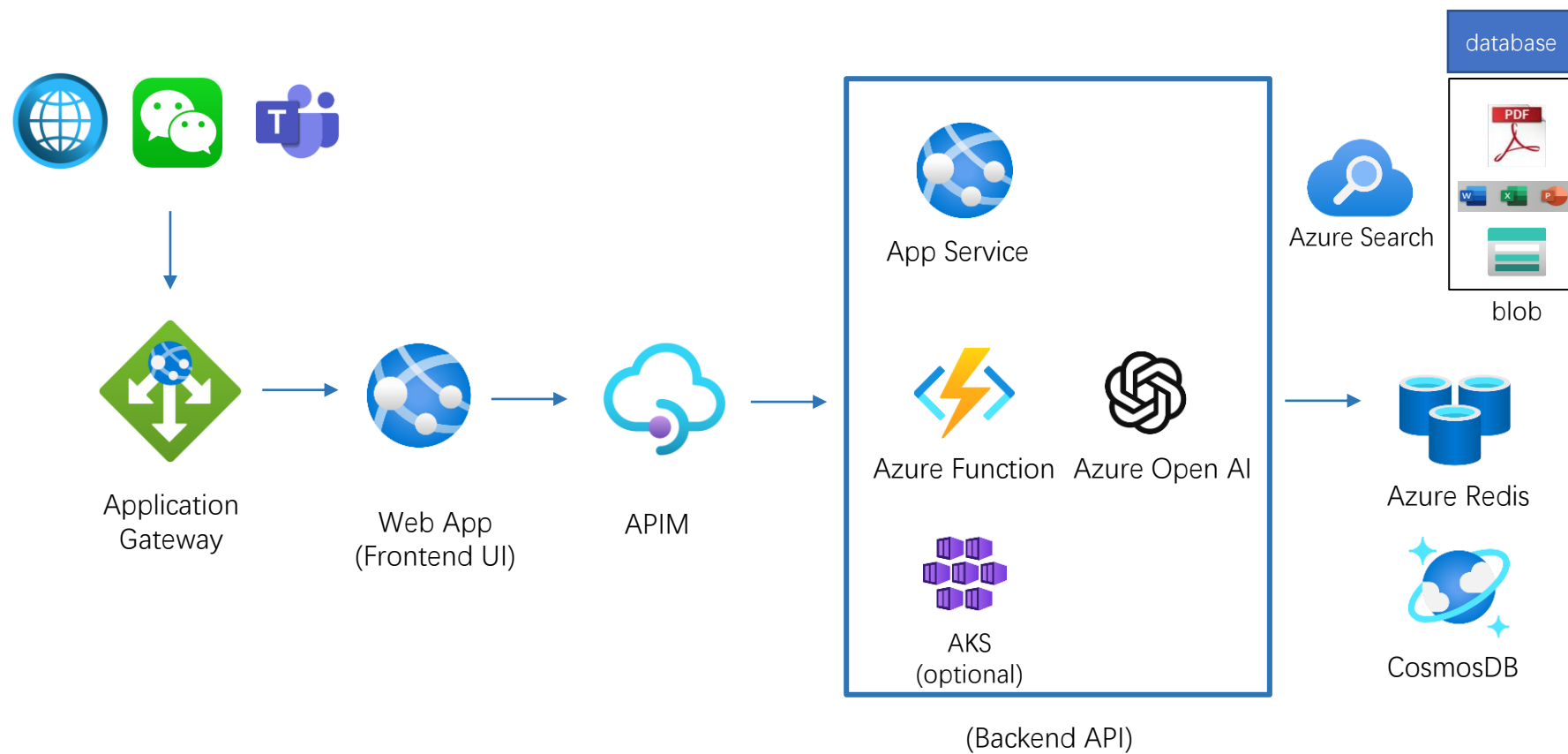


# FY23 Q3 Azure In A Day 沉浸式体验之旅





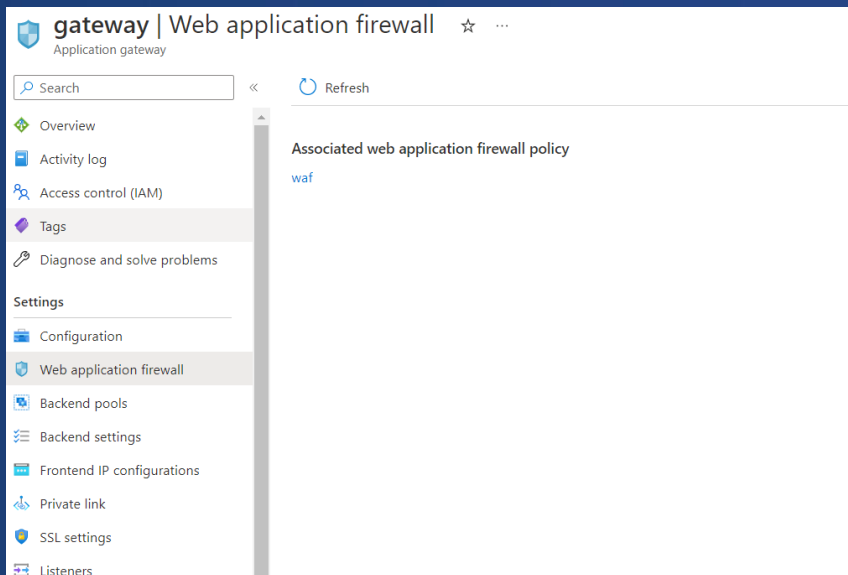






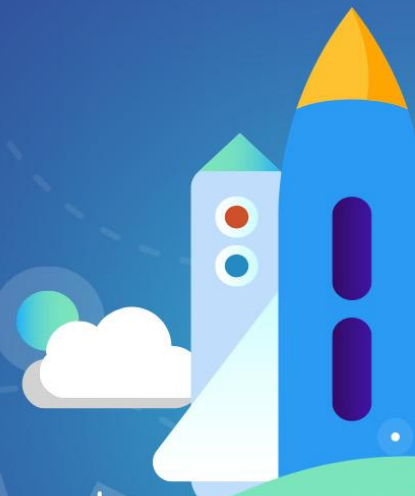
企业知识 (来自内部pdf文件)



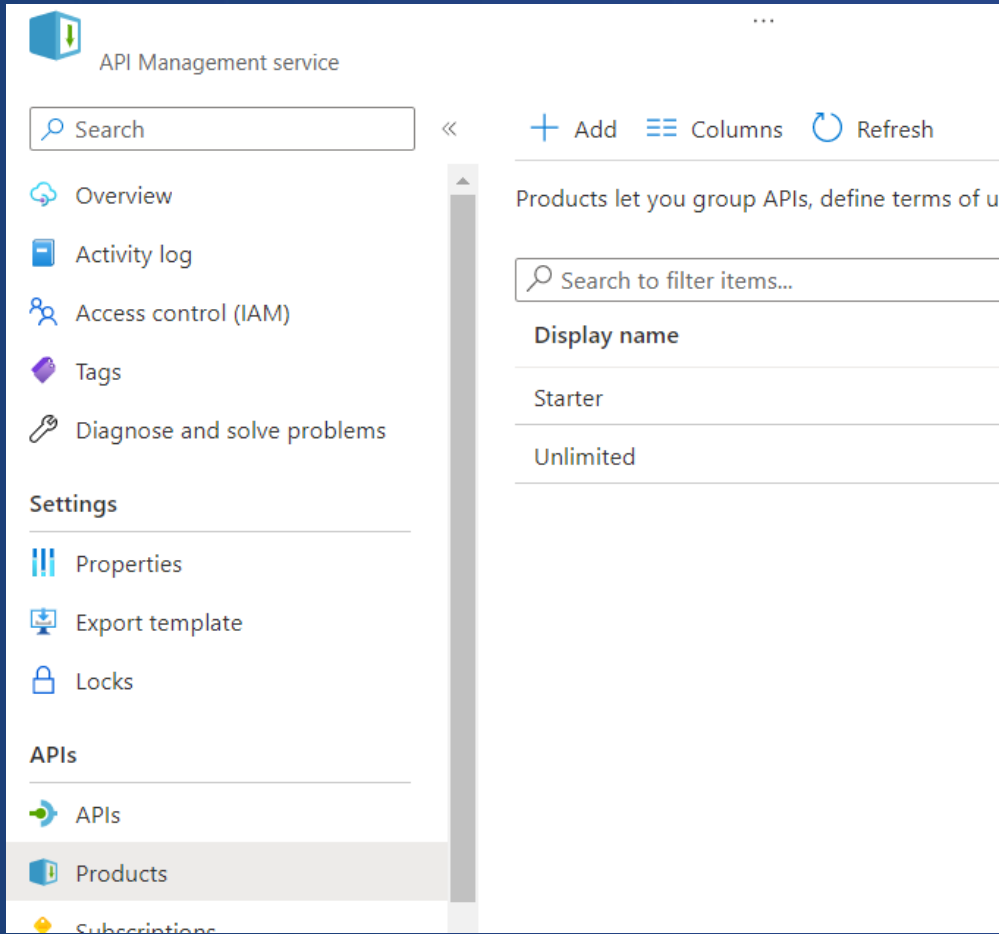


## 创建应用程序网关 (Application Gateway)

1. 选择 Azure 门户左侧菜单上的“创建资源”。此时会显示“新建”窗口。
2. 选择“网络”，然后在“特色”列表中选择“应用程序网关”。
3. 在“基本信息”选项卡上，输入这些值作为以下应用程序网关设置：
  - **资源组**：选择“新建”，创建一个新的。
  - **应用程序网关名称**：输入 *myAppGateway* 作为应用程序网关的名称。
  - **层**：选择“WAF V2”。
  - **WAF 策略**：选择新建，键入新策略的名称，然后选择确定。这会创建具有托管核心规则集 (CRS) 的基本 WAF 策略。
4. 设置前端选项卡：选择“公共”
5. 设置后端选项卡：选择“不包含目标的后端池”（后续步骤再配置目标）
6. 设置“配置”选项卡”

