Eagle Eye Application Note - AN071

Using Power BI for Data Visualization of Video Search Metadata

2024-07-26 Revision 1.0

Target Audience

This document is intended to be used by an end user's data analyst with an understanding of and experience using Python and Power BI who wishes to extract business insights from video analytics metadata. It's ideal for customers in retail, but the concepts also apply across industries.

Introduction

Organizations have multiple sources of data such as POS terminals, environmental sensors, enterprise resource planning (ERP) systems, etc. Organizations may already be using a data visualization tool such as Power BI or similar to perform data analysis. They can get valuable insights by combining this with metadata extracted from video surveillance cameras. The Eagle Eye Cloud VMS can convert nearly any industry standard ONVIF-compliant camera into an AI camera by leveraging AI capabilities in the Cloud. The metadata extracted from AI is available via Eagle Eye's open APIs and can be viewed and manipulated with data visualization tools.

Suggested Process

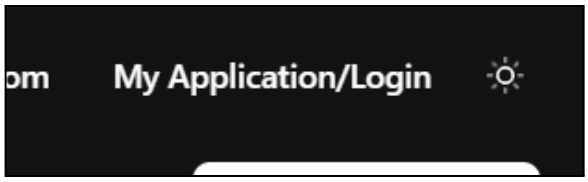
The suggested process outlined below describes how a Python script can be used to pull data from the Eagle Eye Networks API and inject it into Power BI.

1. Create an app in Eagle Eye Networks developer portal. Get an application ClientID and client secret. Obtain account access token and base URL.
2. Using the Python code provided, make the edits specified below. Once complete, run the script. The script will create a JSON text file in the local directory with metadata from Eagle Eye Network's API.
3. Import the JSON file into Power BI to be used for data visualization.

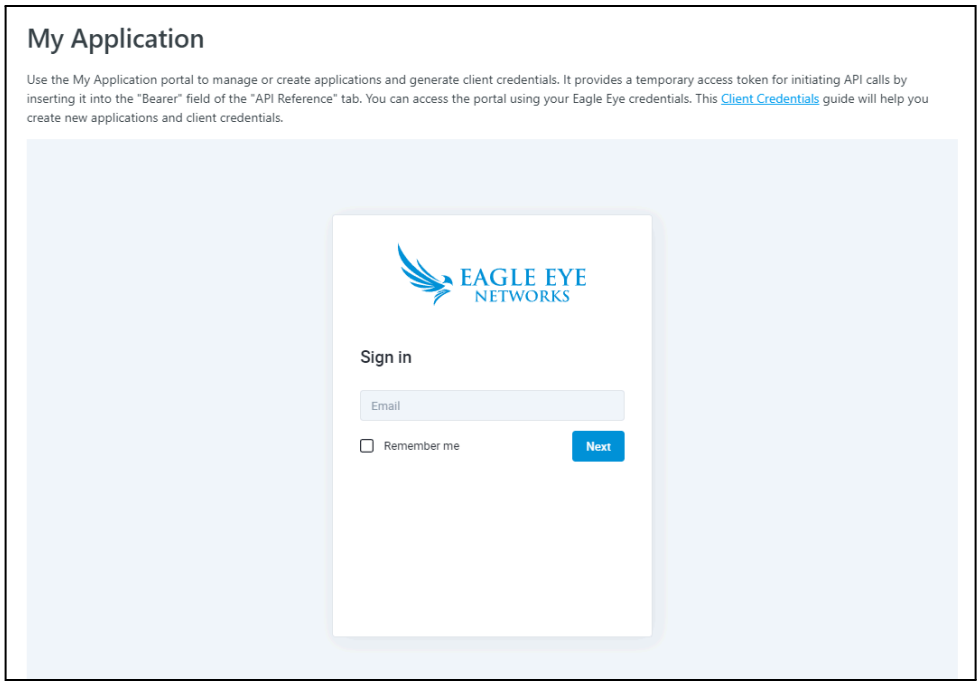
Create an App

Navigate to the Eagle Eye Networks developer portal using the following URL: https://developer.eagleeyenetworks.com/

