

Index

[1. Key Vault 3](#_Toc171787880)

[2. Azure Functions 3](#_Toc171787881)

[3. API Management Service 8](#_Toc171787882)

[4. Azure Data Factory 9](#_Toc171787883)

[5. Logic App 14](#_Toc171787884)

[6. Azure Machine Learning Workspace 16](#_Toc171787885)

[7. Azure Open AI 16](#_Toc171787886)

[Configurating Open AI in Azure 16](#_Toc171787887)

[Deploying the model 17](#_Toc171787888)

[8. Power BI: 19](#_Toc171787889)

[Importing data from Dataverse 19](#_Toc171787890)

[Importing a Date Table 19](#_Toc171787891)

[Loading in Post Code data 22](#_Toc171787892)

[Connecting to the weather API 23](#_Toc171787893)

[Calling OpenAI from Power Query 25](#_Toc171787894)

# Key Vault

Use Key Vault to securely store your secrets

* Create 4 new secrets in Key Vault
  + AppId – The service principal Application ID
  + SecretId – The service principal secret value
  + InstanceUri – The url for the dataverse environment

# Azure Functions

Use Azure Functions to extract, clean and standardise data

* Create a new project using Visual Studio and select Azure Functions as the project template

A screenshot of a computer

Description automatically generated

* Provide a name and use the default values

A screenshot of a computer

Description automatically generated

* Install the **Microsoft.PowerPlatform.Dataverse.Client** NuGet
* Using the code available in GitHub you can create 3 functions.
  + **AnonymiseData**
    - **Description:**
      * Connects to a dataverse table and anonymise the selected columns using a hash value
    - **Parameters:** 
      * TableName
      * Column Name (can use a list of columns separated by a comma)
  + **ValidatePostCode**
    - **Description:**
      * Connects to a dataverse table and validates if a post code follows the correct UK format. If invalid, sets the value to blank
    - **Parameters:** 
      * TableName
      * ColumnName (can use a list of columns separated by a comma)
  + **ExtractWeatherData**
    - **Description:**
      * Calls the Open Meteo API and returns the results in a tabular format
    - **Parameters:** 
      * latitude
      * longitude
      * start\_date
      * end\_date
      * daily
      * wind\_speed\_unit
* Publish the project to the Azure Function. Right-click on the project name, select publish and then follow the wizzard

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* Once published, navigate to the Azure Function in the Azure Portal and add the following Environment variables. Replace the key vault name accordingly. If properly configured, a green tick will be added
  + AppId – @Microsoft.KeyVault(SecretUri=https://{keyvaultname}.vault.azure.net/secrets/AppId)
  + SecretId – @Microsoft.KeyVault(SecretUri=https://{keyvaultname}.vault.azure.net/secrets/SecretId)
  + InstanceUri – @Microsoft.KeyVault(SecretUri=https:// {keyvaultname}.vault.azure.net/secrets/InstanceUri)

A screenshot of a computer

Description automatically generated

# API Management Service

Use the API Management Service to manage the APIs and provide a customised Url without sharing authentication keys

* Open the API Management Service, navigate to APIs and select Function App
* Using the Browse button, select the Azure Function instance recently deployed and select the 3 functions
* Fill the respective details and select Create

A screenshot of a computer

Description automatically generated

* Select the recently create API and navigate to Settings. Untick the box Subscription required. **Be aware that by doing that, the API is accessible by anyone with access to the Url. You can read about how to Protect the API** [**here**](https://learn.microsoft.com/en-us/azure/api-management/transform-api)**.** **This action should just done for the hackathon**

# Azure Data Factory

Use Azure Data Factory to extract data from Dataverse and the Weather API and stored it in the Azure Blob Storage as parquet

