Lab 14. Application Lifecycle Management Light

Author: Serge Luca aka "Doctor Flow"

Updated by: Dattatray-Patil

Learning objective: a good practice in programming is to be able to reuse code. Creating reusable and generic flow is a great way to reuse code and to make your code more robust. Child flow are part of the standard Power Platform license and does not require a premium license.

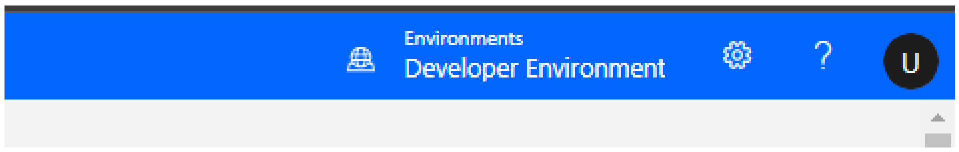
Duration: 30 minutes

Scenario: we will create a flow that will retrieve information from a SharePoint list and from an excel file. In order to make this flow easier to be deployed across several environment we will dynamically provide the list and file url. We will create a child flow that will retrieve this information from environment variables. Child flow must be created from a Solution, so you will create a solution as well.

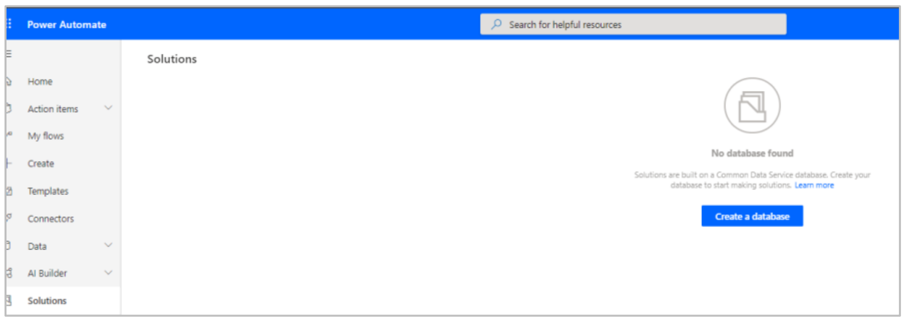
PART 1. Define the parameters

Tasks:

1. Make sur you have a premium license; this will be necessary to retrieve Environment variable from a flow 'see lab0).
2. Switch to your Developer environment



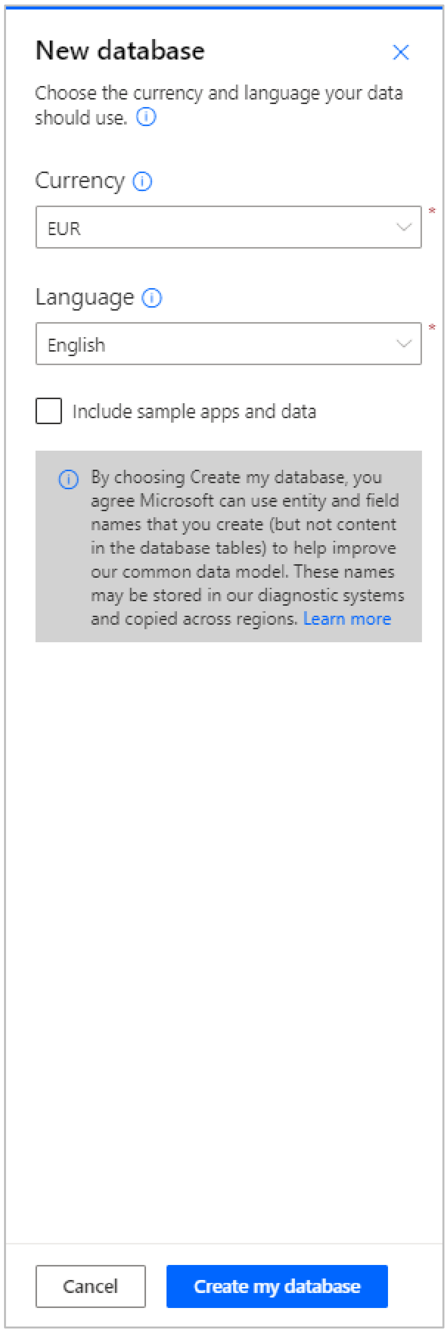
3. Go to Solutions and create a new solution from scratch; if requested click **Create a database**.



No database found

Solutions are built on a Common Data Service database. Create your database to start making solutions. Learn more

4. Fill in the database properties and click **create my database**:


New database

Choose the currency and language your data should use.

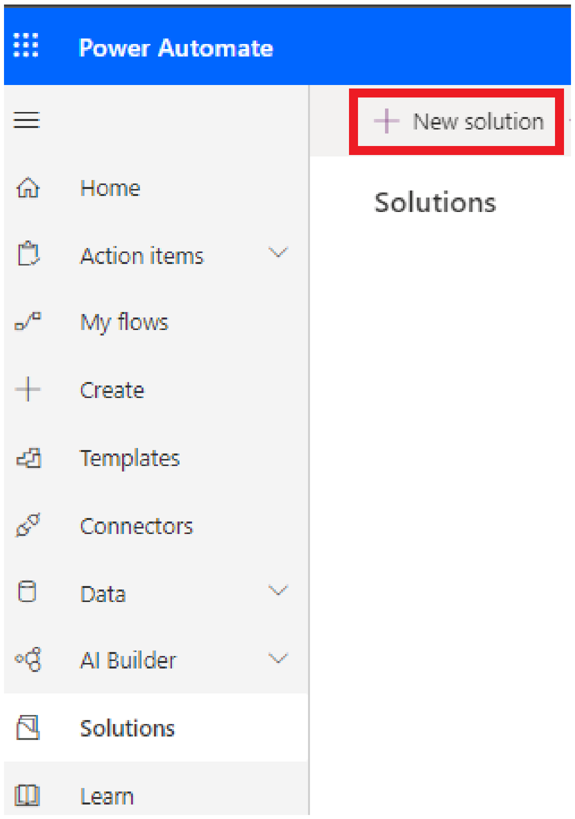
Currency: EUR

Language: English

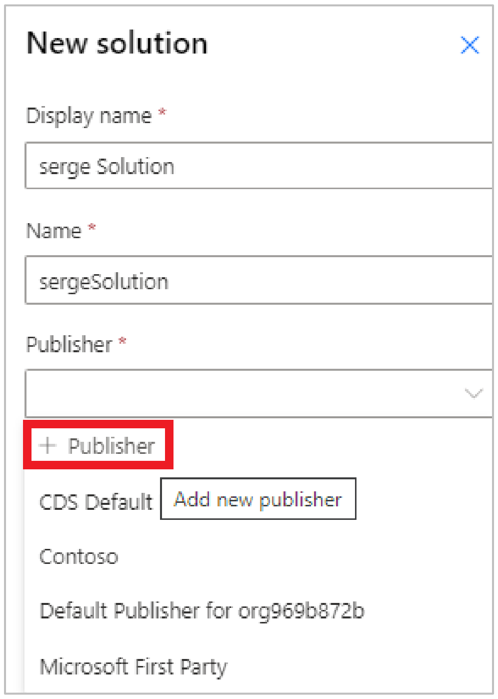
☐ Include sample apps and data

By choosing Create my database, you agree Microsoft can use entity and field names that you create (but not content in the database tables) to help improve our common data model. These names may be stored in our diagnostic systems and copied across regions. Learn more