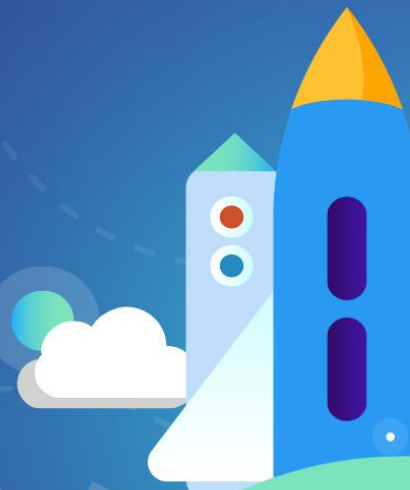




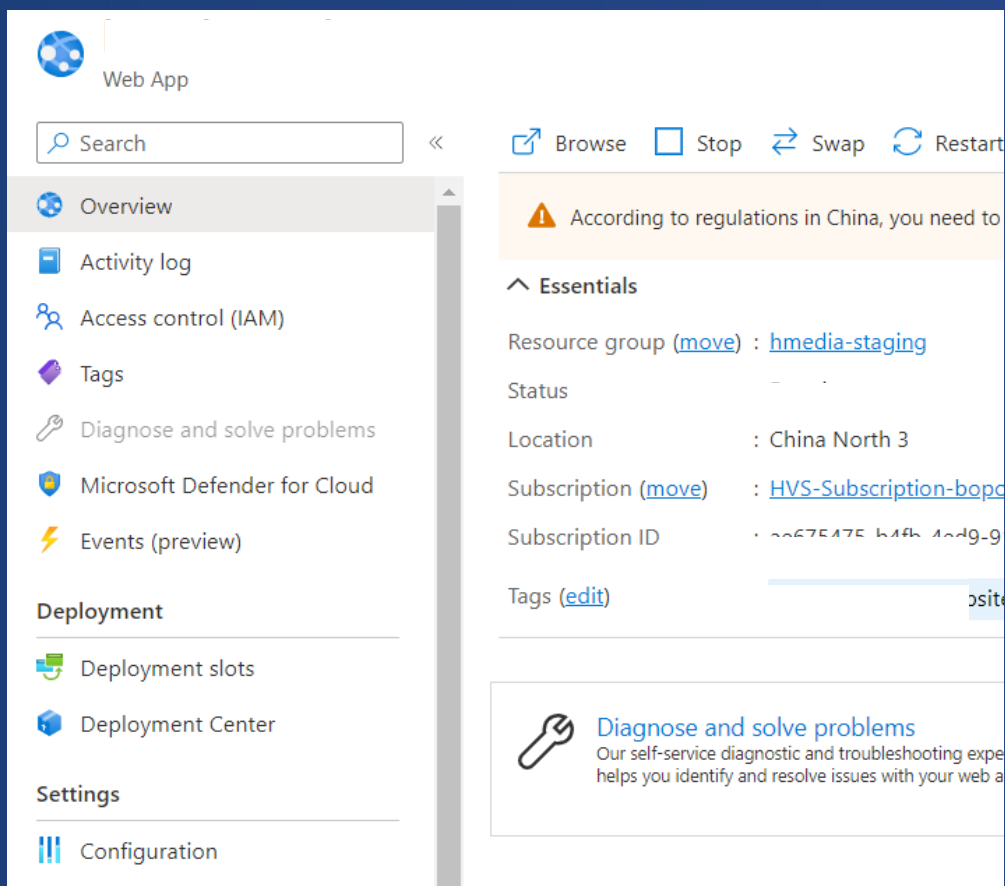


## 创建Azure API 管理 (APIM)

1. 在 Azure 门户菜单中，选择“创建资源”。还可以在 Azure“主页”上选择“创建资源”。
2. 在“创建资源”页上，选择“集成”>“API 管理”。
3. 在“创建 API 管理”页中，输入设置
4. 选择“查看 + 创建”。
5. 导入和发布API (后续步骤)
6. 包含你的API (后续步骤)



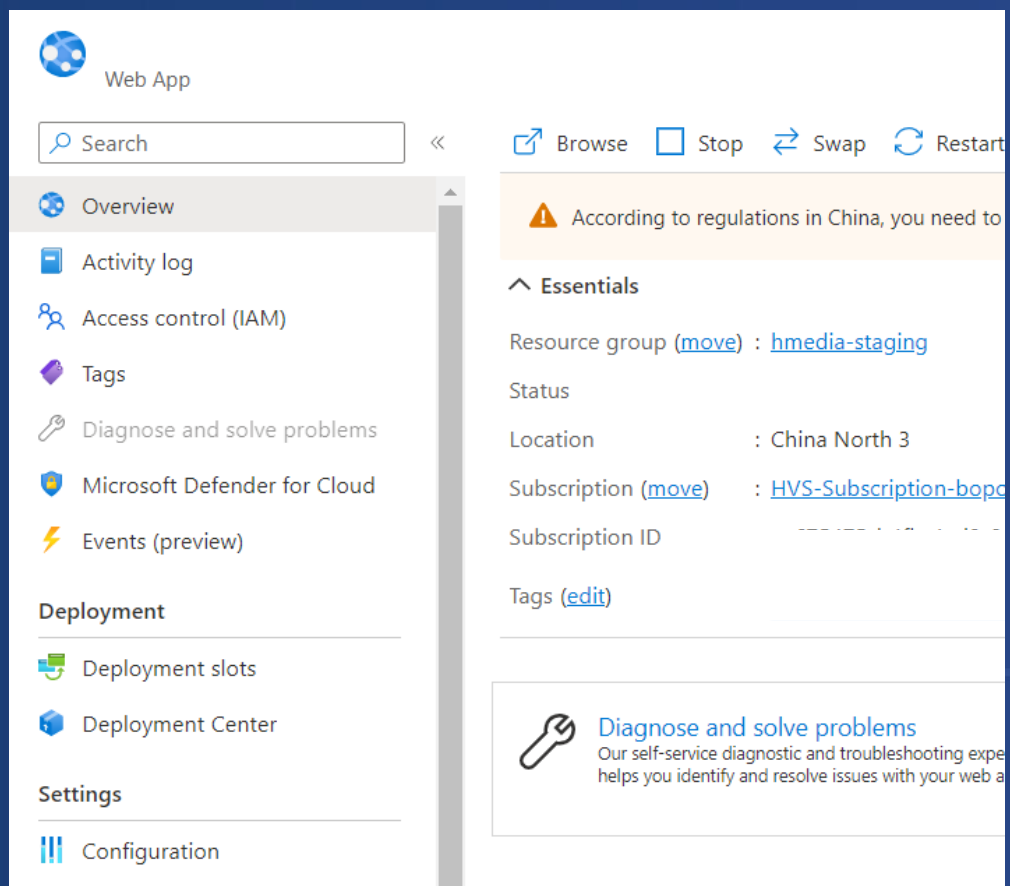




## 创建应用程序服务 (App Service) :前端UI应用

1. 在Azure门户中搜索框中键入“应用服务”。在“服务”下选择“应用程序服务”
2. 在“应用服务”页面中，请选择“+ 创建”。
3. 在“基本信息”选项卡中的“项目详细信息”下，确保选择了正确的订阅，然后选择“新建”来新建资源组。
4. 设置“实例信息”选项卡：
  - 在“名称”下，为 Web 应用键入全局唯一名称。
  - 在“发布”下，选择“代码”。
  - 在“运行时堆栈”下，选择“.NET 6 (LTS)”。
  - 选择“操作系统”：Windows或者Linux均可以。
  - 选择实例运行的“区域”：任意。
5. 在“应用服务计划”下，选择“新建”来新建应用服务计划：
  - 键入名称。
  - 选择“更改大小”，选择一个定价层，例如S3或者P3V2。
6. 部署前端应用 (后续步骤)

