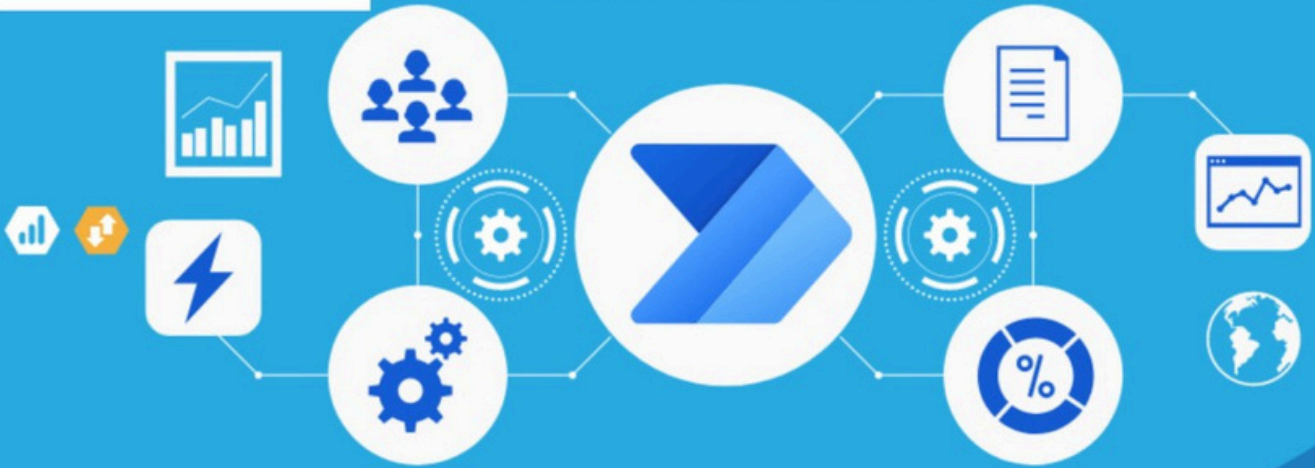




Power Automate

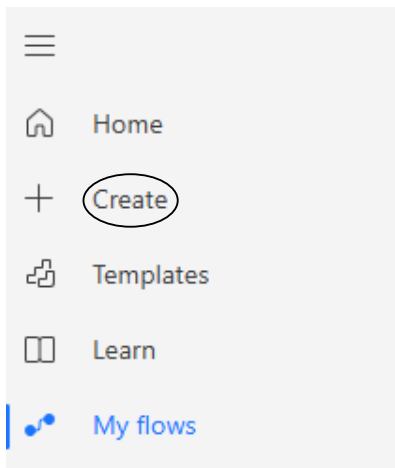


HOW TO GET DATA FROM MICROSOFT FORMS TO MICROSOFT AZURE STORAGE

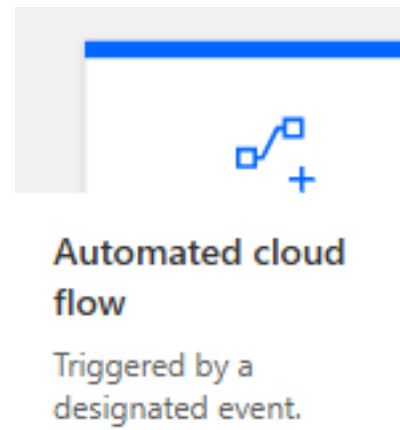
Is possible to integrate Microsoft Forms and Microsoft Azure Storage through Microsoft Power Automate flows. That way, we can collect generated data from answers given by users and use these data to develop indicators about the data collected. You will need one Microsoft Power Automate premium license to use some components.

So, let's get started!