Wait a couple of minutes for the database to be created.
Click New Solution



Fill-in the solution properties and under Publisher click + Publisher:


New solution

Display name *

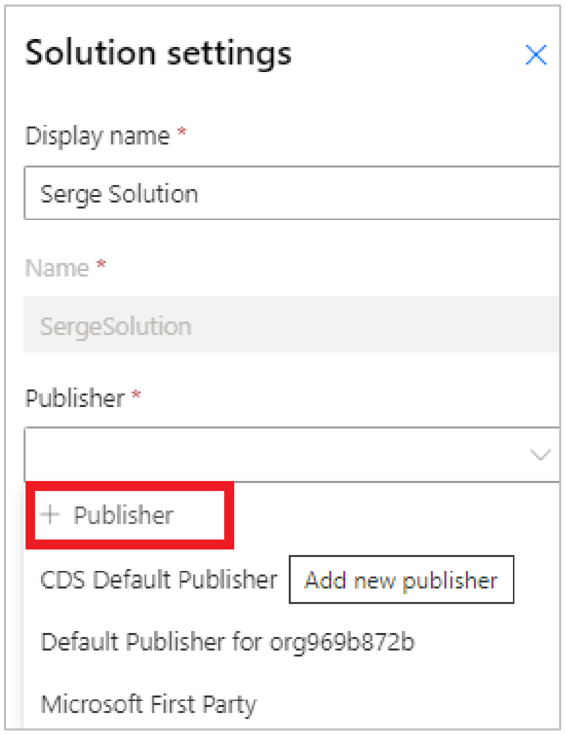
serge Solution

Name *

sergeSolution

Publisher *

In the next windows, replace the existing publisher with a new one by clicking on **+ Publisher**:


Solution settings

Display name *

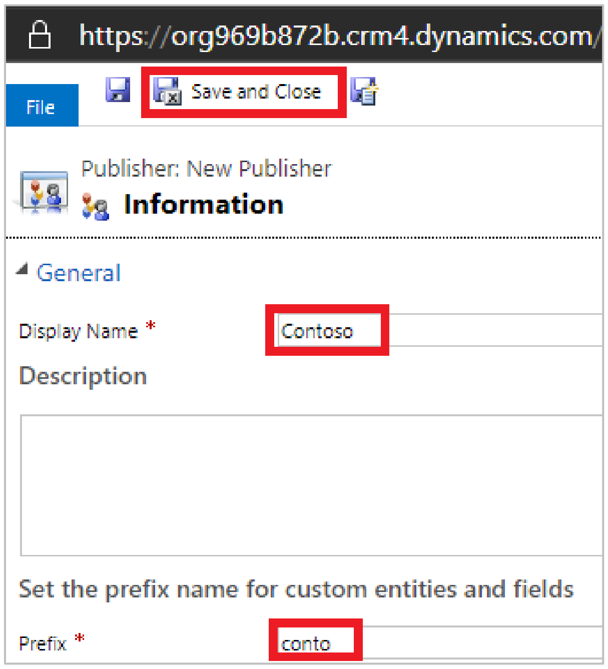
Serge Solution

Name *

SergeSolution

Publisher *

1. A new window will pop up where you can select a new Display Name like **Contoso** and a prefix like **conto**:



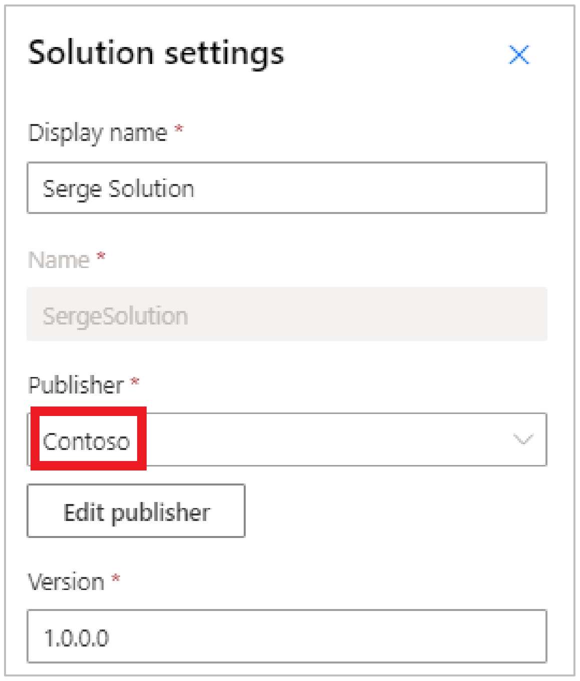
Display Name * Contoso

Description

Set the prefix name for custom entities and fields

Prefix * conto

Select **Contoso** as the new publisher and click Update:



Solution settings

Display name *: Serge Solution

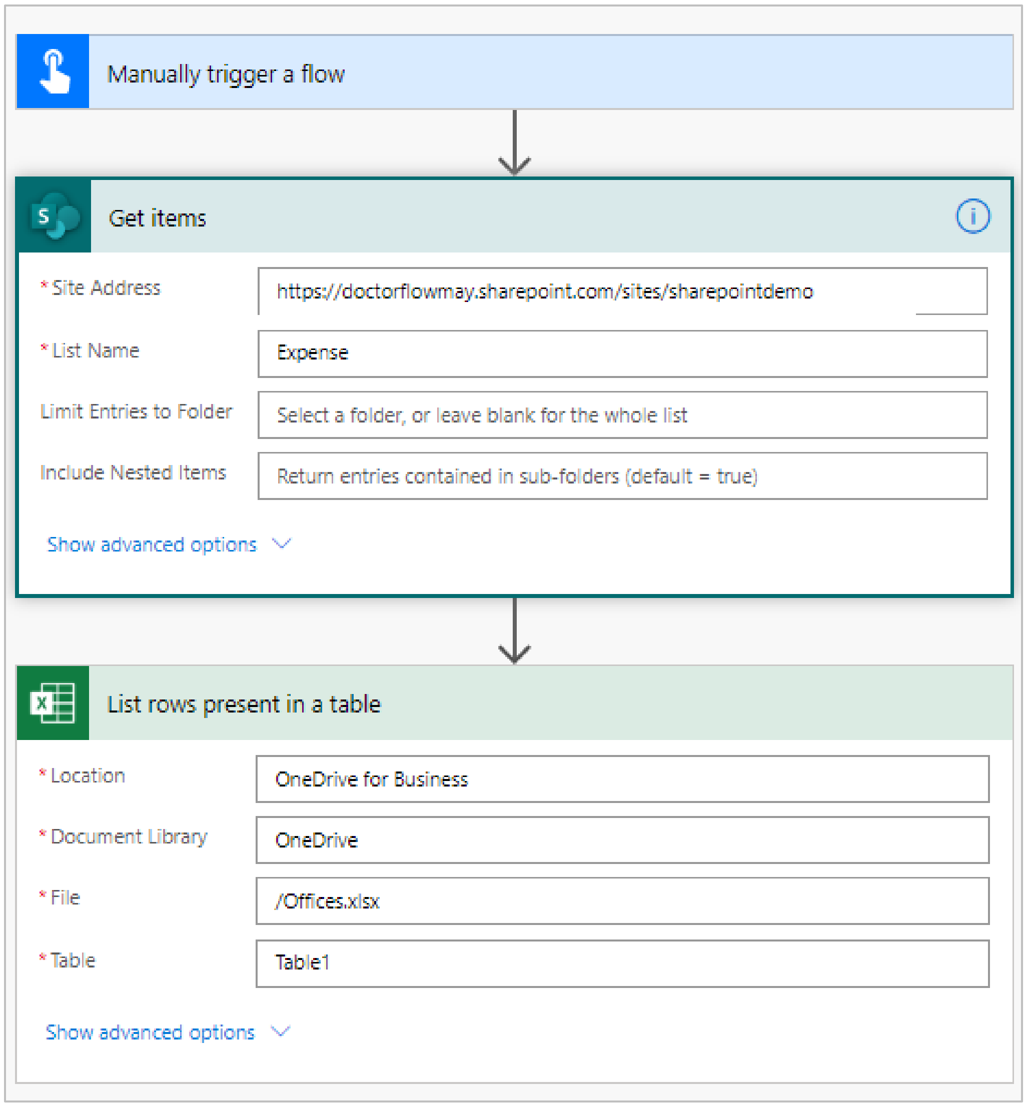
Name *: SergeSolution

Publisher *: Contoso

Version *: 1.0.0.0

2. Click Create:

Click your solution and create a Flow that is started manually and that connects to a SharePoint list and to an Excel document with a table:



Manually trigger a flow

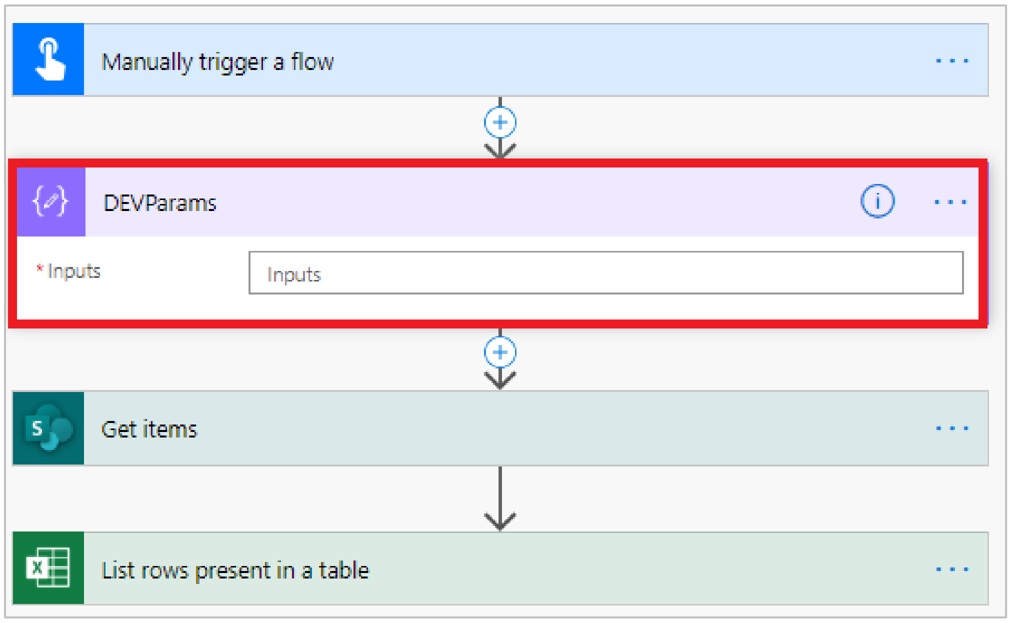
Get items

* Site Address	https://doctorflowmay.sharepoint.com/sites/sharepointdemo
* List Name	Expense
Limit Entries to Folder	Select a folder, or leave blank for the whole list
Include Nested Items	Return entries contained in sub-folders (default = true)

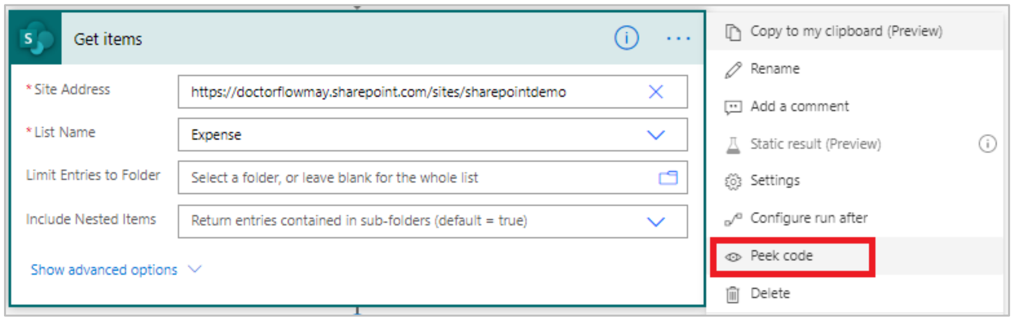
List rows present in a table

* Location	OneDrive for Business
* Document Library	OneDrive
* File	/Offices.xlsx
* Table	Table1

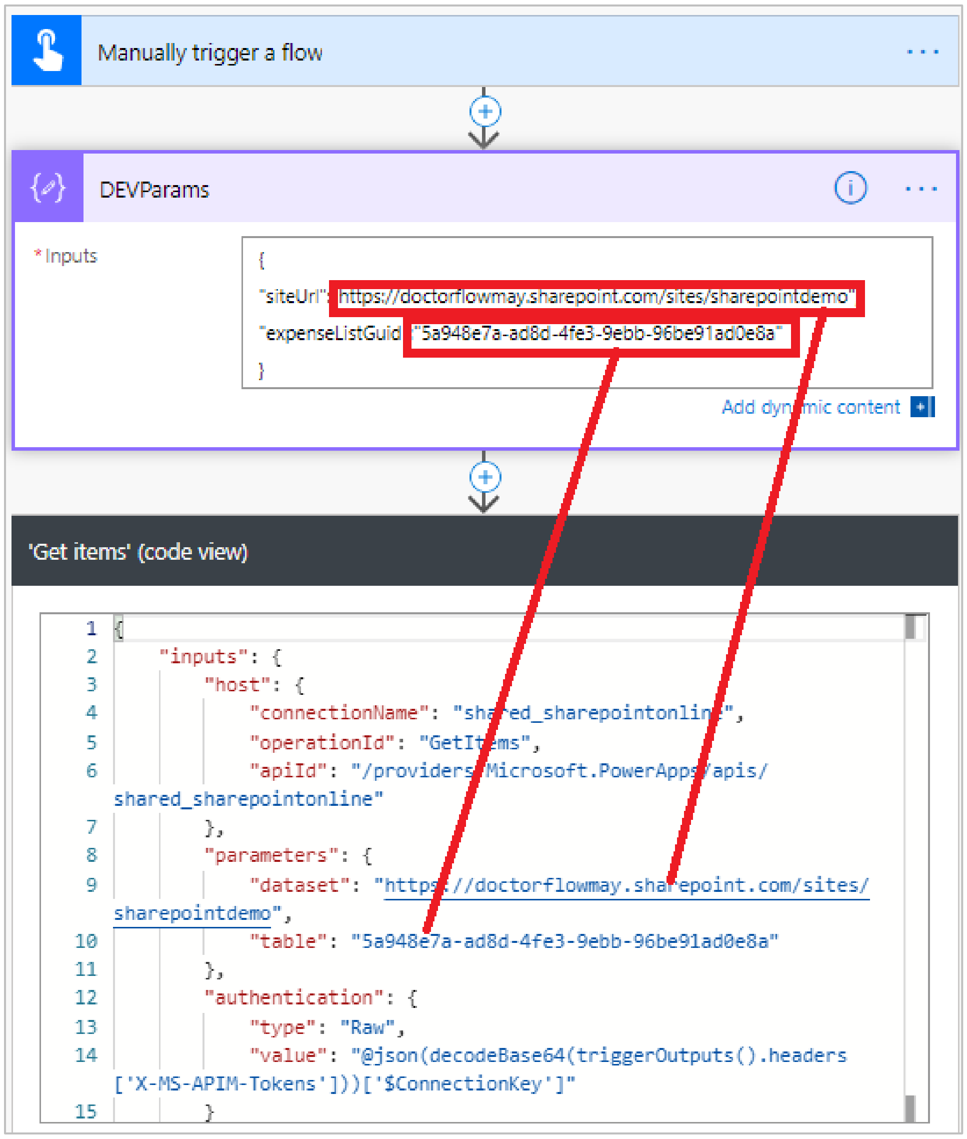
Enterprise flows can have many actions and deploying these flows from different environments (DV, TEST, PROD) can quickly become a nightmare.
The first option is to create a new Compose called DEVParams that will contain the internal references:



To retrieve the references, you need to click the Peek code option of the action:


Get items

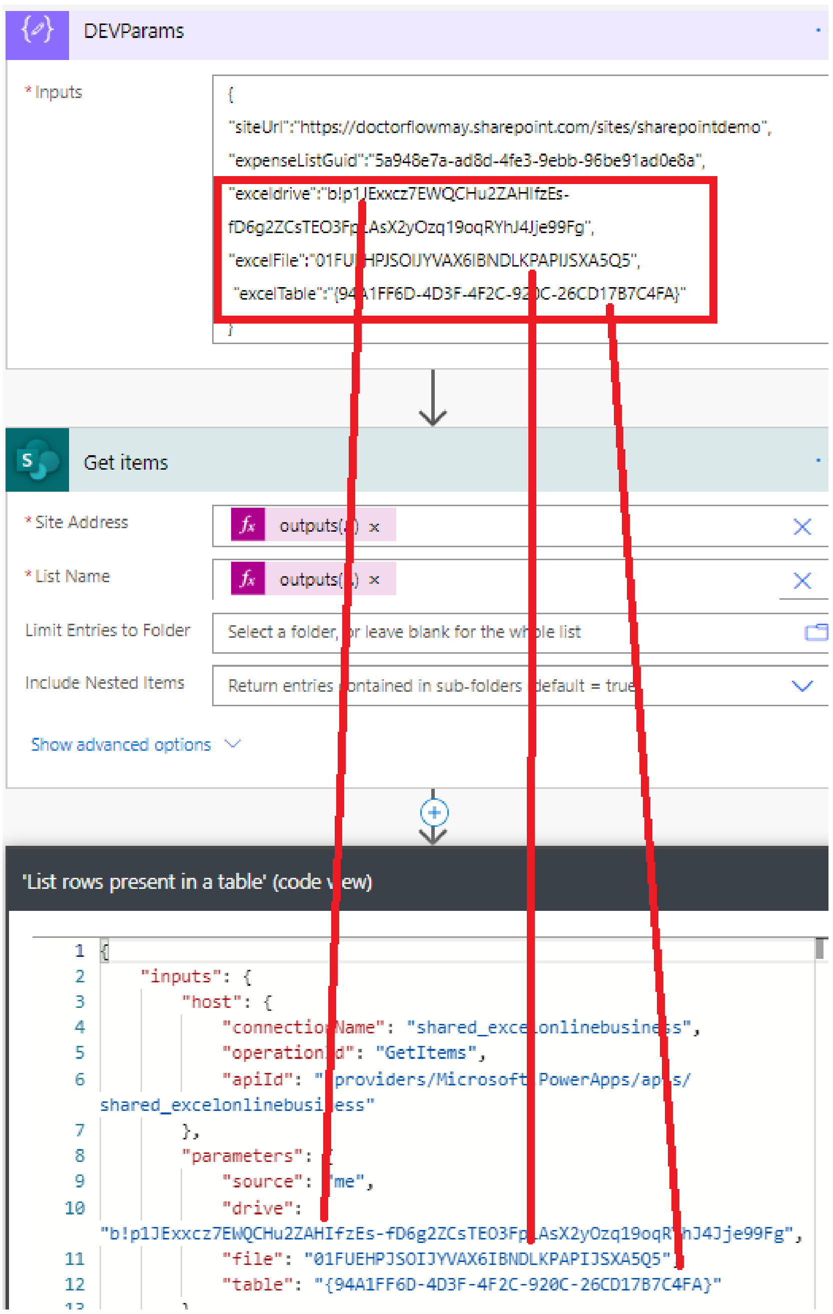
* Site Address	https://doctorflowmay.sharepoint.com/sites/sharepointdemo
* List Name	Expense
Limit Entries to Folder	Select a folder, or leave blank for the whole list
Include Nested Items	Return entries contained in sub-folders (default = true)



```
{
"siteUrl": "https://doctorflowmay.sharepoint.com/sites/sharepointdemo",
"expenseListGuid": "5a948e7a-ad8d-4fe3-9ebb-96be91ad0e8a"
}
```

```
{
    "inputs": {
        "host": {
            "connectionName": "shared_sharepointonline",
            "operationId": "GetItems",
            "apiId": "/providers/Microsoft.PowerApps/apis/
shared_sharepointonline"
        },
        "parameters": {
            "dataset": "https://doctorflowmay.sharepoint.com/sites/
sharepointdemo",
            "table": "5a948e7a-ad8d-4fe3-9ebb-96be91ad0e8a"
        },
        "authentication": {
            "type": "Raw",
            "value": "@json(decodeBase64(triggerOutputs().headers
['X-MS-APIM-Tokens']))['$ConnectionKey']"
        }
```

Proceed like this to retrieve the references used by the Excel action :



DEVParams

* Inputs

```
{
"siteUrl":"https://doctorflowmay.sharepoint.com/sites/sharepointdemo",
"expenseListGuid":"5a948e7a-ad8d-4fe3-9ebb-96be91ad0e8a",
"exceldrive":"b!p1JExxcz7EWQCHu2ZAHIfzEs-
fD6g2ZCsTEO3FpLAsX2yOzq19oqRYhJ4Jje99Fg",
"excelFile":"01FUEHPJSOIJYVAX6IBNDLKPAPIJSXA5Q5",
 "excelTable":"{94A1FF6D-4D3F-4F2C-920C-26CD17B7C4FA}"
}
```

Get items

* Site Address: outputs(...)

* List Name: outputs(...)