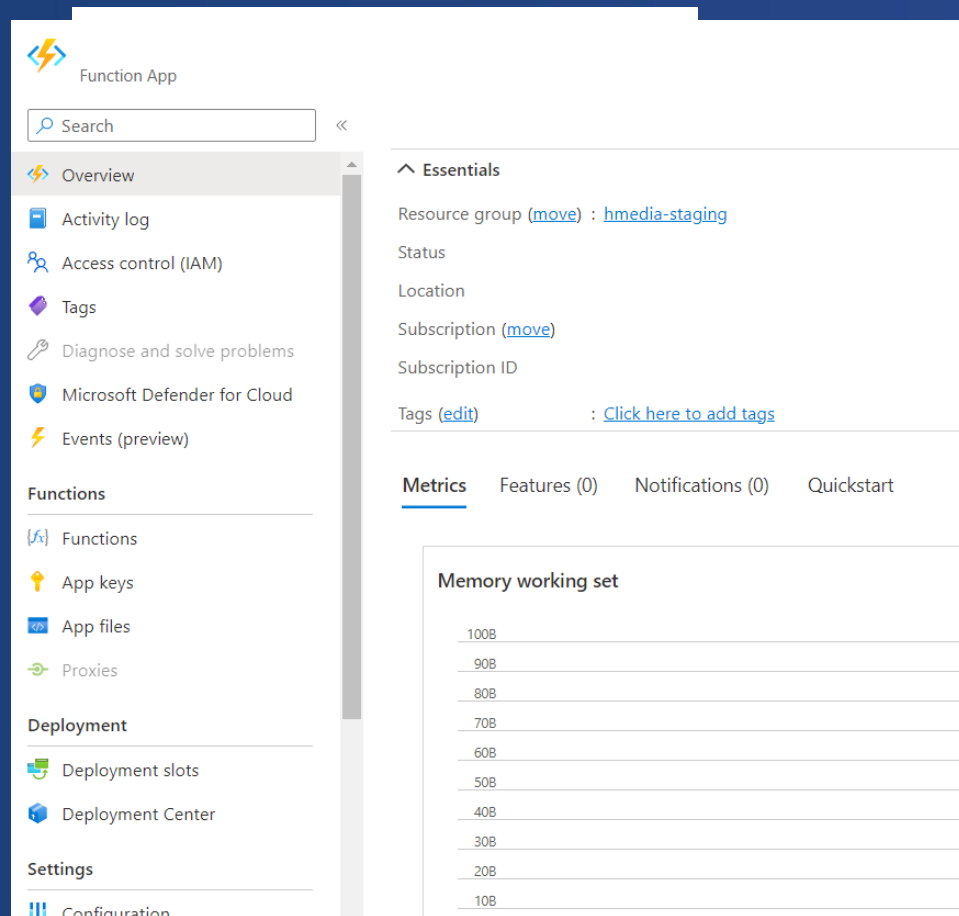




## 创建应用程序服务 (App Service): 后端API服务

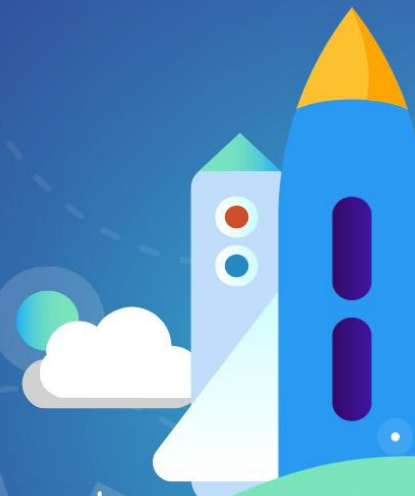
1. 在Azure门户中搜索框中键入“应用服务”。在“服务”下选择“应用程序服务”
2. 在“应用服务”页面中，请选择“+ 创建”。
3. 在“基本信息”选项卡中的“项目详细信息”下，确保选择了正确的订阅，然后选择“新建”来新建资源组。
4. 设置“实例信息”选项卡：
  - 在“名称”下，为 Web 应用键入全局唯一名称。
  - 在“发布”下，选择“代码”。
  - 在“运行时堆栈”下，选择“.NET 6 (LTS)”。
  - 选择“操作系统”：选择Linux (注意本次Workshop后端流式API需使用Linux OS)
  - 选择实例运行的“区域”：任意。
5. 在“应用服务计划”下，选择“新建”来新建应用服务计划：
  - 键入名称。
  - 选择“更改大小”，选择一个定价层，例如S3或者P3V2。
6. 部署后端API服务 (后续步骤)

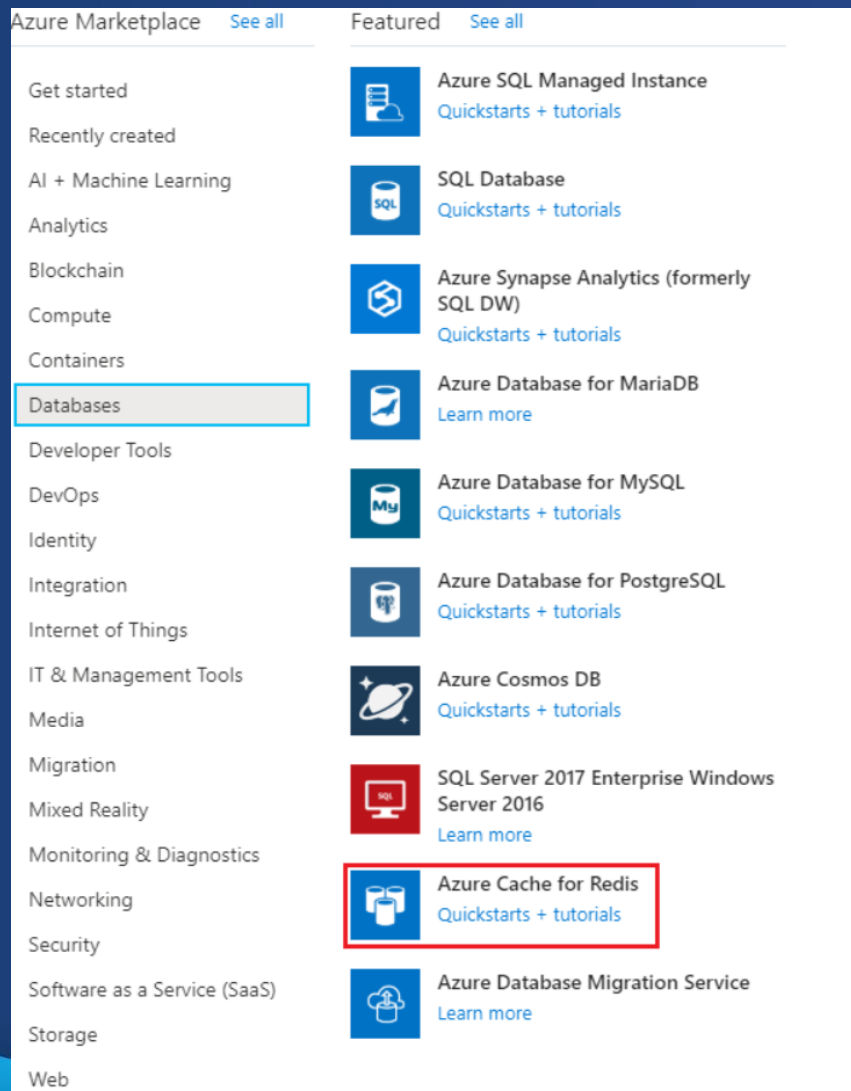




## 创建函数应用服务 (Azure Function): 后端API服务

1. 在 Azure 门户菜单上或在门户主页中，选择“创建资源”。
2. 在“新建”页面，选择“计算”>“函数应用”。
3. 在“基本信息”页面上，设置函数应用：
  - 订阅
  - 资源组
  - 名称
  - 运行时语言：C#类库
  - 版本：最新
4. 选择“下一步: 托管”。设置：
  1. 存储账号
  2. 操作系统: Windows
  3. 计划：选择“消耗”（无服务模式）
5. 下一步
6. 创建
7. 部署API服务 （后续步骤）





## 创建缓存服务(Azure Redis)

1. 登录到 Azure 门户并选择“创建资源”。
2. 在“新建”页上选择“数据库”，然后选择“Azure Cache for Redis”。
3. 在“新建 Redis 缓存”页上配置新缓存的设置：
  - 订阅
  - 资源组
  - DNS名称：唯一名称
  - 位置
  - 缓存类型：选择一个定价层，例如“C6或者P3”
4. 下一步设置网络
5. 下一步“高级”，选择最新的Redis版本
6. 创建
7. 获取Redis连接字符串（后续步骤）



## 创建CosmosDB

- 1.在 Azure 门户菜单或主页中，选择“创建资源”。
- 2.搜索“Azure Cosmos DB”。选择“创建”>“Azure Cosmos DB”。
- 3.在“创建 Azure Cosmos DB 帐户”页上，选择“Azure Cosmos DB for NoSQL”部分中的“创建”选项。
- 4.选择DB API：NoSQL
- 5.在“创建 Azure Cosmos DB 帐户”页中，输入新 Azure Cosmos DB 帐户的基本设置：
  - 订阅
  - 资源组
  - 帐户名：唯一名称
  - 区域
  - 容量模式：选择“预配吞吐量”，设置最大吞吐量7000（或选择“无服务器模式”）
6. 下一步
7. 创建
8. 获取数据库连接字符串（后续步骤）

Home > Azure Cosmos DB >

### Create Azure Cosmos DB Account

**Basics** Global Distribution Networking Back

Azure Cosmos DB is a globally distributed, multi-model, fully managed database service.

**Project Details**

Select the subscription to manage deployed resources and create new resources in.