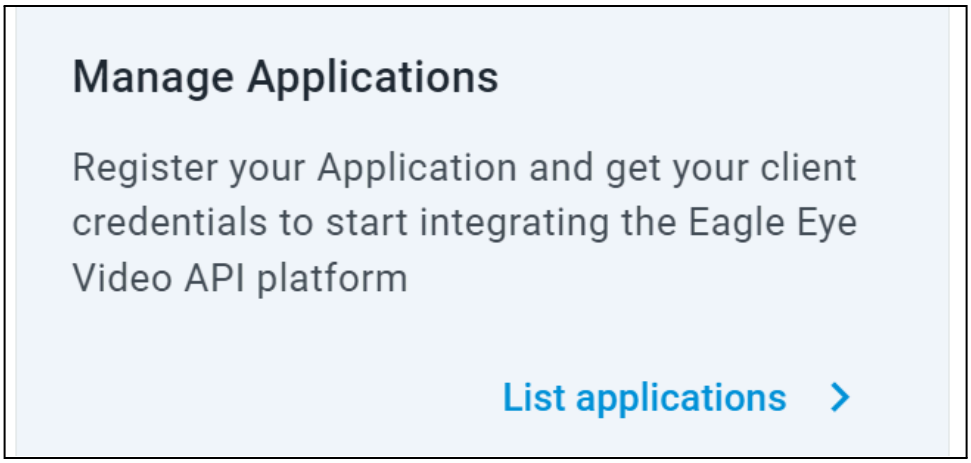
- Select “My Application/Login” in the top right-hand corner.



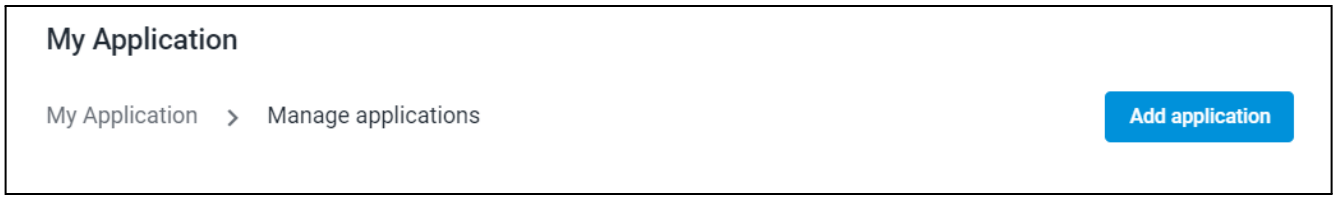
- Log in to the account using your Eagle Eye Cloud VMS credentials.