1 - CREATING THE MICROSOFT POWER AUTOMATE FLOW

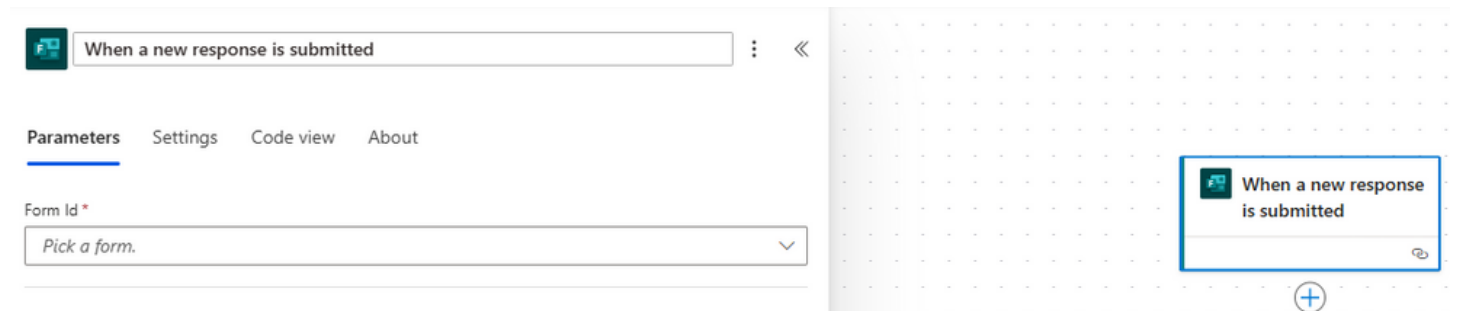


1.1 - Click on “Create”.

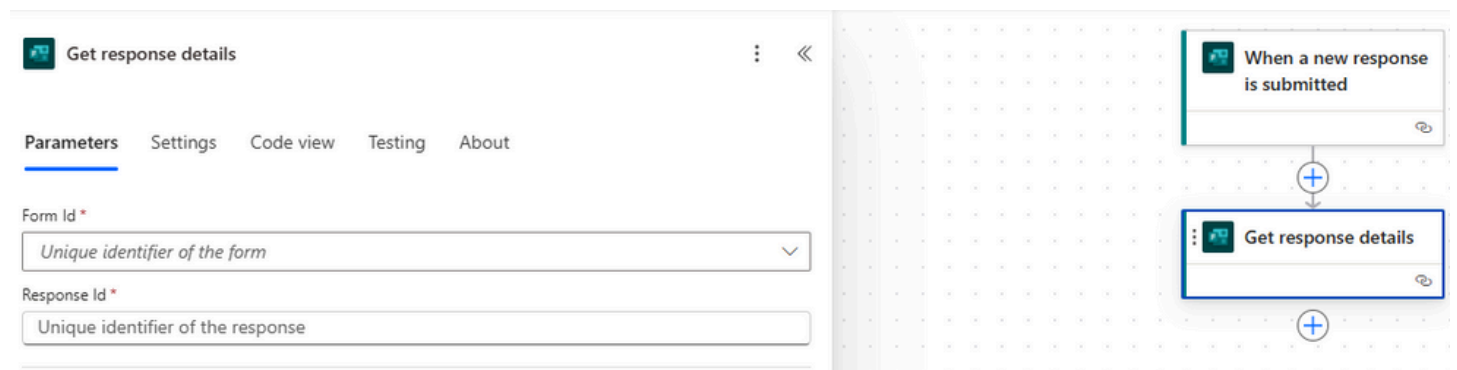


1.2 - Select the “Automated Cloud Flow”.

1.3 - Search for “When a new response is submitted” component and pick your form. Your form must to be created using the same user as the Microsoft Power Automate user.



1.4 - Add “Get response details” component.



1.5 - Create a blob storage connection using the same user as the one used by Microsoft Forms and Microsoft Power Automate connection. Put the .csv extension in the blob name box to save as a .csv file.

Create blob (V2)

Parameters Settings Code view Testing About

Storage account name or blob endpoint *
Azure Storage account name or blob endpoint.

Folder path *
Specify folder path to upload.

Blob name *
Specify name of the blob to create.

Blob content *
Specify the content of the blob to upload.

Advanced parameters
Showing 0 of 1 Show all Clear all

Flow diagram steps:
1. When a new response is submitted
2. Get response details
3. Create blob (V2)

1.6 - Create a Microsoft Azure Data Factory pipeline execution using the same user as the one used until now. This pipeline can execute one Databricks Data Platform notebook with a code to read the .csv file saved in Microsoft Azure Storage and save it in a Databricks Data Platform Delta Table. Select the advanced parameters option to set a variable that will be used in Microsoft Azure Data Factory pipeline execution.

Create a pipeline run

Parameters Settings Code view Testing About

Subscription *
The unique identifier for the Microsoft Azure subscription. The subscription ID forms p...

Resource Group *
Resource group name.

Data Factory Name *
The name of the Data Factory.

Data Factory Pipeline Name *
The name of the Data Factory pipeline.

Advanced parameters
parameters Show all Clear all

☐ Reference pipelineRunId.
☒ parameters

Flow diagram steps:
1. When a new response is submitted
2. Get response details
3. Create blob (V2)
4. Create a pipeline run

1.7 - Variable set up in Microsoft Power Automate and in Microsoft Azure Data Factory.