Subscription \*

Resource Group \*  [Create new](#)

**Instance Details**

Account Name \*

API \*  ⓘ

Location \*

Capacity mode ⓘ ☒ Provisioned [Learn more about capacity modes](#)

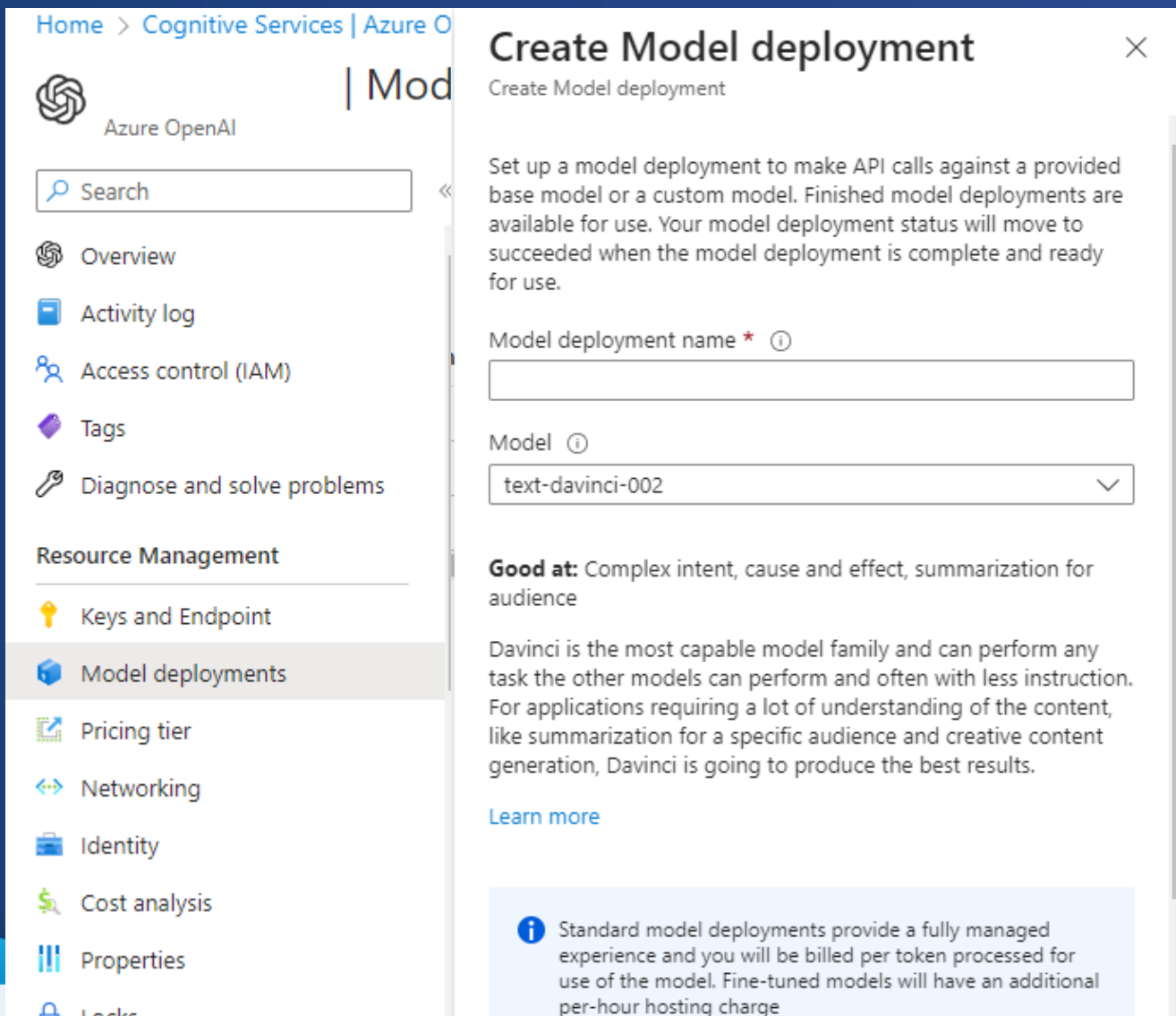
With Azure Cosmos DB free tier, you will get the first 100 GB of storage free for 90 days.

Apply Free Tier Discount ☒ Apply (Free tier discount applies to new accounts only.)

Limit total account throughput ☒ Limit the throughput of the account [Learn more about throughput limits](#)

☒ This limit applies to the account throughput.





The screenshot shows the 'Create Model deployment' page in the Azure portal. The left sidebar contains navigation links: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management (Keys and Endpoint, Model deployments, Pricing tier, Networking, Identity, Cost analysis, Properties, Locks), and Mod. The main content area is titled 'Create Model deployment' and includes a description of the process, a form for 'Model deployment name' and 'Model' (set to 'text-davinci-002'), and a 'Good at' section describing the model's capabilities. A 'Learn more' link is also present.

Home > Cognitive Services | Azure OpenAI | Model deployments

## Create Model deployment

Create Model deployment

Set up a model deployment to make API calls against a provided base model or a custom model. Finished model deployments are available for use. Your model deployment status will move to succeeded when the model deployment is complete and ready for use.

Model deployment name \* ⓘ

Model ⓘ

text-davinci-002

**Good at:** Complex intent, cause and effect, summarization for audience

Davinci is the most capable model family and can perform any task the other models can perform and often with less instruction. For applications requiring a lot of understanding of the content, like summarization for a specific audience and creative content generation, Davinci is going to produce the best results.

[Learn more](#)

**i** Standard model deployments provide a fully managed experience and you will be billed per token processed for use of the model. Fine-tuned models will have an additional per-hour hosting charge

## 创建Azure OpenAI

- 1.在 Azure 门户菜单或主页中, 选择“创建资源”。
- 2.搜索“Azure OpenAI”。 选择“创建”
- 3.创建部署模型: 选择text-davinci-003 (GPT3模型)
- 4.“在Keys and Endpoint”获取实例的
  - API地址
  - Key

## 代码解析: 关于 Azure OpenAI restful API

POST <https://{your-resource-name}.openai.azure.com/openai/deployments/{deployment-id}/completions?api-version={api-version}>

Path参数	类型	是否必填	说明
your-resource-name	string	是	Azure OpenAI Resource.
deployment-id	string	是	部署实例名称
api-version	string	是	YYYY-MM-DD 格式

Body 参数	类型	默认值	说明
prompt	string		提示词
max_tokens	int	16	最大tokens数
stream	bool	false	YYYY-MM-DD 格式

```
var options = new CompletionsOptions
{
    Prompt = { prompt },
    MaxTokens = MaxTokens
};
var completions = await _client.GetCompletionsAsync(_config.DeploymentId, options);
var completion = completions.Value.Choices[0].Text;
return completion;
```

<https://learn.microsoft.com/en-us/azure/cognitive-services/openai/reference#completions>



## 代码解析: 关于使用Azure Redis缓存服务

1. 通过应用层缓存降低Azure OpenAI的调用频次
2. 自定义上下文缓存机制比较弱
3. 通过Redis Queue提升应用层性能

```
public class Engine: IEngine
{
    private readonly EngineConfig _config;
    private readonly OpenAIClient _client;
    private readonly CosmosClient _cosmosClient;
    public Engine(EngineConfig config)
    {
        var cachedCompletion = await TryGetCachedCompletionAsync(userId, prompt);
        if (cachedCompletion != null)
        {
            return cachedCompletion;
        }
        else
        {
            var options = new CompletionsOptions
            {
                Prompt = { prompt }
            };
            var completions = await _client.GetCompletionsAsync(_config.DeploymentId, options);
            var completion = completions.Value.Choices[0].Text;
            await AddToCacheAsync(userId, prompt, completion);
            await SaveToDatabaseAsync(userId, prompt, completion);
            return completion;
        }
    }
}
```

关于Redis Windows 本地开发:

<https://learn.microsoft.com/en-us/azure/azure-cache-for-redis/cache-development-faq>