Limit Entries to Folder: Select a folder, or leave blank for the whole list

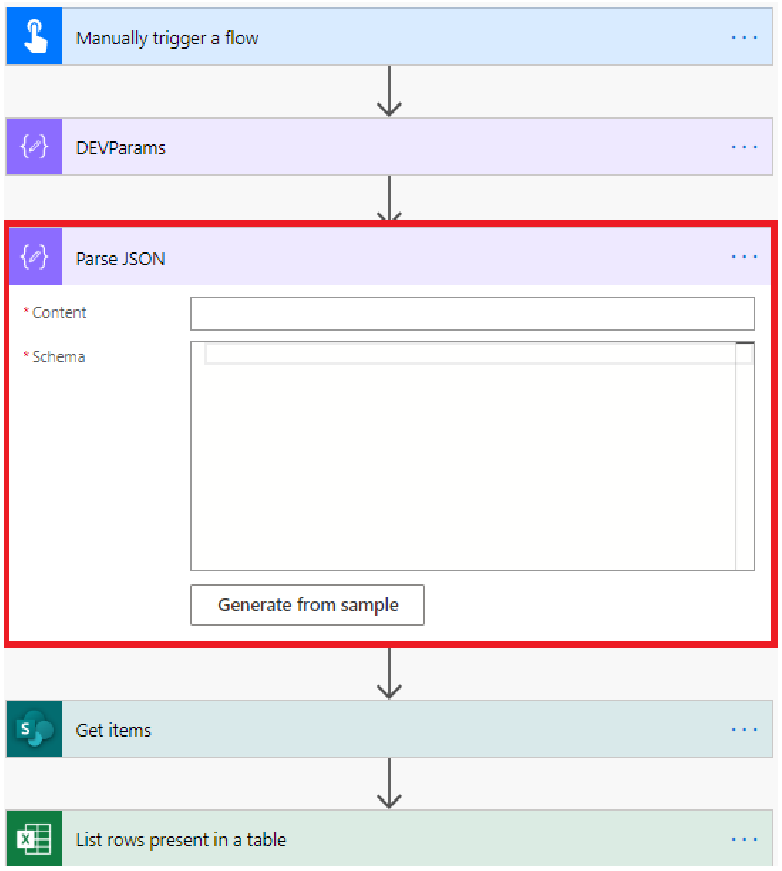
Include Nested Items: Return entries contained in sub-folders (default = true)

Show advanced options

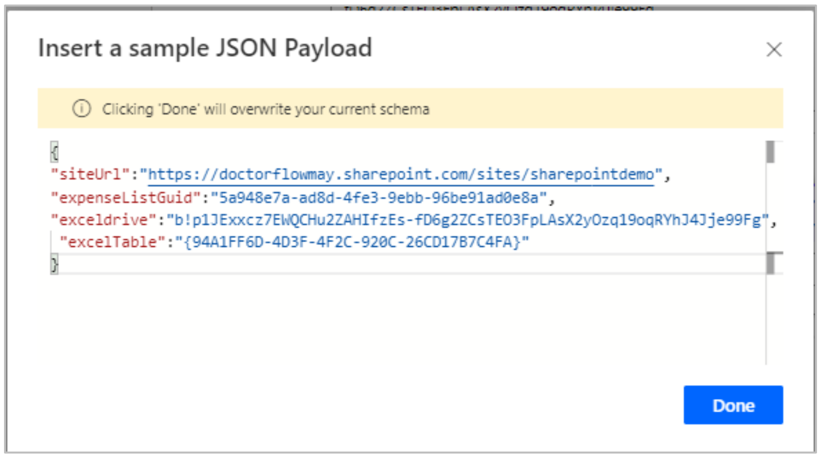
'List rows present in a table' (code view)

```
{
    "inputs": {
        "host": {
            "connectionName": "shared_excelonlinebusiness",
            "operationId": "GetItems",
            "apiId": "/providers/Microsoft.PowerApps/apis/
shared_excelonlinebusiness"
        },
        "parameters": {
            "source": "me",
            "drive":
"b!p1JExxcz7EWQCHu2ZAHIfzEs-fD6g2ZCsTEO3FpLAsX2yOzq19oqRYhJ4Jje99Fg",
            "file": "01FUEHPJSOIJYVAX6IBNDLKPAPIJSXA5Q5",
            "table": "{94A1FF6D-4D3F-4F2C-920C-26CD17B7C4FA}"
```

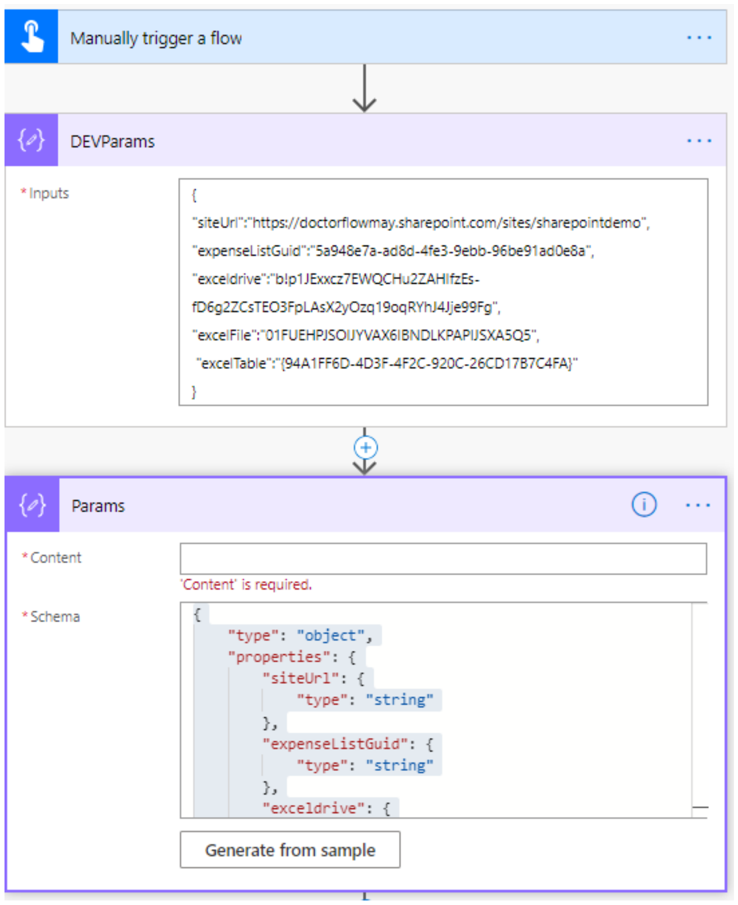
We need to use dynamic properties to use these references in our code. Add a Parse JSOn action and rename it **Params**:



Copy the JSOn information from the action DEVParams and click **Generate from sample**, paste the info in there:


Insert a sample JSON Payload
Clicking 'Done' will overwrite your current schema
Done

```
{
"siteUrl":"https://doctorflowmay.sharepoint.com/sites/sharepointdemo",
"expenseListGuid":"5a948e7a-ad8d-4fe3-9ebb-96be91ad0e8a",
"exceldrive":"b!p1JExxcz7EWQCHu2ZAHIfzEs-fD6g2ZCsTEO3FpLAsX2yOzq19oqRYhJ4Jje99Fg",
 "excelTable":"{94A1FF6D-4D3F-4F2C-920C-26CD17B7C4FA}"
}
```



* Inputs

```
{
"siteUrl":"https://doctorflowmay.sharepoint.com/sites/sharepointdemo",
"expenseListGuid":"5a948e7a-ad8d-4fe3-9ebb-96be91ad0e8a",
"exceldrive":"b!p1JExxcz7EWQCHu2ZAHIfzEs-
fD6g2ZCsTEO3FpLAsX2yOzq19oqRYhJ4Jje99Fg",
"excelFile":"01FUEHPJSOIJYVAX6IBNDLKPAPIJSXA5Q5",
 "excelTable":"{94A1FF6D-4D3F-4F2C-920C-26CD17B7C4FA}"
}
```