* Create 4 linked Services. Update the values accordingly

**Azure Blob Storage**

A screenshot of a computer

Description automatically generated

**Azure Key Vault**

A screenshot of a computer

Description automatically generated

**Dataverse**

A screenshot of a computer

Description automatically generated

**Rest Service**

A screenshot of a computer

Description automatically generated

Create 3 datasets

* + **RestResource**
    - **Parameters**
      * RelativeUrl

A screenshot of a computer

Description automatically generated

* + **DataverseEntity**
    - **Parameters**
      * EntityName

A screenshot of a computer

Description automatically generated

* + **Parquet**
    - **Parameters**
      * TargetFolder

A screenshot of a computer

Description automatically generated

* Create 2 pipelines
  + **Copy Dataverse to Storage**
    - **Parameters**
      * EntityName
      * TargetFolder

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Extract Weather Data**

* + - **Parameters**
      * TargetFolder
      * latitude
      * longitude
      * start\_date
      * end\_date
      * daily
      * wind\_speed\_unit
    - **Variables**
      * RelativeUrl

A screenshot of a computer

Description automatically generated

**Value:** ExtractWeatherData?latitude=@{pipeline().parameters.latitude}&longitude=@{pipeline().parameters.longitude}&start\_date=@{pipeline().parameters.start\_date}&end\_date=@{pipeline().parameters.end\_date}&daily=@{pipeline().parameters.daily}&wind\_speed\_unit=@{pipeline().parameters.wind\_speed\_unit}

A screenshot of a computer

Description automatically generated

**Value:** @replace(variables('RelativeUrl'),' ','%20')

A screenshot of a computer

Description automatically generated

# Logic App

Use logic app to trigger a new data workflow

* Add the **Microsoft Dataverse > Add a new row (legacy)** action and configure it by selecting the table containing pertinent data
* Add the HTTP > HTTP action and configure it to validate the post code

A screenshot of a computer

Description automatically generated

* Add the HTTP > HTTP action and configure it to anonymise the data

A screenshot of a computer

Description automatically generated

* Add the Azure Data Factory > Create a pipeline run action and configure it to extract the dataverse data
* Add the Azure Data Factory > Create a pipeline run action and configure it to extract the weather data

A screenshot of a computer

Description automatically generated

# Azure Machine Learning Workspace

Use the Azure Machine Learning workspace to create a new time-series forecasting model

* Navigate to Data > Datastores and create a new data store by connecting to the Azure Blob Storage
* Navigate to Notebooks and upload the following notebooks, available in the GitHub repository
  + **SplitDataForTraining** – Splits the data for training, validation and testing, using the 60-20-20 rule
  + **CategoriseData** – Demonstrates how to use Open AI to analyse a descriptive field and create new categories
* After splitting the data, follow this [tutorial](https://learn.microsoft.com/en-us/azure/machine-learning/tutorial-automated-ml-forecast?view=azureml-api-2) to create a new forecasting model . Be aware the interface changed and the order of some steps will need to be adjusted accordingly

# Azure Open AI

## Configurating Open AI in Azure

Navigate to Open AI in Azure

If you haven’t already created a dedicated Open AI Resource Group, please create a new one:

Name: rg-lewisham-openai

Instance Details:

Region: UK South

Name: Lewisham-openai-hackathon

A screenshot of a computer

Description automatically generatedPricing Tier: Standard

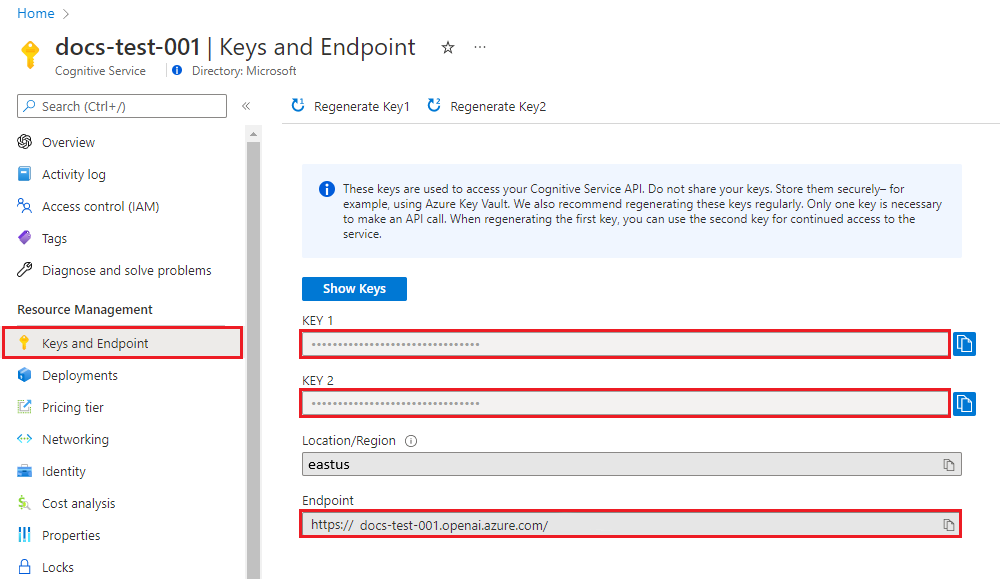
To successfully make a call against Azure OpenAI, you need an endpoint and a key.

ENDPOINT 🡪 This value can be found in the Keys & Endpoint section when examining your resource from the Azure portal. Alternatively, you can find the value in the Azure OpenAI Studio > Playground > Code View. An

example endpoint is: https://docs-test-001.openai.azure.com/.

API-KEY 🡪 This value can be found in the Keys & Endpoint section when examining your resource from the Azure portal. You can use either KEY1 or KEY2.

Go to your resource in the Azure portal. The Keys & Endpoint section can be found in the Resource Management section. Copy your endpoint and access key as you'll need both for authenticating your API calls. You can use either KEY1 or KEY2. Always having two keys allows you to securely rotate and regenerate keys without causing a service disruption.



## Deploying the model

Still inside of the Azure Open AI services in the Azure Portal click on Model Deployments -> Manage Deployment:

A screenshot of a computer

Description automatically generated

Create a new deployment:

Select a model: gpt-4

Model Version: 1106-Preview

Deployment type: Standard

Deployment name: OpenAI-Model-Hackathon-GPT4

Content filter: Default

A screenshot of a computer

Description automatically generated

# Power BI:

## Importing data from Dataverse

You need to connect to Dataverse and retrieve the required tables:

A screenshot of a computer

Description automatically generatedIn the home tab under data select Dataverse:

A screenshot of a computer

Description automatically generatedThen login and search for the required table(s). Once you have selected the desired table(s) click Transform Data:

We will then review this data and apply some renaming and cleansing steps if required.

## Importing a Date Table

A date table is needed to perform historic analysis

A screenshot of a computer

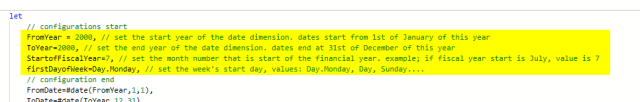
Description automatically generatedCreate a new blank query in Power BI:

A screenshot of a computer

Description automatically generatedIn the Power Query Editor window, go to View tab, and click on Advanced Editor

A screenshot of a computer program

Description automatically generatedCopy and Paste the *Date\_Dimension\_Script* here (replace the existing text in the Advanced Editor window:

You need to configure the Date table based on your need, the first few lines are the configurations that you can set based on your need and then press done;

A screenshot of a computer

Description automatically generatedNext rename the table to Date Table and press Close & Apply:

You will notice it has automatically created a date hierarchy which we can use later for drill downs.

You then want to create a join between your sample data and the date table be creating a Many to One join on the date columns:

A screenshot of a computer

Description automatically generated

## Loading in Post Code data

Download the London Postcode file and save it to your local storage on your pc.

Under Get data select CSV:

A screenshot of a computer

Description automatically generated

Then transform the data and apply the same name changes and cleansing steps if required. Rename it to London Postcode table and load it into the report.