<https://github.com/microsoftarchive/redis/releases>



## 代码解析: 关于使用Cosmos DB

1. 会话记录
2. Embeddings 存储 (领域知识库)

```
var database = _cosmosClient.GetDatabase("completionsDB");

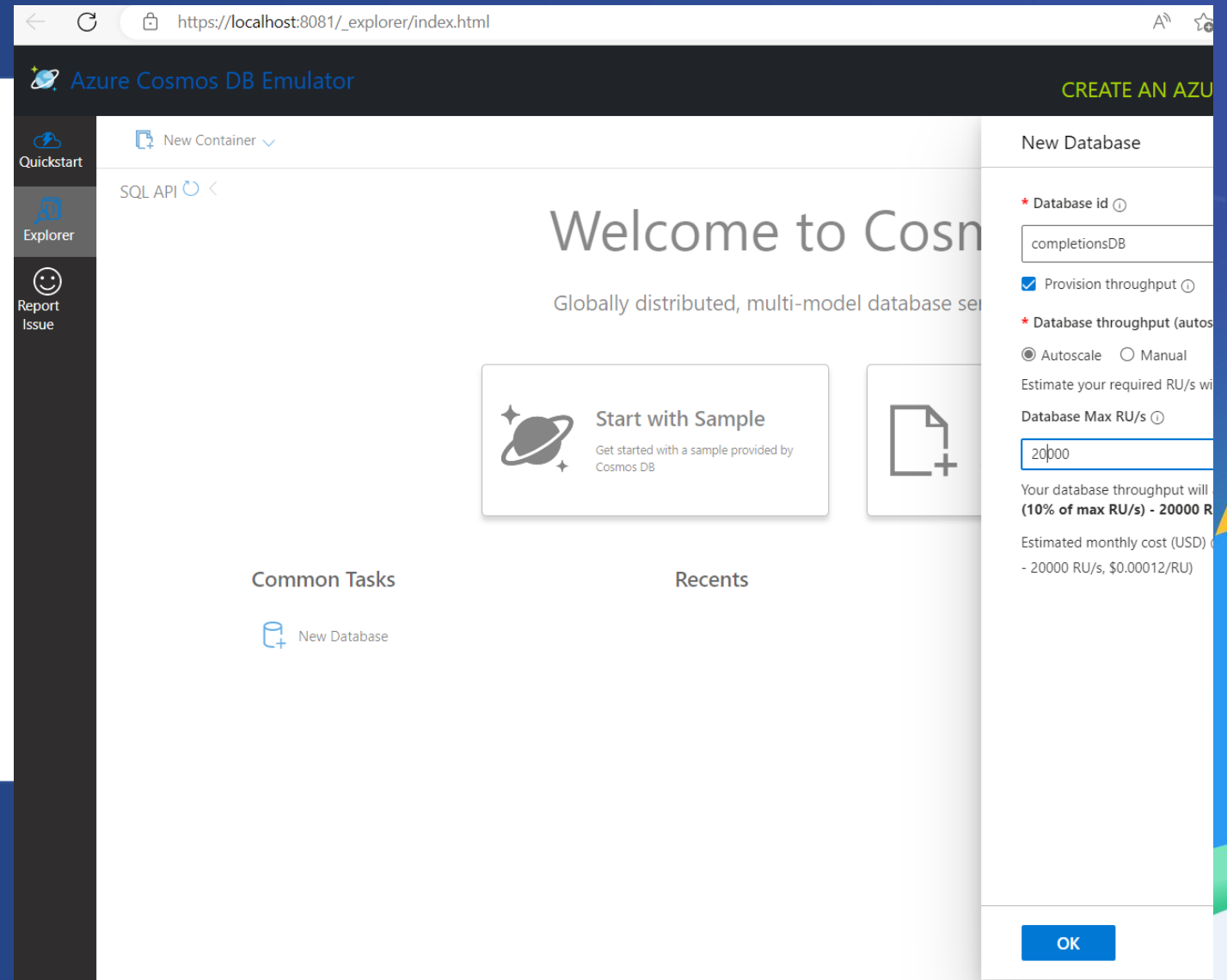
var containerResponse = await database.CreateContainerIfNotExistsAsync(
    id: "completions",
    partitionKeyPath: "/userId",
    throughput: 400
);

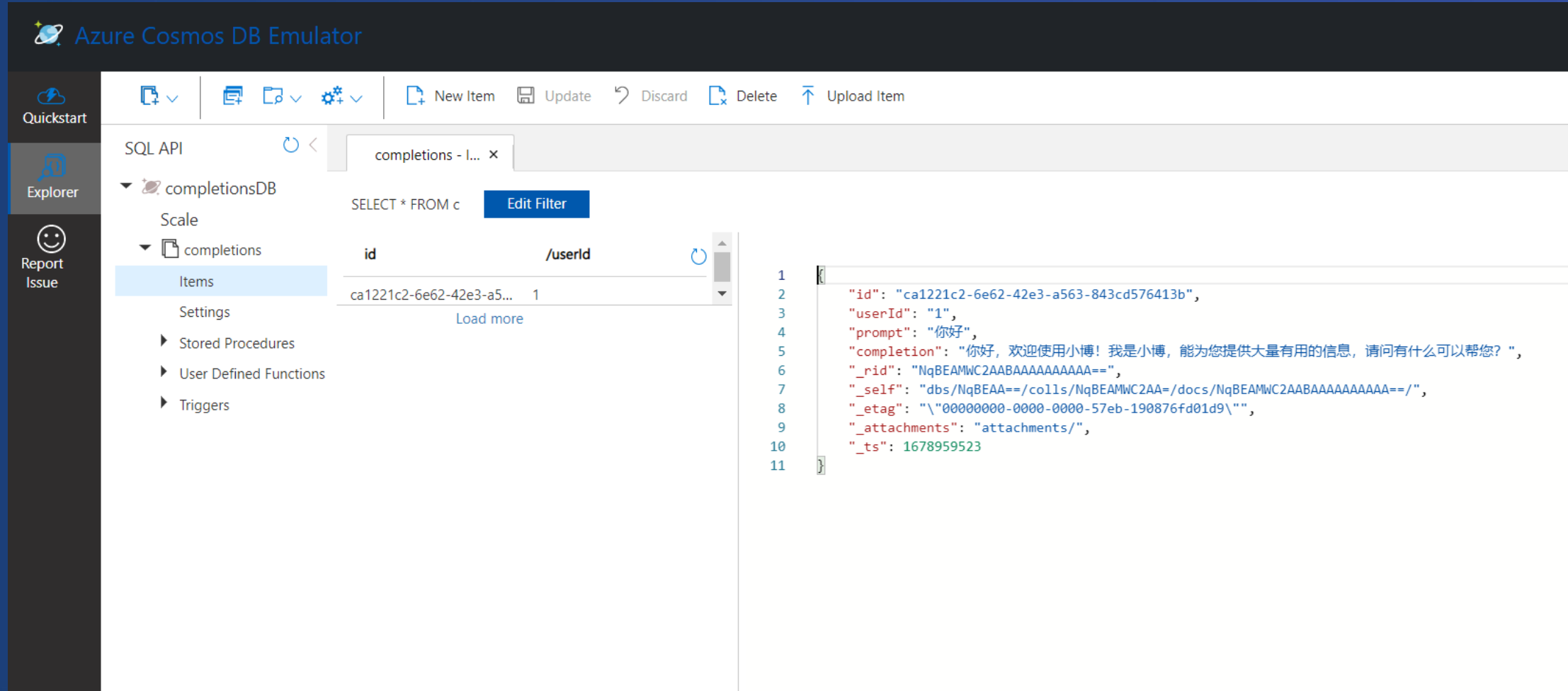
await containerResponse.Container.CreateItemAsync<CompletionCacheItem>(new CompletionCacheItem
{
    UserId = userId,
    Prompt = prompt,
    Completion = completion
});
```

关于CosmosDB本地开发：

<https://learn.microsoft.com/en-us/azure/cosmos-db/local-emulator?tabs=ssl-netstd21>

[https://localhost:8081/\\_explorer/index.html](https://localhost:8081/_explorer/index.html)





The screenshot displays the Azure Cosmos DB Emulator interface. On the left, a sidebar contains 'Quickstart', 'Explorer', and 'Report Issue' buttons. The 'Explorer' panel shows a tree view with 'completionsDB' expanded, containing 'Scale', 'completions' (with 'Items' selected), 'Settings', 'Stored Procedures', 'User Defined Functions', and 'Triggers'. The main area shows a SQL query 'SELECT \* FROM c' with an 'Edit Filter' button. Below the query, a table displays the results of the query, with columns 'id' and '/userId'. The first row shows 'ca1221c2-6e62-42e3-a5...' and '1'. A 'Load more' button is visible below the table. On the right, a code editor shows the JSON representation of the query result, with line numbers 1 through 11 on the left margin.

```
1 {
2   "id": "ca1221c2-6e62-42e3-a563-843cd576413b",
3   "userId": "1",
4   "prompt": "你好",
5   "completion": "你好, 欢迎使用小博! 我是小博, 能为您提供大量有用的信息, 请问有什么可以帮您? ",
6   "_rid": "NqBEAMWC2AABAAAAAAAAA==",
7   "_self": "dbs/NqBEAA==/colls/NqBEAMWC2AA=/docs/NqBEAMWC2AABAAAAAAAAA==/",
8   "_etag": "\"00000000-0000-0000-57eb-190876fd01d9\"",
9   "_attachments": "attachments/",
10  "_ts": 1678959523
11 }
```

## 代码解析: 关于配置信息

必须配置如下信息:

AzureOpenAIAPIURL

AzureOpenAIKey

AzureOpenAIDeploymentId

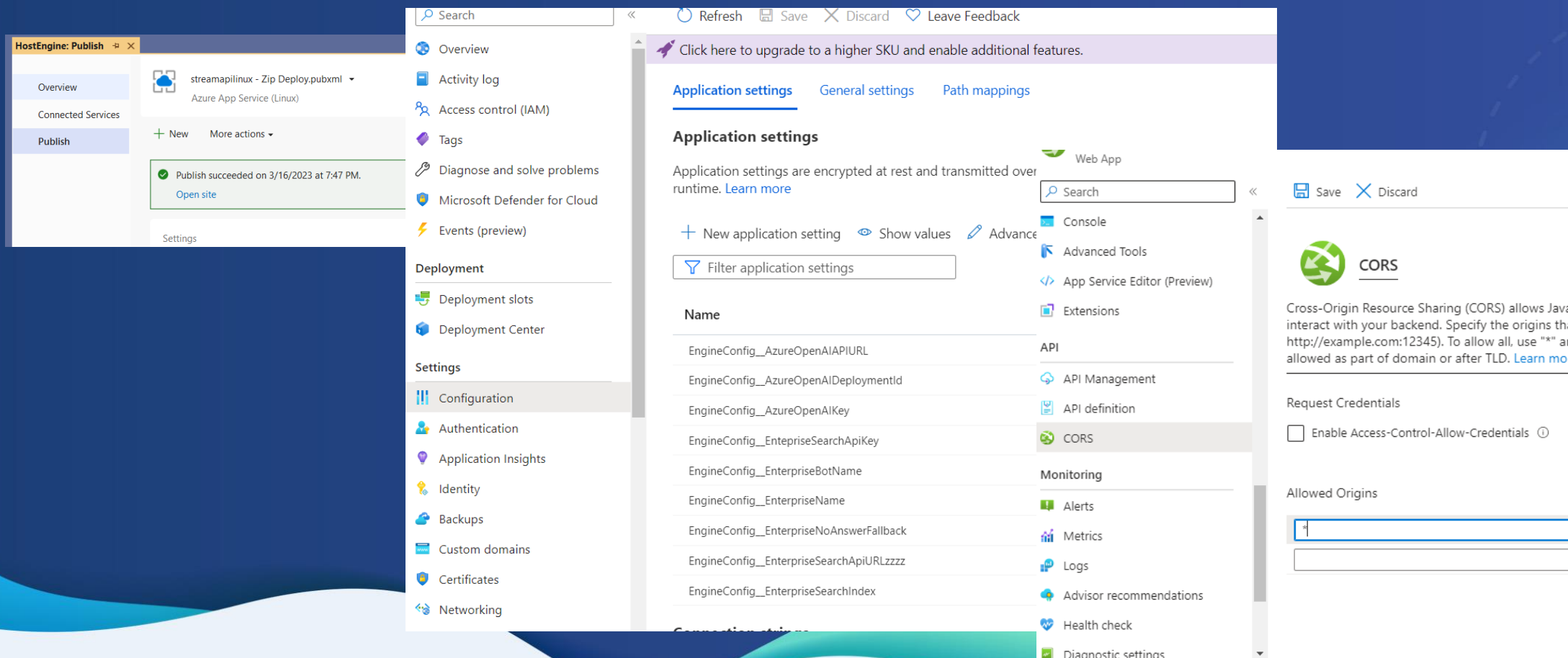
RedisConnectionString

CosmosbDBConnectionString

CosmosDBName

```
"EngineConfig": {  
  "AzureOpenAIAPIURL": "",  
  "AzureOpenAIKey": "",  
  "AzureOpenAIDeploymentId": "",  
  "RedisConnectionString": "",  
  "CosmosDBConnectionString": "",  
  "CosmosDBName": "completionsDB",  
}
```

## 部署与配置: 后端服务 (API)



The screenshot displays the Azure Portal interface for configuring a Web App. The left sidebar shows the 'HostEngine: Publish' tab with a success message: 'Publish succeeded on 3/16/2023 at 7:47 PM. Open site'. The main area shows the 'Application settings' tab for a Web App. The 'Application settings' list includes 'EngineConfig\_AzureOpenAIURL' and 'EngineConfig\_AzureOpenAIDeploymentId'. The 'CORS' settings are expanded, showing the 'Allowed Origins' field.

**HostEngine: Publish**

- Overview
- Connected Services
- Publish

streamapilinux - Zip Deploy.pubxml  
Azure App Service (Linux)

+ New More actions

Publish succeeded on 3/16/2023 at 7:47 PM.  
Open site

Settings

**Deployment**

- Deployment slots
- Deployment Center

**Settings**

- Configuration
- Authentication
- Application Insights
- Identity
- Backups
- Custom domains
- Certificates
- Networking

**Application settings**

Application settings are encrypted at rest and transmitted over runtime. [Learn more](#)

Click here to upgrade to a higher SKU and enable additional features.

Application settings General settings Path mappings

Web App

Search

+ New application setting Show values Advance

Filter application settings

Name	Value
EngineConfig_AzureOpenAIURL	API
EngineConfig_AzureOpenAIDeploymentId	
EngineConfig_AzureOpenAIKey	
EngineConfig_EnterpriseSearchApiKey	
EngineConfig_EnterpriseBotName	
EngineConfig_EnterpriseName	
EngineConfig_EnterpriseNoAnswerFallback	
EngineConfig_EnterpriseSearchApiURLzzzz	
EngineConfig_EnterpriseSearchIndex	

**Monitoring**

- Alerts
- Metrics
- Logs
- Advisor recommendations
- Health check
- Dagnostic settings

**CORS**

Cross-Origin Resource Sharing (CORS) allows JavaScript to interact with your backend. Specify the origins that are allowed to interact with your backend. To allow all, use "\*" and allowed as part of domain or after TLD. [Learn more](#)

Request Credentials

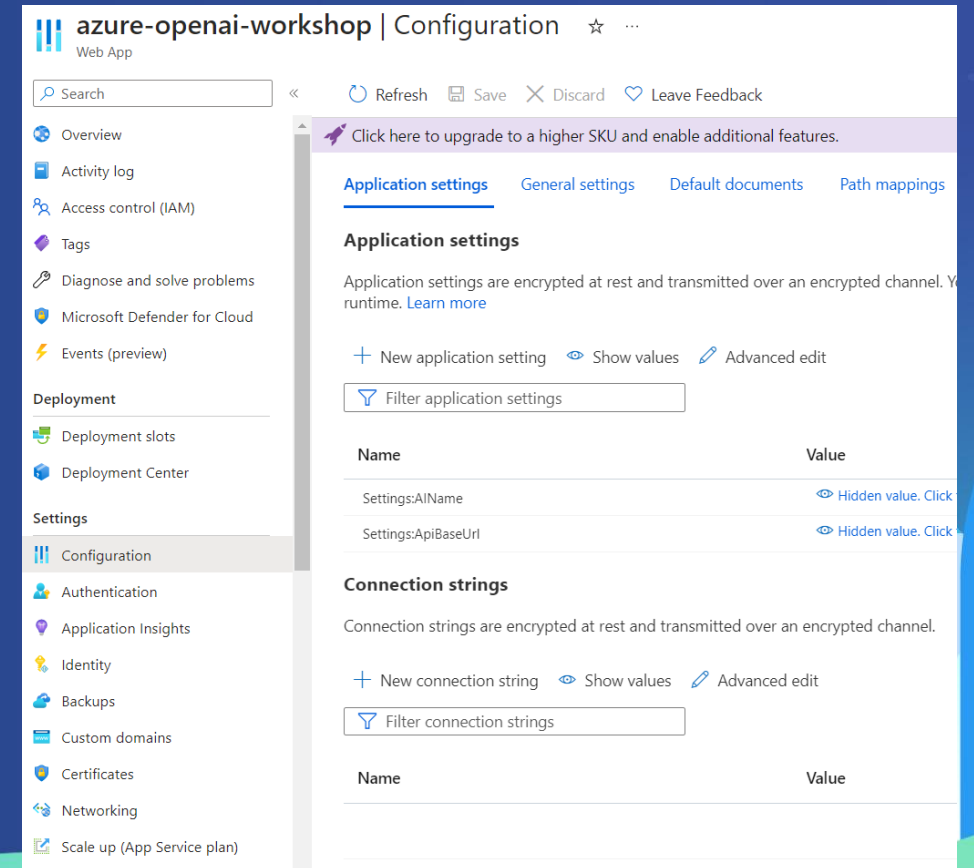
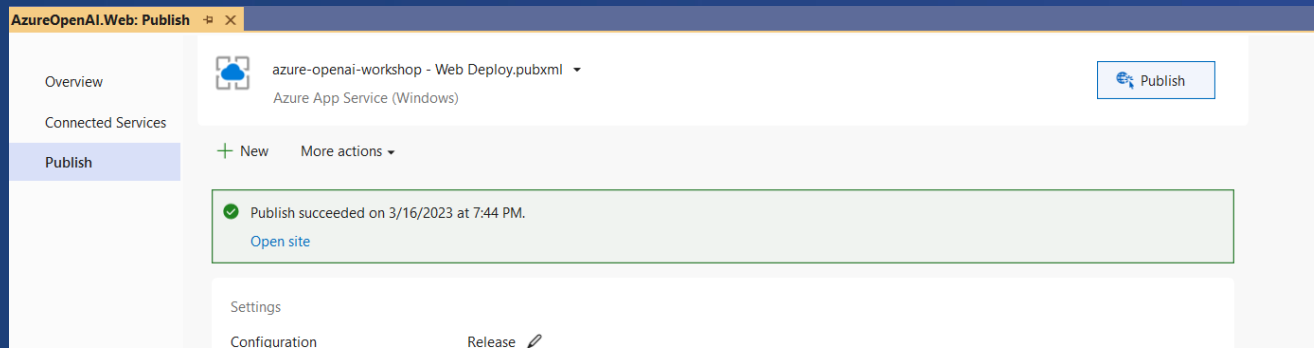
☐ Enable Access-Control-Allow-Credentials ⓘ

Allowed Origins

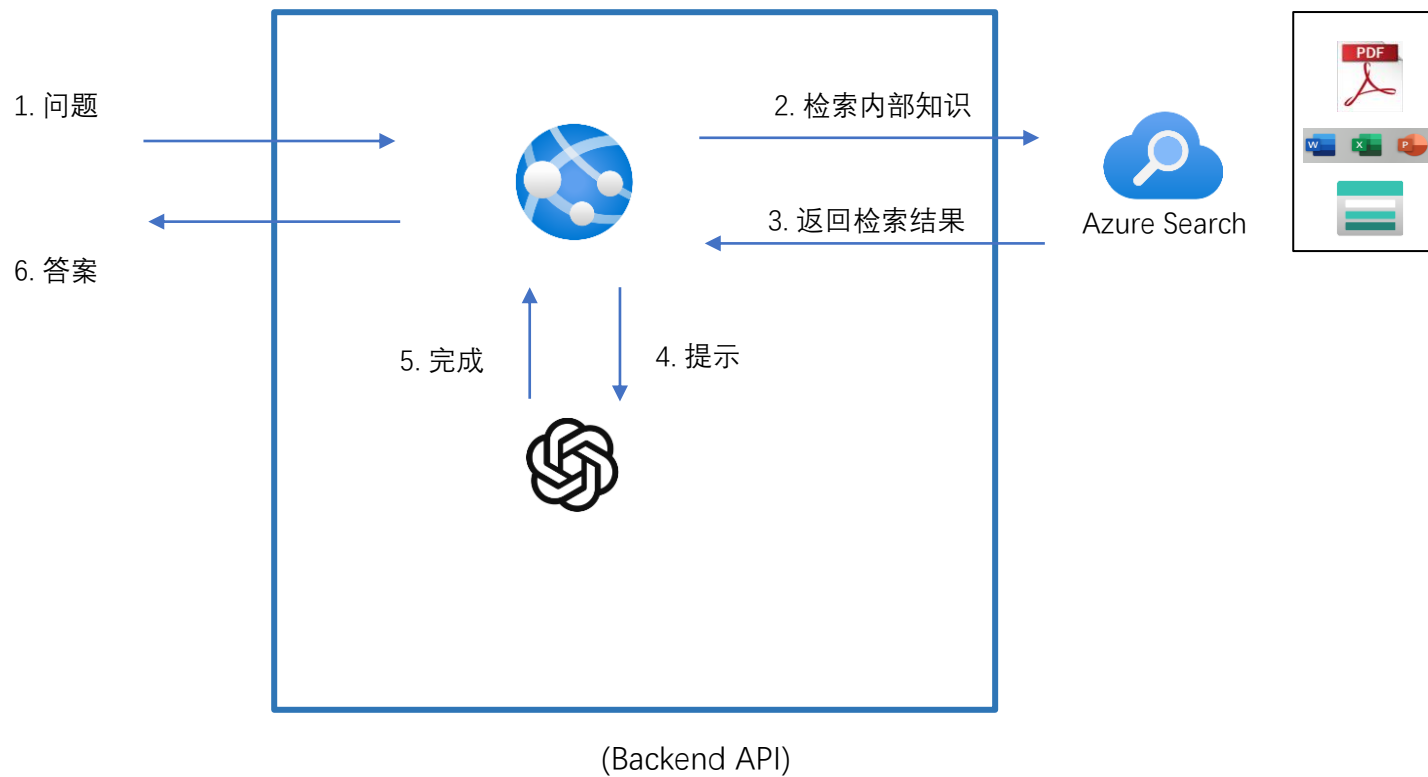
+



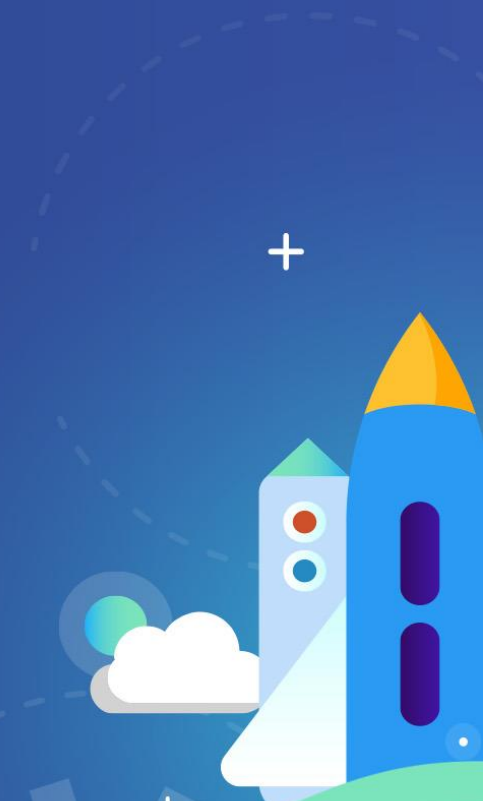
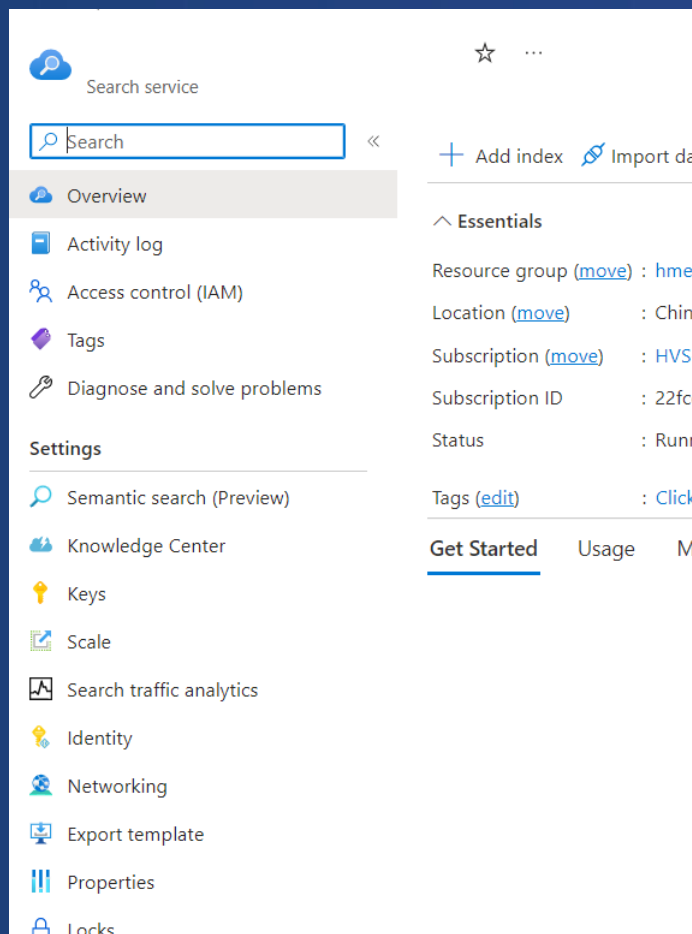
## 部署与配置: 前端应用 (UI)



## ChatGPT + 企业知识库/数据



## 创建Azure Search 服务



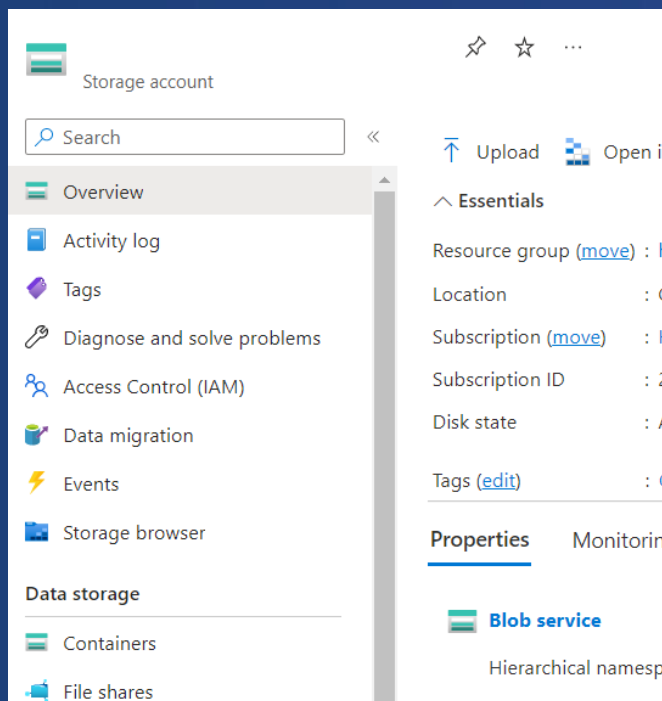
## 创建Azure Search 索引

```
var indexClient = new SearchIndexClient(new Uri(searchServiceEndPoint), new AzureKeyCredential(searchServiceKey));
var indexerClient = new SearchIndexerClient(new Uri(searchServiceEndPoint), new AzureKeyCredential(searchServiceKey));

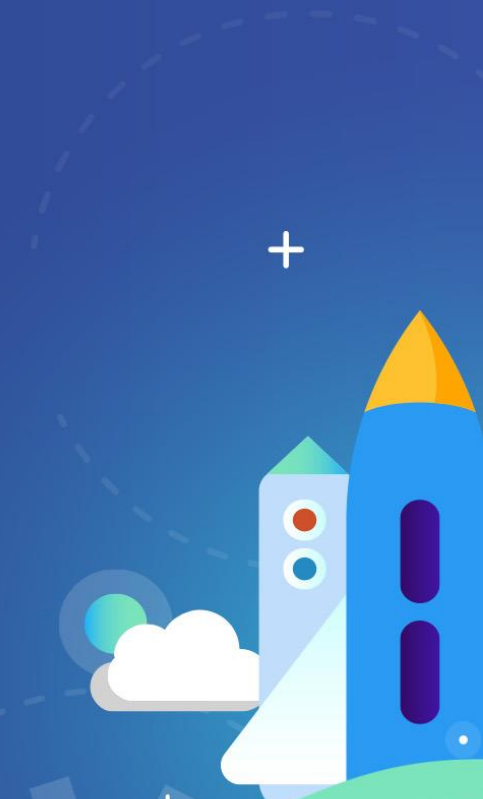
var searchFields = new List<SearchField>
{
    new SimpleField("Id", SearchFieldType.String) { IsKey = true, IsFilterable = true, IsSortable = true },
    new SearchableField("Name") { IsFilterable = true, IsSortable = true },
    new SearchableField("Content"), // large content don't enable filterable, sortable, faceting
};
var index = new SearchIndex(searchIndexName, searchFields);
await indexClient.CreateOrUpdateIndexAsync(index);

var docDataSource = new SearchIndexerDataSourceConnection(
    assetIndexDataSourceName,
    SearchIndexerDataSourceType.AzureBlob,
    assetBlobConnectionString,
    new SearchIndexerDataContainer(assetBlobContainerName)
);
await indexerClient.CreateOrUpdateDataSourceConnectionAsync(docDataSource);
var docIndexerParameters = new IndexingParameters();
docIndexerParameters.IndexingParametersConfiguration = new IndexingParametersConfiguration();
docIndexerParameters.IndexingParametersConfiguration.IndexedFileNameExtensions = ".pdf, .docx, .doc, .docm, .pptx, .ppt, .pptm";
docIndexerParameters.IndexingParametersConfiguration.DataToExtract = BlobIndexerDataToExtract.ContentAndMetadata;
var docIndexer = new SearchIndexer(assetIndexerName, docDataSource.Name, index.Name)
{
    Parameters = docIndexerParameters,
    Schedule = new IndexingSchedule(TimeSpan.FromDays(1)),
    FieldMappings =
    {
        new FieldMapping("Id") { TargetFieldName = "Id" },
        new FieldMapping("Name") { TargetFieldName = "Name", MappingFunction = new FieldMappingFunction("urlDecode") },
        new FieldMapping("content") { TargetFieldName = "Content" }
    }
};
await indexerClient.CreateOrUpdateIndexerAsync(docIndexer);
```

## 创建 Storage Account (用于存储企业知识库文件)



1. 创建存储账户
2. 在Blob服务中创建容器 “workshop”



## 上传样例文档 (企业知识库文件)

ENGINE CONTROL

CONTENT-TYPE

application/pdf

CONTENT-MD5

L+SViHGU5Mz2DbryqkByO...

CONTENT-ENCODING

CONTENT-LANGUAGE

CONTENT-DISPOSITION

LEASE STATUS

Unlocked

LEASE STATE

Available

LEASE DURATION

-

COPY STATUS

-

COPY COMPLETION TIME

-

Undelete

Metadata

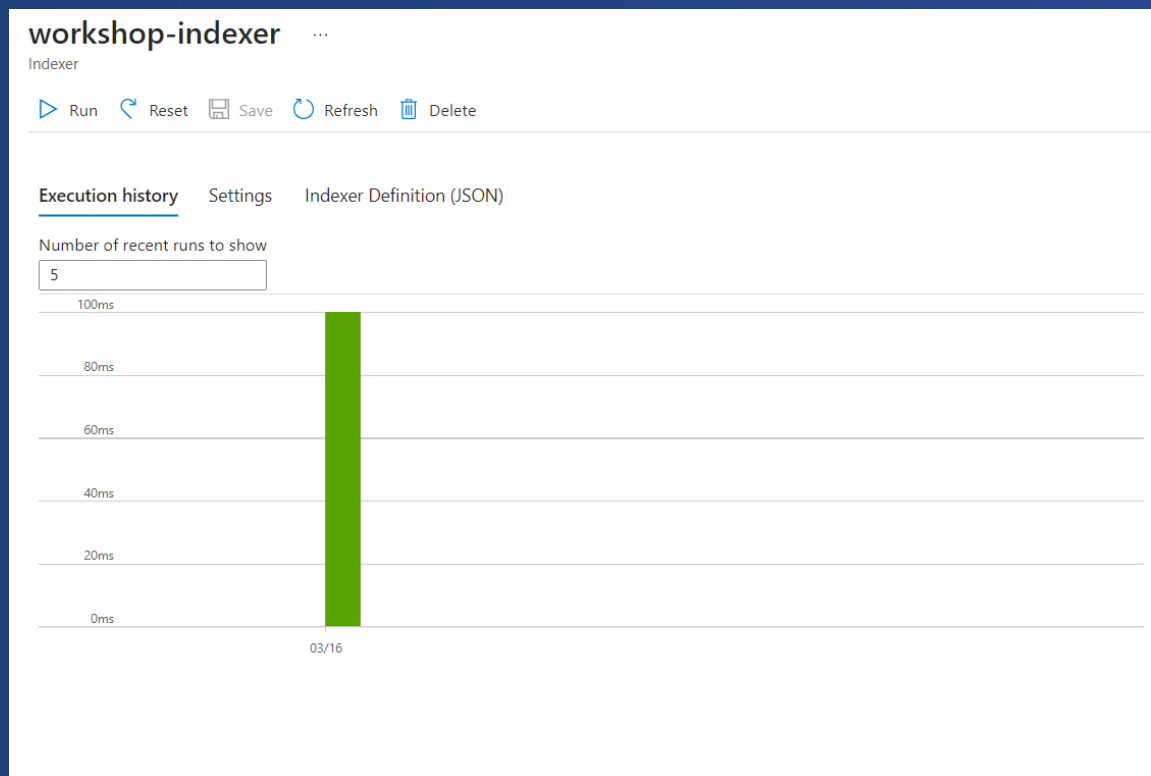
Key	Value
Id	doc1
Name	产品介绍

Blob index tags

1. 上传文档
2. 编辑文档metada:  
Id  
Name



## 运行索引器





## 调试索引

workshop-index ...

Save Discard Refresh Create Demo App Edit JSON Delete

Documents 0 Storage 0 Bytes

Search explorer Fields CORS Scoring profiles

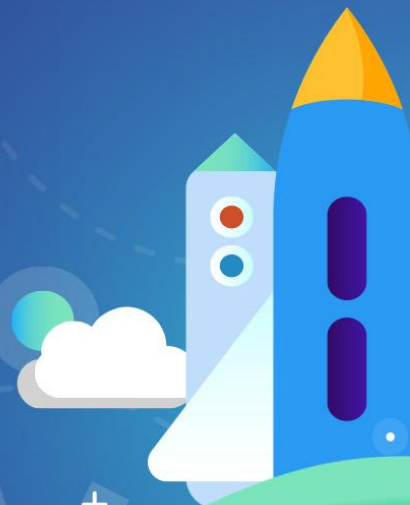
Query string 产品 API version 2021-04-30-Preview Search

Request URL fexes/workshop-index/docs?api-version=2021-04-30-Preview&search=%E4%B7%A7%E5%93%81

Results

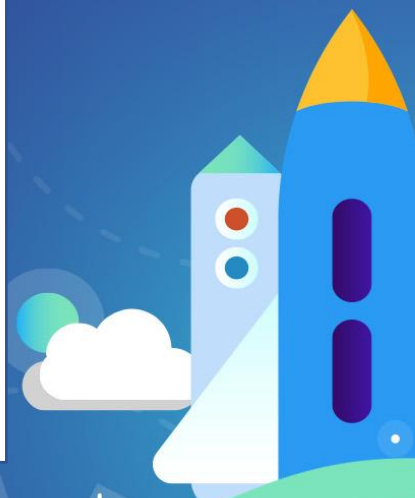
```
1 {
2   "@odata.context": "https://hmedia-search-test12.search.azure.cn/indexes('workshop-index')/$metadata#docs(*)",
3   "value": [
4     {
5       "@search.score": 1.6726563,
6       "Id": "doc1",
7       "Name": "产品介绍",
8       "Content": "\nbopoda\n产品介绍\n\nbopoda 博普达\n\n\nbopoda 博普达\n\n产品家族\n\nwww.bopoda.cn\n2\n\nVideo Hub\n视频中
```

参考：  
<https://learn.microsoft.com/en-us/azure/search/search-explorer>



## ChatGPT + 企业知识库/数据

```
var internalData = await SearchEnterpriseData(prompt);
var internalResult = internalData?.Content ?? "";
var options = new CompletionsOptions
{
    MaxTokens = MaxTokens,
    Prompt = { BuildPropmtGPT3(prompt, internalResult) }
};
var completions = await _client.GetCompletionsAsync(_config.DeploymentId, options);
var completion = completions.Value.Choices[0].Text;
return completion;
```



## 代码解析: 关于配置信息

必须配置如下信息:

EnterpriseBotName

EnterpriseName

EnterpriseNoAnswerFallback

EnterpriseSearchApiURL

EnterpriseSearchApiKey

EnterpriseSearchIndex

```
"EnterpriseBotName": "",  
"EnterpriseName": "",  
"EnterpriseNoAnswerFallback": "",  
"EnterpriseSearchApiURL": "",  
"EnterpriseSearchApiKey": "",  
"EnterpriseSearchIndex": ""  
}
```

# Thanks