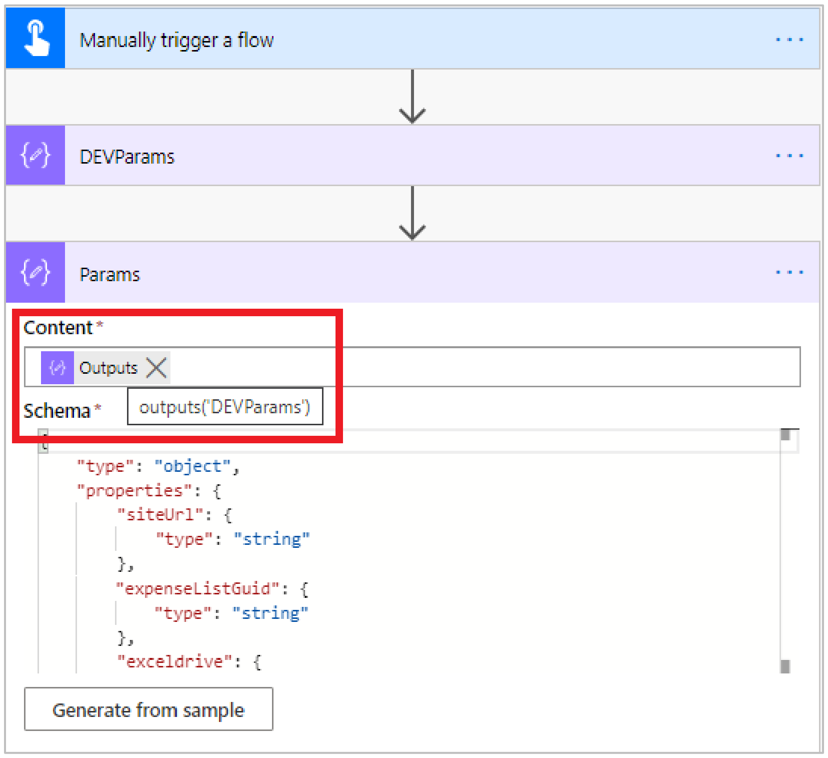
* Content

'Content' is required.

* Schema

```
{
    "type": "object",
    "properties": {
        "siteUrl": {
            "type": "string"
        },
        "expenseListGuid": {
            "type": "string"
        },
        "exceldrive": {
```

Use DevParams output in Params content:



Content *

Outputs

Schema * outputs('DEVParams')

```
{
    "type": "object",
    "properties": {
        "siteUrl": {
            "type": "string"
        },
        "expenseListGuid": {
            "type": "string"
        },
        "exceldrive": {
```

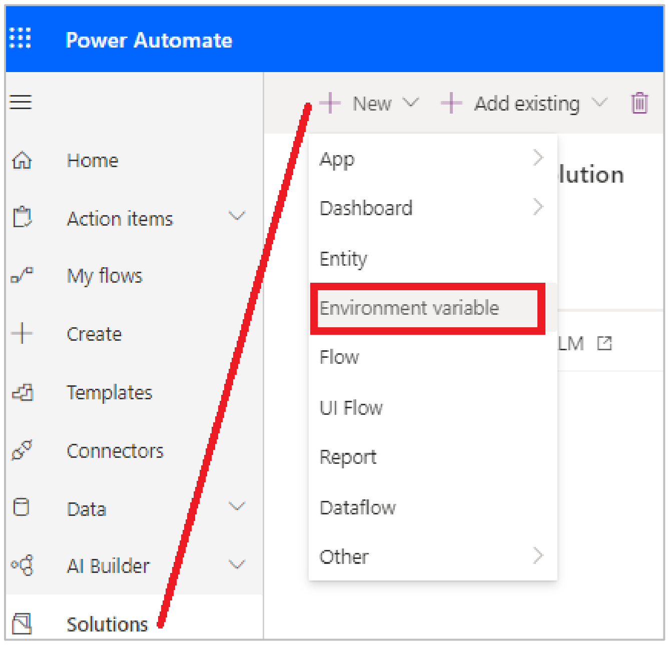
Replace the reference in the SharePoint and Excel actions with dynamic value. Now if you need to deploy your flow to another environment (like another SharePint Test site), you can create a new Compose object named TESTParams with the reference in test. And you just bind Params with TESTParams. Test your flow, it should still be working.

PART 2. Use environment variables

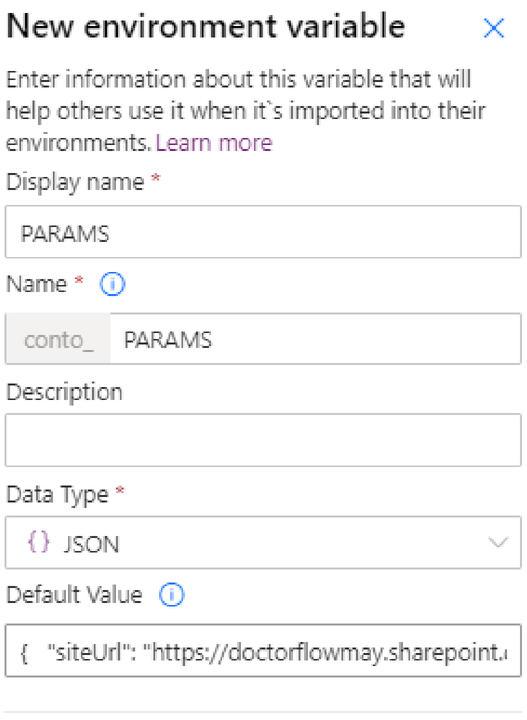
This JSOn information can be stored in an environment variable. In the next part we will create a child flow that will retrieve this information from environment variables.

Tasks:

1. Click your solution, select New and create an environment variable:



Name the environment variable "PARAMS" and select the Data type JSON and store the JSON data stored in the action DEVParams and click Save:



New environment variable

Enter information about this variable that will help others use it when it`s imported into their environments. Learn more

Display name *

PARAMS

Name *

conto_ PARAMS

Description

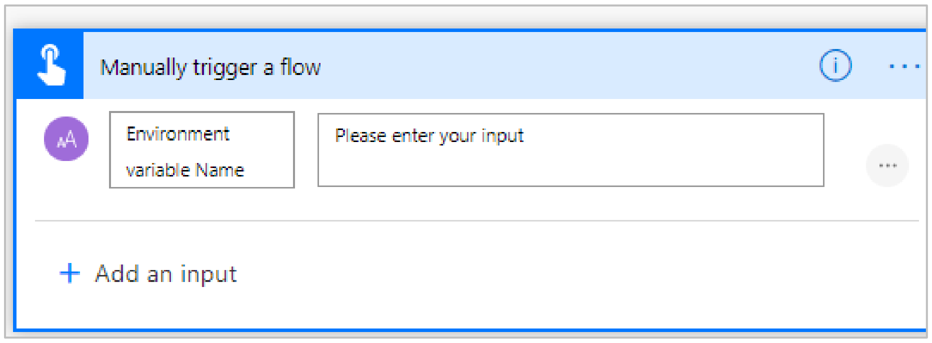
Data Type *

{} JSON

Default Value

{ "siteUrl": "https://doctorflowmay.sharepoint.

We now need to dynamically retrieve this information from our flows. From the solution create new flow and named it "Find Params". The trigger should be the "Manually trigger a flow" trigger. Add a first parameter called Environment variable:

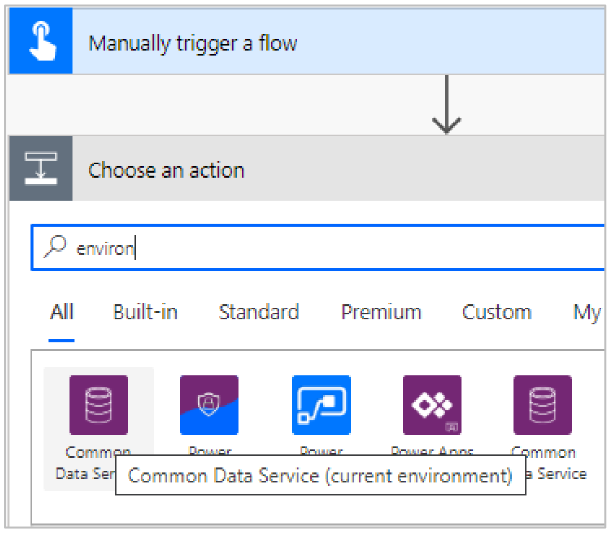


Manually trigger a flow

Environment variable Name

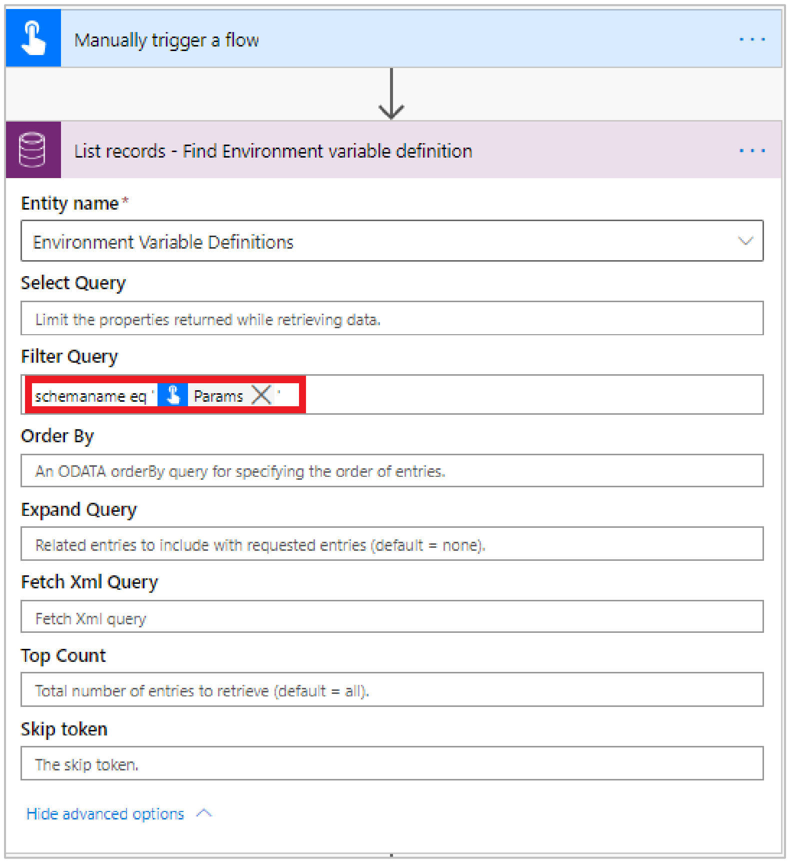
Please enter your input

Add an action **List records** from the Common Data Service (current environment) connector:



Rename the action as **List records - Find Environment variable definition**

In the filter **Query propery**, filter on *schename eq '<your environment variable name>'*:



List records - Find Environment variable definition

Entity name*: Environment Variable Definitions

Select Query: Limit the properties returned while retrieving data.

Filter Query: schemaname eq ' Params '

Order By: An ODATA orderBy query for specifying the order of entries.

Expand Query: Related entries to include with requested entries (default = none).

Fetch Xml Query: Fetch Xml query

Top Count: Total number of entries to retrieve (default = all).

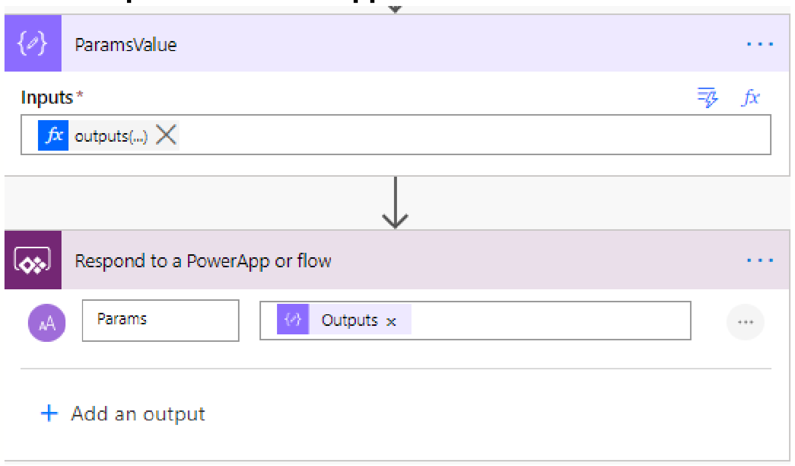
Skip token: The skip token.

Add a **Compose** action and name it **ParamsValue** :

Set its value to

```
outputs('List_records_-
_Find_Environment_variable_definition')?['body']?['value']?[0]?['defaultvalue']
```

Add a **Respond to a PowerApps or flow** action to return the **ParamsValue**:


ParamsValue

Inputs *

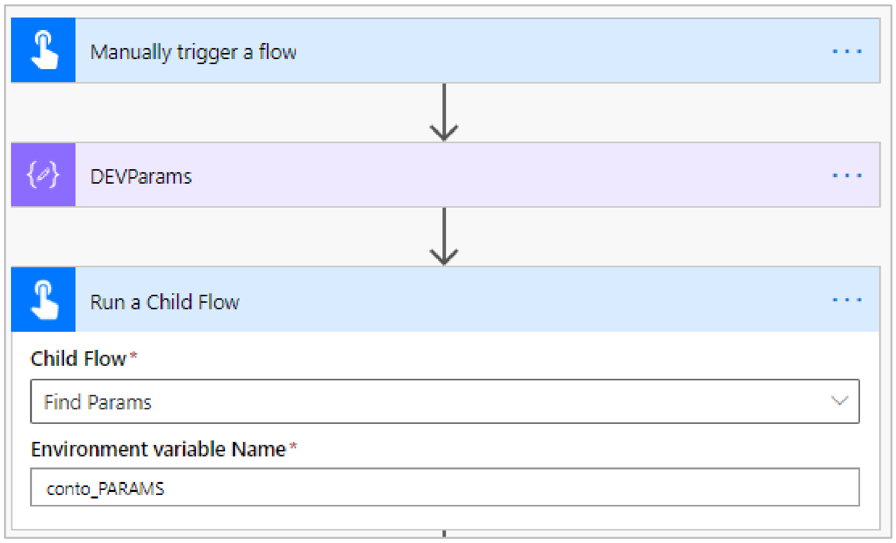
outputs(...)