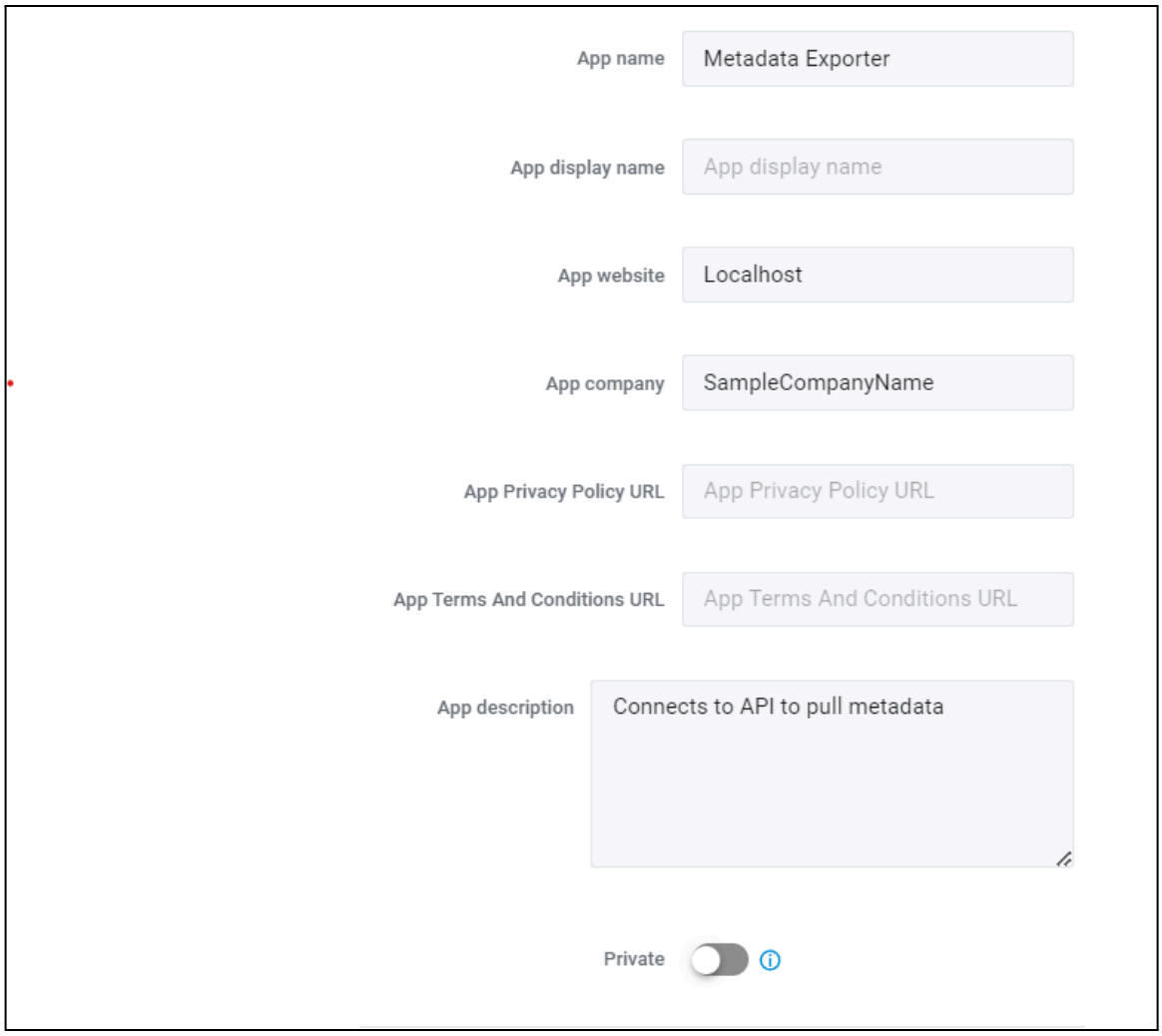
- Select “List applications” on the Manage Applications tile.

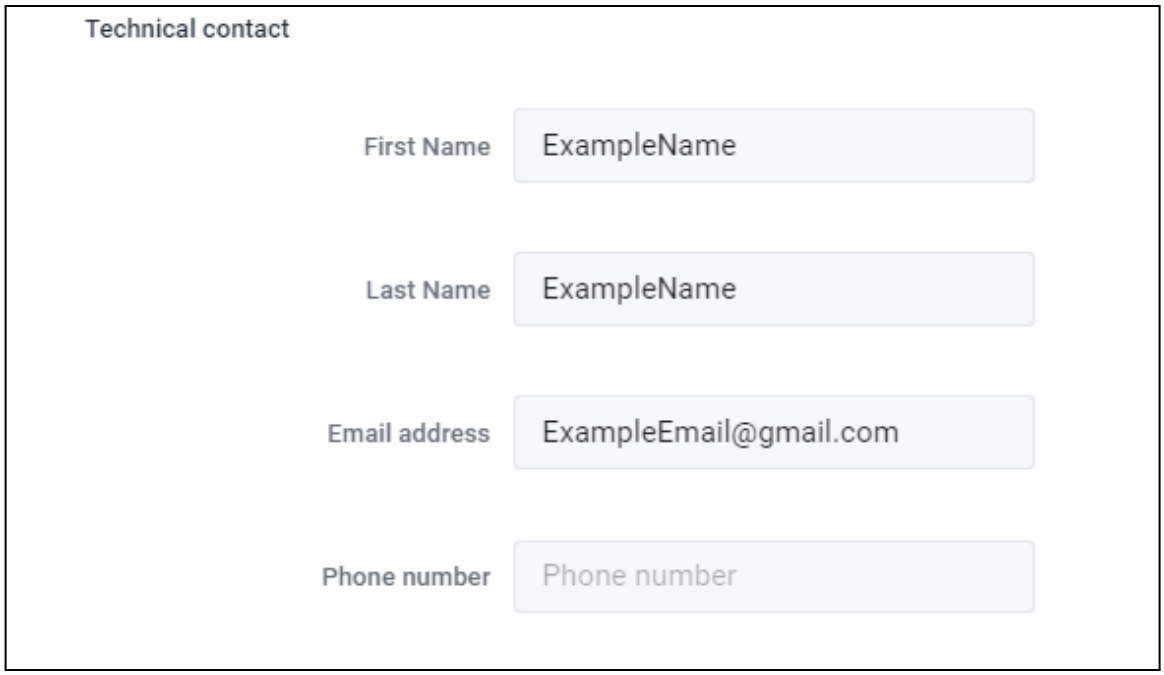


- Select “Add Application.”

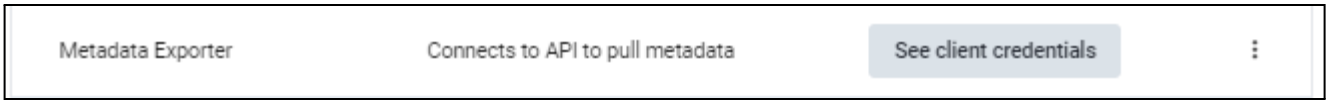


- Complete the required fields and select “Save Changes.”

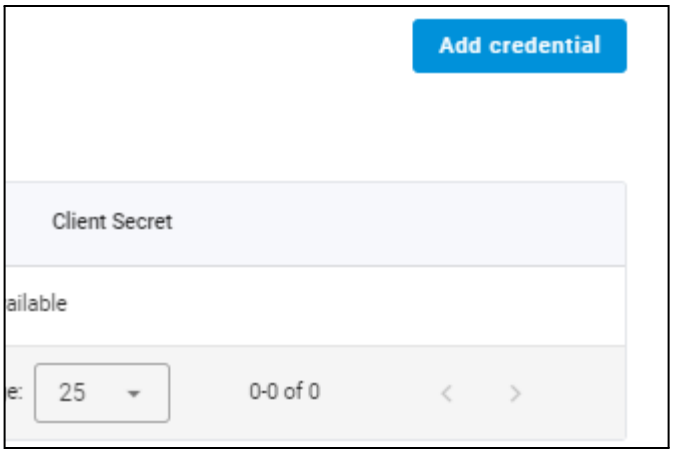




- The application will now show in the application list. Select “See client credentials.”

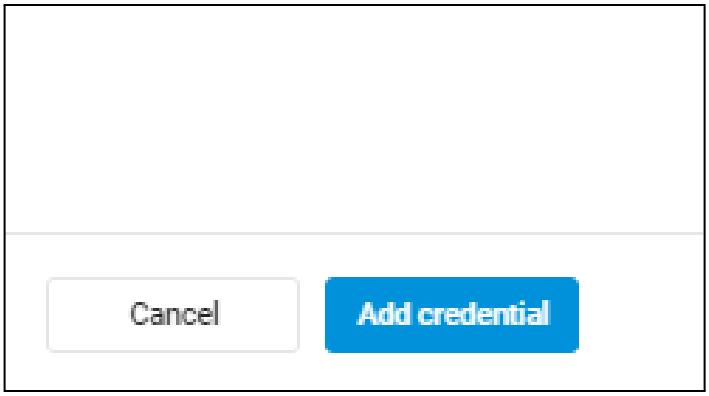


- Select “Add credential.”



- Complete the required fields as shown below.

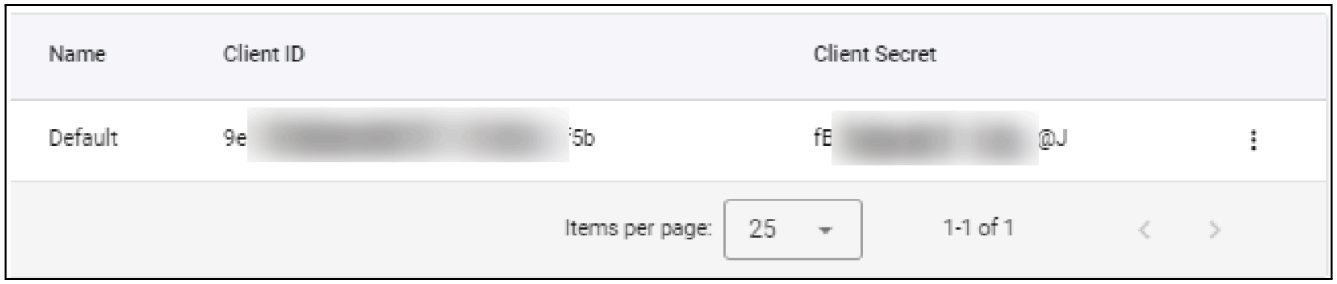
- Select “Add credential.”



Client ID and Client Secret

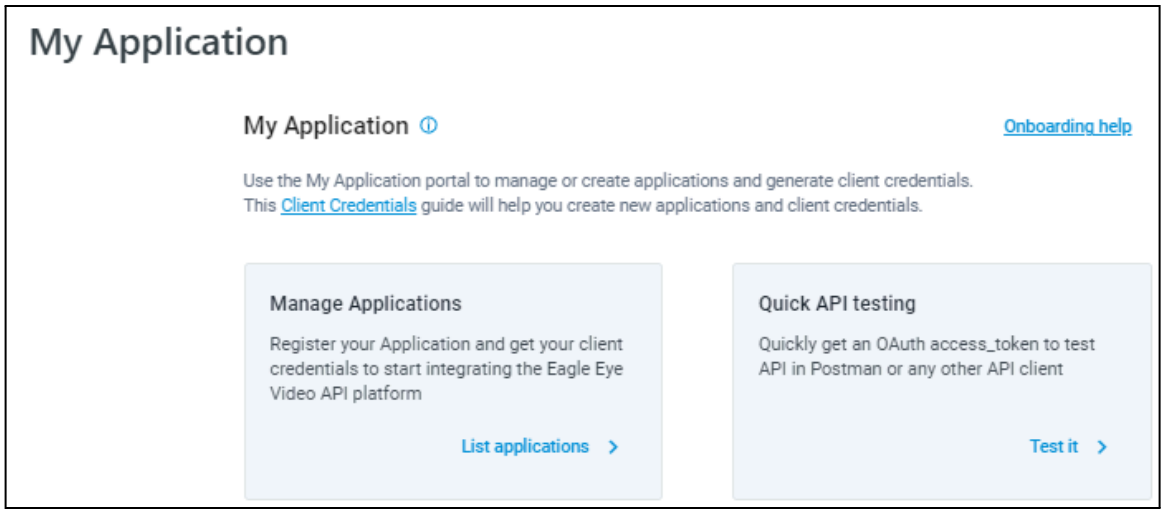
Now that an application has been created, the Client ID and client secret can be viewed.

Record your Client ID and Client Secret. Please note these are specific to your application.

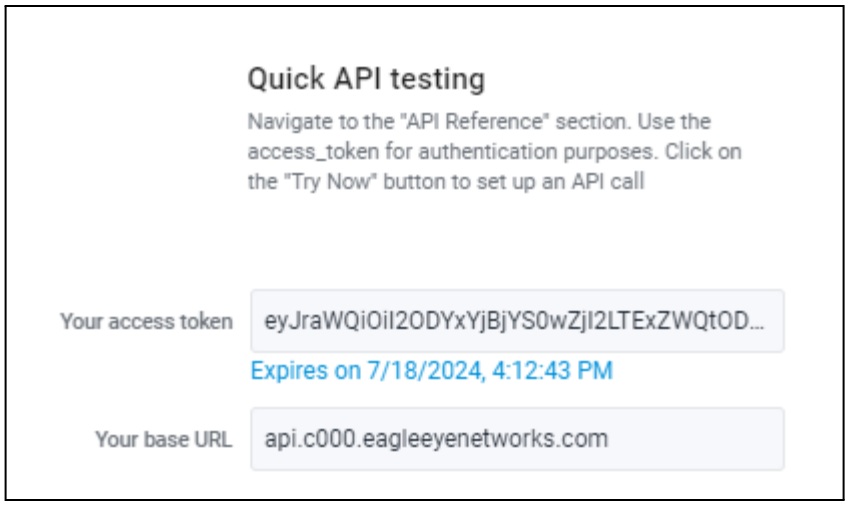


Access Token and Base URL

- Navigate back to the “My Application" Login page and select the “Test it” button on the Quick API testing tile.



- Record the contents of both “Your access token” and “Your base URL.”



Python Script

Use the script below with your Client ID, client secret, access token, and base URL to pull metadata from Eagle Eye Networks' API.

```
import json, requests, csv
from flask import Flask, request

##Start the webserver
hostName          = "127.0.0.1"
port              = 3333

##Set Client Credentials
#UNIQUE TO ACCOUNT
clientId          = "Your Client ID"
#UNIQUE TO ACCOUNT
clientSecret      = "Your client secret"


def getTokens(code):
   url =
"https://auth.eagleeyenetworks.com/oauth2/token?grant_type=authorization_code&scope=vms.all
&code="+code+"&redirect_uri=http://"+hostName + ":" + str(port)
   response = requests.post(url, auth=(clientId, clientSecret))
   return response.text

app = Flask(__name__)

@app.route('/')
def index():
   code = request.args.get('code')

   if (code):
      oauthObject = getTokens(code)
      print(oauthObject);
      return "You are logged in"
   else:
      endpoint          = "https://auth.eagleeyenetworks.com/oauth2/authorize"
      requestAuthUrl     =
endpoint+"?client_id="+clientId+"&response_type=code&scope=vms.all&redirect_uri=http://"+hostN
ame + ":" + str(port)
```

```
        return "<a href='"+requestAuthUrl+"'>Login with Eagle Eye Networks</a>"

    if __name__ == '__main__':
        app.run(host=hostName, port=port)


##Call to API to perform parse on native-language search
url = "https://api.c000.eagleeyenetworks.com/api/v3.0/videoAnalyticEvents:parse"
#UNIQUE SEARCH
payload = { "query": "male in red shirt" }
#AUTHORIZATION CHANGES TO CURRENT TOKEN
headers = {
    "accept": "application/json",
    "content-type": "application/json",
    "authorization": "Bearer Your access token"
}

parseresponse = requests.post(url, json=payload, headers=headers)

#TIME STAMP, UNIQUE SEARCH
url =
"https://api.c000.eagleeyenetworks.com/api/v3.0/videoAnalyticEvents:deepSearch?timestamp__gt
e=2024-07-01T01%3A00%3A00.000%2B00%3A00&timestamp__lte=2024-07-05T01%3A00%3A00.0
00%2B00%3A00&actor__in=&roiName__in=&pageSize=100&sort=-timestamp"

payload = json.loads(parseresponse.text)
#AUTHORIZATION CHANGES TO CURRENT TOKEN
headers = {
    "accept": "application/json",
    "content-type": "application/json",
    "authorization": "Bearer Your access token"
}

response = requests.post(url, json=payload, headers=headers)

metadata = json.loads(response.text)




print(response.text)


data_file = open('responsedata.json', 'w', newline='')
```

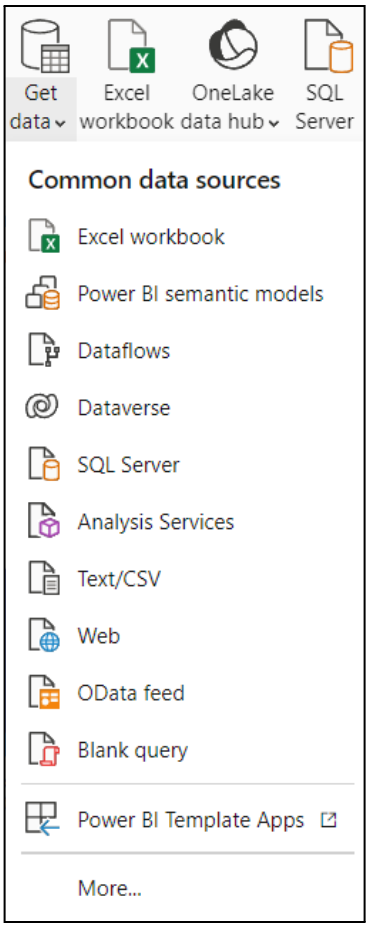
```
json.dump(metadata['results'], data_file)
data_file.close()
```

This script will create a JSON text file in your local directory titled “responsedata.json.”

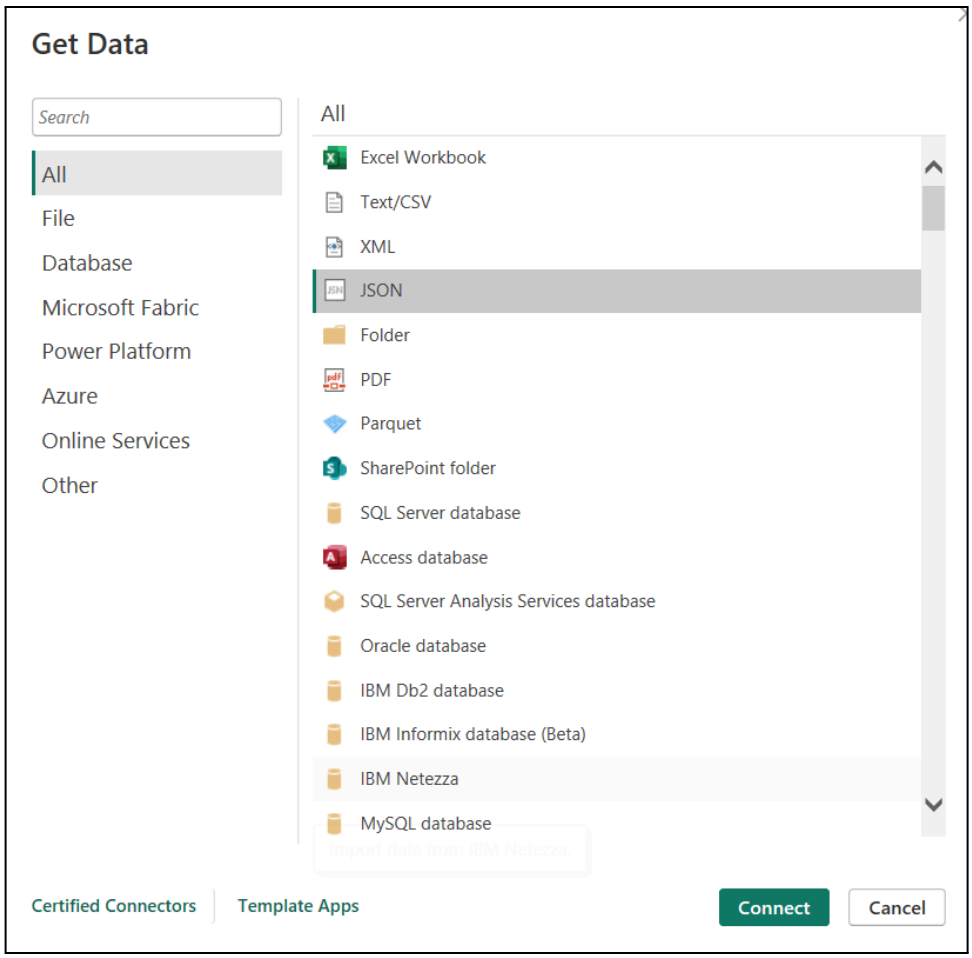
Using Power BI

Using Power BI desktop, import the data from the “Get Data” tab.

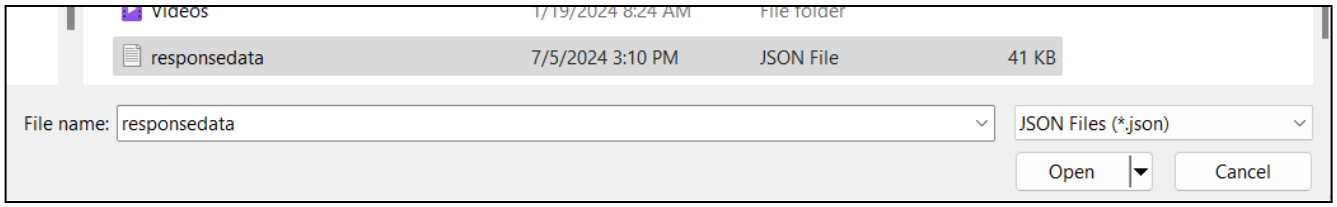
- Select “More…” from the drop-down menu.



- Select JSON, then click the Connect button.



- Navigate to the directory where the file titled “responsedata.json” was saved, and select Open.



The data is now loaded into Power BI, where it can be transformed and manipulated for data visualization.