Advanced parameters

Showing 1 of 2


Show all


Clear all


Parameters


```
{
  "fileParam":  body/Name
}
```


X

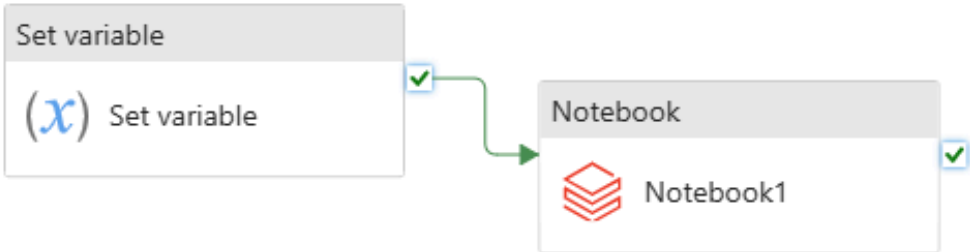
 Save

 Save as template

 Validate

 Debug

 Add trigger






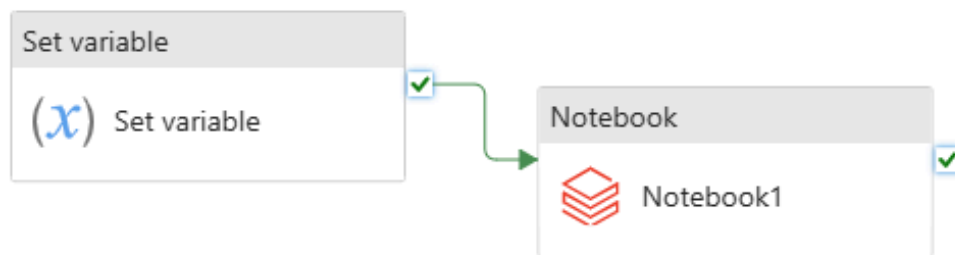
Parameters Variables Settings Output

+ New

 Delete

<input type="checkbox"/>	Name	Type	Default value	
<input type="checkbox"/>	<div>formParam</div>	<div>String</div>	<div>Value</div>	

 Save  Save as template  Validate  Debug  Add trigger

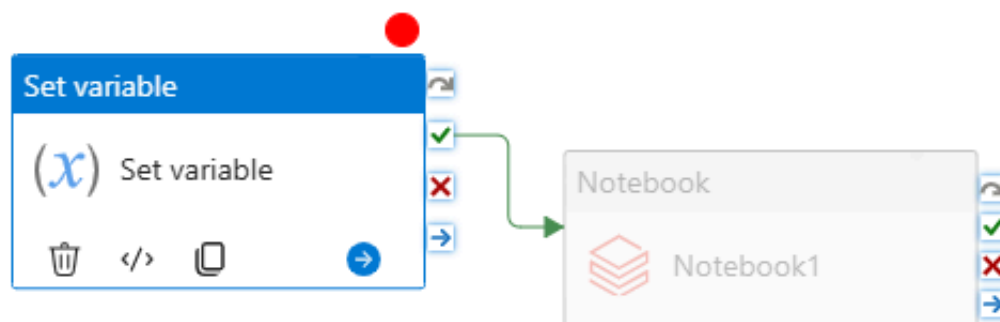


Parameters **Variables** Settings Output

 New  Delete

<input type="checkbox"/>	Name	Type	Default value
<input type="checkbox"/>	id_answer	String	Value

 Save  Save as template  Validate  Debug  Add trigger



General **Settings** User properties

Variable type ^①

☒ Pipeline variable ☐ Pipeline return value

Name *

id_answer

 New

Value

@pipeline().parameters.formParam

Save Save as template Validate Debug Add trigger

Set variable

(x) Set variable

Notebook



Notebook1



General Azure Databricks¹ Settings¹ User properties

Notebook path *

Browse

Open

Base parameters

+ New

Delete

☐

Name *

Value

☐

formParam

@pipeline().parameters.formParam



```
max_file = dbutils.widgets.get("fileParam")
file = 'Your Microsoft Azure Storage Dir' + max_file + '.csv'
df_forms = spark.read.csv(file)
df_forms = df_forms.withColumnRenamed("_c0", "Your Column Name") \
                    .withColumnRenamed("_c1", "Your Column Name") \
                    .withColumnRenamed("_c2", "Your Column Name") \
                    .withColumnRenamed("_c3", "Your Column Name")
df_forms.createOrReplaceTempView('view_forms_' + max_file[0:-4])
spark.sql(
    f"""INSERT INTO "Your Table Name"
        SELECT
            Your Column Name,
            Your Column Name,
            Your Column Name,
            Your Column Name
        FROM view_forms_{max_file[0:-4]}""")
```

Python