Respond to a PowerApp or flow

Params | Outputs

You will now call his flow from the flow you created in part 1 of this lab. Edit the flow

and add a Run a child flow action



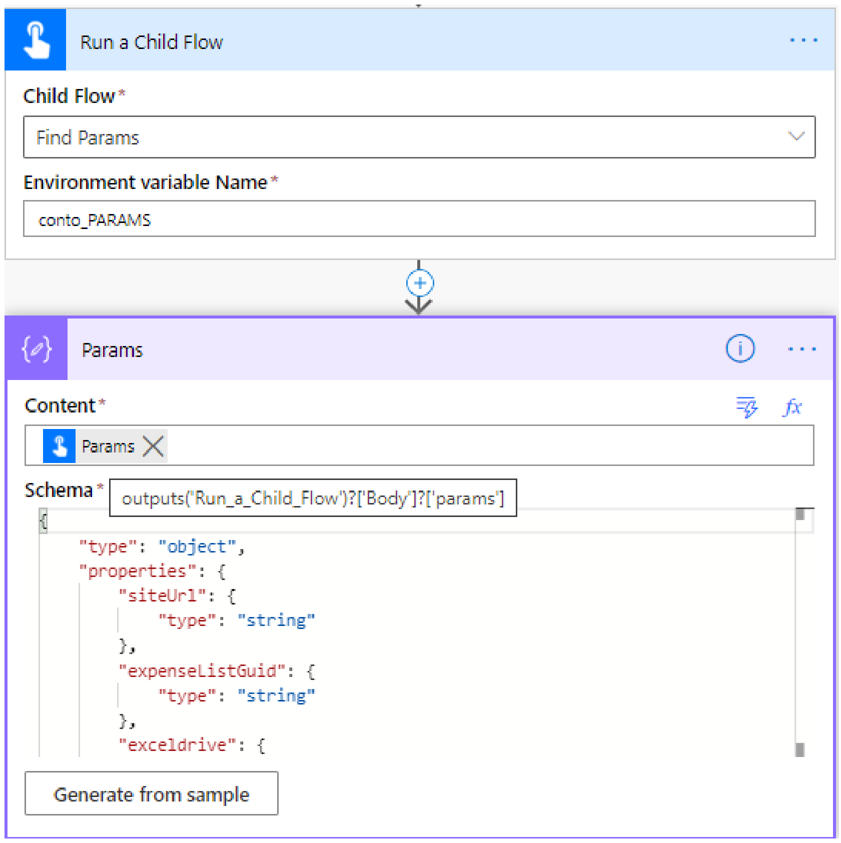
Child Flow *

Find Params

Environment variable Name *

conto_PARAMS

In the Params action, grab the returned value of Run a child flow :



Run a Child Flow

Child Flow*

Find Params

Environment variable Name*

conto_PARAMS

Params

Content*

Params

Schema*

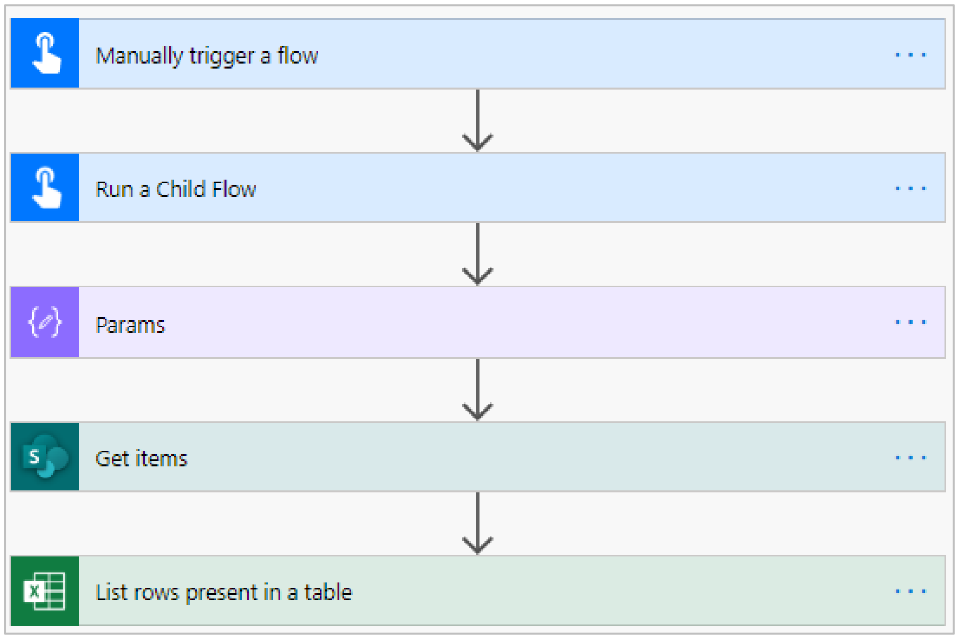
```
outputs('Run_a_Child_Flow')?['Body']?['params']
```

```
{
    "type": "object",
    "properties": {
        "siteUrl": {
            "type": "string"
        },
        "expenseListGuid": {
            "type": "string"
        },
        "exceldrive": {
```

Generate from sample

Test your flow.

The Action **DEVParams** can now be removed.



Manually trigger a flow

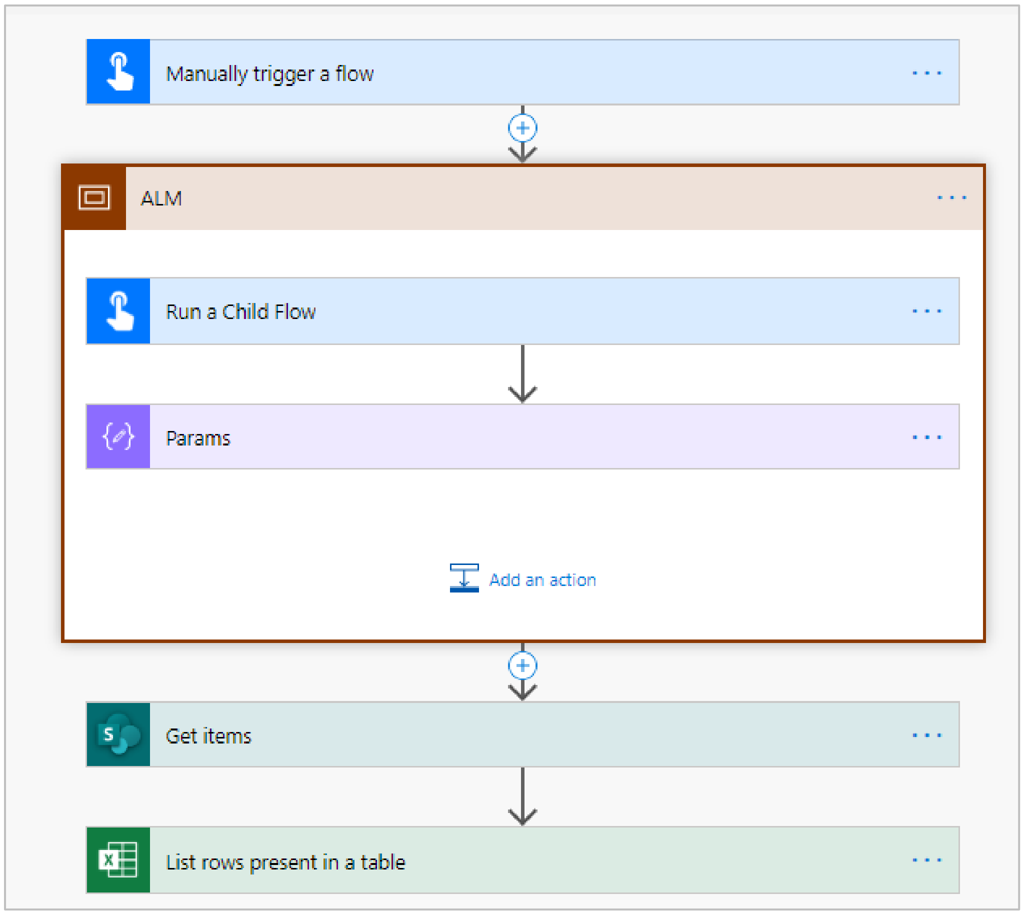
Run a Child Flow

Params

Get items

List rows present in a table

As a good practice we can group Run a child flow and Params in a **scope** called **ALM**:



We need your feedback

Do you want to report an issue or to suggest something? We need your feedback: https://github.com/Power-Automate-in-a-day/Training-by-the-community/issues