Next you will need to create a join between the sample data and the postcode by creating a many to one join from your sample data to the Post Code table on the Post Code columns:

A screenshot of a computer

Description automatically generated

## Connecting to the weather API

This section provides an alternative method to call the weather API directly in Power BI

You will load in the Weather API in a similar method to the Date table.

A screenshot of a computer

Description automatically generatedCreate a new blank query in Power BI:

A screenshot of a computer

Description automatically generatedIn the Power Query Editor window, go to View tab, and click on Advanced Editor

Copy and Paste the *Script-for-Weather-API* here (replace the existing text in the Advanced Editor window.

The *Weather\_API\_Script* calls the weather API directly and applies some transformative steps to get it into the final format

Rename the Table to Weather Table and then press Close and Apply.

You then want to create a join between your sample data and the weather table by creating a Many to One join on the date columns:

A screenshot of a computer

Description automatically generated

A list of the weather parameters can be found below which you may need to refer back to when creating the visuals:

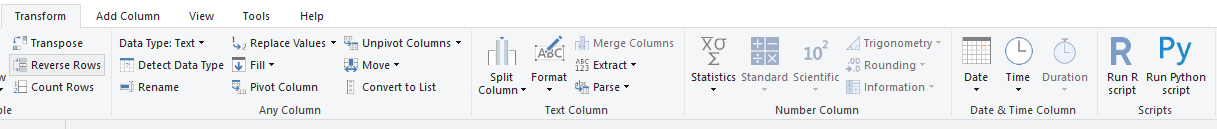
|  |  |  |
| --- | --- | --- |
| **Variable** | **Unit** | **Description** |
| weather\_code | WMO code | The most severe weather condition on a given day |
| temperature\_2m\_max | °C (°F) | Maximum and minimum daily air temperature at 2 meters above ground |
| temperature\_2m\_min | |  |
| apparent\_temperature\_max | °C (°F) | Maximum and minimum daily apparent temperature |
| apparent\_temperature\_min | | |
| precipitation\_sum | mm | Sum of daily precipitation (including rain, showers and snowfall) |
| rain\_sum | mm | Sum of daily rain |
| snowfall\_sum | cm | Sum of daily snowfall |
| precipitation\_hours | hours | The number of hours with rain |
| sunrise | iso8601 | Sun rise and set times |
| sunset |  |  |
| sunshine\_duration | seconds | The number of seconds of sunshine per day is determined by calculating direct normalized irradiance exceeding 120 W/m², following the WMO definition. Sunshine duration will consistently be less than daylight duration due to dawn and dusk. |
| daylight\_duration | seconds | Number of seconds of daylight per day |
| wind\_speed\_10m\_max | km/h (mph, m/s, knots) | Maximum wind speed and gusts on a day |
| wind\_gusts\_10m\_max | |  |
| wind\_direction\_10m\_dominant | ° | Dominant wind direction |
| shortwave\_radiation\_sum | MJ/m² | The sum of solar radiaion on a given day in Megajoules |
| et0\_fao\_evapotranspiration | mm | Daily sum of ET₀ Reference Evapotranspiration of a well watered grass field |

## Calling OpenAI from Power Query

First, install Python if you have not done so already. Please visit the official website (https://www.python.org/downloads/) to download it.

Once Python has been installed, enable Python scripting in Power BI. To do so, open Power BI Desktop. Then, click **File** > **Options and settings** and go to **Options** > **Python** **scripting**. It should automatically detect the python file location, then click **OK**.

Next press Transform data to open Power Query, navigate to the desired data set and in the Transform Tab select Run Python Script:



Paste in the *AOAI\_PBI\_Script.* You will need to modify this script to enter in your own Azure Open AI Endpoint and Key configures earlier. You will also need to change it point it at your free text field Column. You can also modify the prompt to best serve your needs for the desired outcome.

An example of what the input and output looks like can be found below:

