

Azure Data Factory

Lab : Control Flow Activity with Logic App

Pre-requisites:

- Azure Pass subscription
- Azure Data Lake Storage Gen2 storage account
- Azure Data Factory

Lab Objective:

After completing this lab, you will be able to:

- Use control flow activities
- Send custom email through logic app

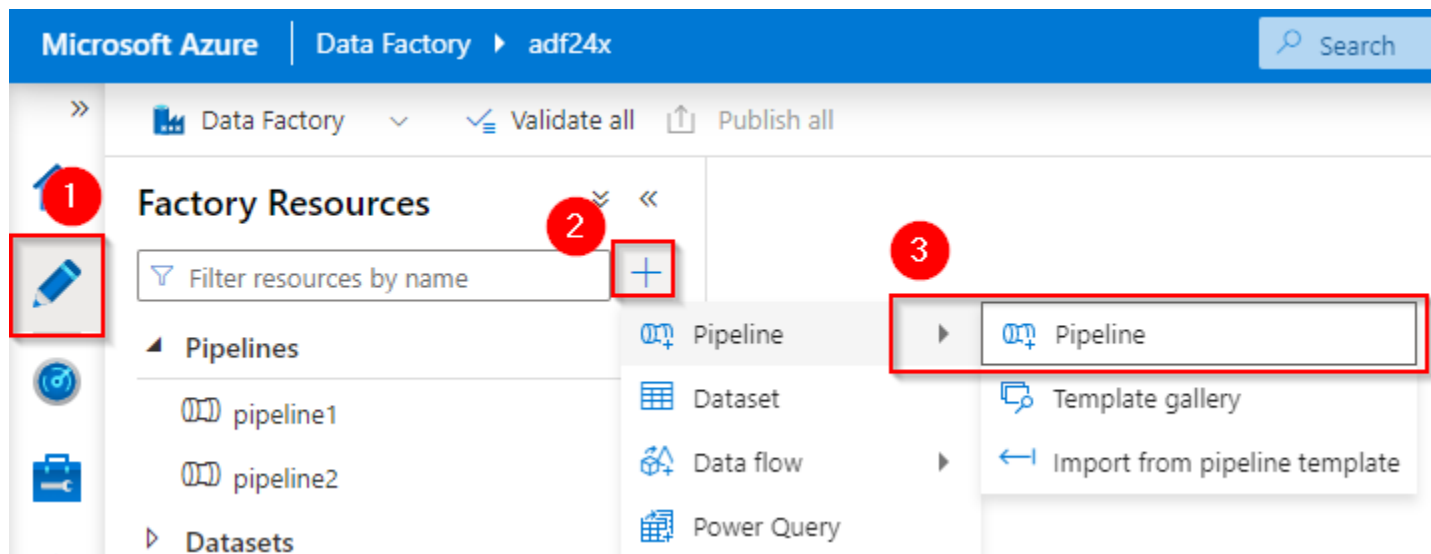
Exercise 1: Create a Pipeline to retrieve the valid list of file from Storage Account.

The task for this exercise are as follows:

1. Create a Pipeline and configure it
2. Append files to array based on modified date and size.

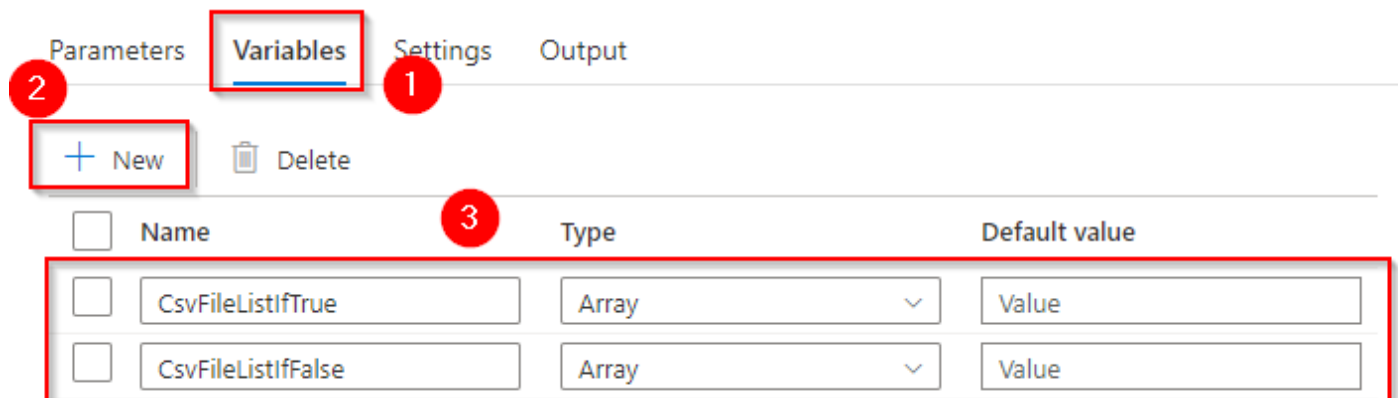
Task 1: Create a Pipeline and configure it.

1. Create a Pipeline in data factory **adfxx**.



2. Create two variables of array type in pipeline as follows:

- a. Name: CsvFileListIfTrue Type: Array
- b. Name: CsvFileListIfFalse Type: Array



Task 2: Append files to array based on modified date and size.

1. Add **Get Metadata** activity from **Activities** canvas under **General** Section to pipeline and configure it setting as:

a. **Dataset:**

- **Click + New**
- **Type:** Azure Data Lake Storage Gen 2
- **Format:** DelimitedText
- **Name:** FileListDataset
- **Linked Service:** AzureDataLakeStorage
- **File path:** data / output /
- **First row as header:** Unchecked
- **Import schema:** None
- **Click Ok**

b. **Field list:**

- Click + New
- From the drop down select: **Child items**

The screenshot shows the configuration interface for the 'Get Metadata' activity. The 'Settings' tab is selected. The 'Dataset' dropdown is set to 'FileListDataset'. The 'Field list' section shows a '+ New' button and a 'Delete' button. Below these, there is a checkbox for 'Argument' and a dropdown menu for 'Child items'. The 'Filter by last modified' and 'Skip line count' options are also visible.

General **Settings** User properties

1

2

3

4

Dataset *

Field list *

+ New Delete

☐ Argument

☐ Child items

Start time (UTC)

End time (UTC)

Filter by last modified ⓘ

Skip line count

##: you can do the debug run to check the out of Get Metadata activity

2. Add **Filter** activity to pipeline from activities canvas under **General** section and configure as follows:

General -> Name: FilterCsvFiles

Setting:

Items: @activity('Get Metadata1').output.childItems

Condition: @contains(item().name, '.csv')

The screenshot shows the configuration for the Filter activity. The 'Settings' tab is selected, indicated by a red circle with the number 1. The 'Items' field is set to '@activity('Get Metadata1').output.child...' and is highlighted with a red circle with the number 2. The 'Condition' field is set to '@contains(item().name, '.csv')' and is highlighted with a red circle with the number 3.

Connect Get Metadata activity on success to Filter activity

3. Add **ForEach** activity to pipeline from Activity canvas under **Iteration and conditionals** and configure it as:

Setting -> **items:** @activity('FilterCsvFiles').output.value

The screenshot shows the configuration for the ForEach activity. The 'Settings' tab is selected, indicated by a red circle with the number 1. The 'Items' field is set to '@activity('FilterCsvFiles').output.value' and is highlighted with a red circle with the number 2.

Activities -> Click on Pencil Icon.

The screenshot shows the 'Activities' tab for the ForEach activity. The 'Activities (0)' tab is selected, indicated by a red circle with the number 1. The table below shows the current state of the activities:

Case	Activity
ForEach	No activities

A red circle with the number 2 highlights the pencil icon in the bottom right corner, which is used to add or edit activities.

Connect Filter Activity to ForEach activity

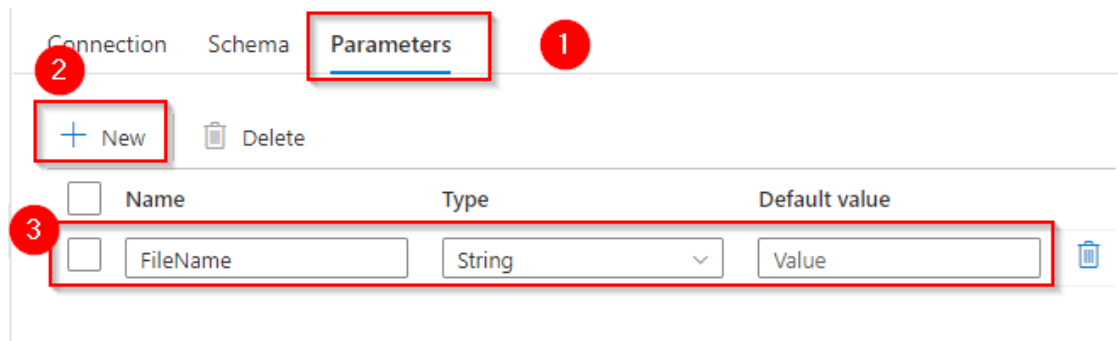
4. Inside the ForEach Activity add Get Metadata and configure it as:

General -> Name: Get Csv File Size

Setting:

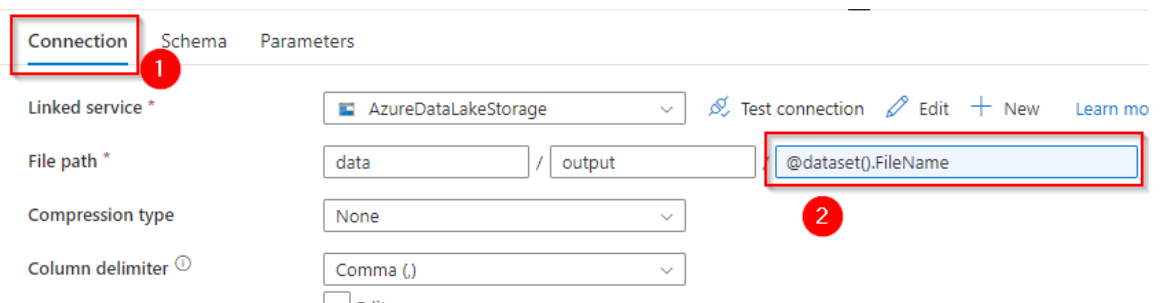
a. **Dataset:**

- **Click + New**
- **Type:** Azure Data Lake Storage Gen 2
- **Format:** DelimitedText
- **Name:** FileMetadataInfo
- **Linked Service:** AzureDataLakeStorage
- **File path:** data / output /
- **First row as header:** Unchecked
- **Import schema:** None
- **Advance:** Open this dataset
 - In Parameters tab create a new parameter
Name: FileName



Name	Type	Default value
FileName	String	Value

- In connection tab for File path give the File Name: **@dataset().FileName**



Linked service * AzureDataLakeStorage Test connection Edit + New Learn mo

File path * data / output @dataset().FileName

Compression type None

Column delimiter ① Comma (,)

b. Dataset Properties

FileName: @item().name

c. Field list:

- Click + New
- From the drop down select:
 - **Size**
 - **Last Modified**
 - **Item name**

The screenshot shows the 'Settings' tab of a data tool. The 'Dataset' dropdown is set to 'FileMetadataInfo'. The 'Dataset properties' table shows 'FileName' with the value '@item().name'. The 'Field list' section shows a list of fields: 'Argument', 'Size', 'Last modified', and 'Item name', each with a checkbox and a dropdown menu.

Name	Value
FileName	@item().name

Field	Value
Argument	
Size	
Last modified	
Item name	

5. Add **If condition** from **Iteration & Conditionals** section **inside** the **ForEach** activity in pipeline and enter the following expression under activities tab:

```
@and(  
    greaterOrEquals(  
        activity('Get Csv File Size').output.size,  
        500  
    ),  
  
    lessOrEquals(  
        activity('Get Csv File Size').output.lastModified,  
        convertFromUtc(utcNow(), 'India Standard Time')  
    )  
)
```

General **Activities (0)** User properties

1

Expression * ⓘ

2



```
@and(
  greaterOrEquals(
    activity('Get Csv File Size').output.size,
    500
  ),
  lessOrEquals(
    activity('Get Csv File Size').output.lastModified,
    convertFromUtc(utcNow(), 'India Standard Time')
  )
)
```

6. Click the pencil icon for True case

General **Activities (0)** User properties

Expression * ⓘ

```
@and(
  greaterOrEquals(
    activity('Get Csv File
```

Case	Activity	
True	No activities	
False	No activities	

Add **Append Variable** activity

Under **setting** tab of it provide the following details

Name : From Drop down list select **CsvFileListIfTrue**

Value: **@activity('Get Csv File Size').output.itemName**

General **Settings** User properties

1

Name *

2

CsvFileListIfTrue

Value

3

@activity('Get Csv File Size').output.ite...

+ New

Disclaimer: Append Variables Only Supports Adding To 'Array' Type Variables

7. Go back to **ForEach** activity, Click the pencil icon for **False case** in **If Condition** Activity.

Add **Append Variable** activity

Under **setting** tab of it provide the following details

Name : From Drop down list select **CsvFileListIfFalse**

Value: **@activity('Get Csv File Size').output.itemName**

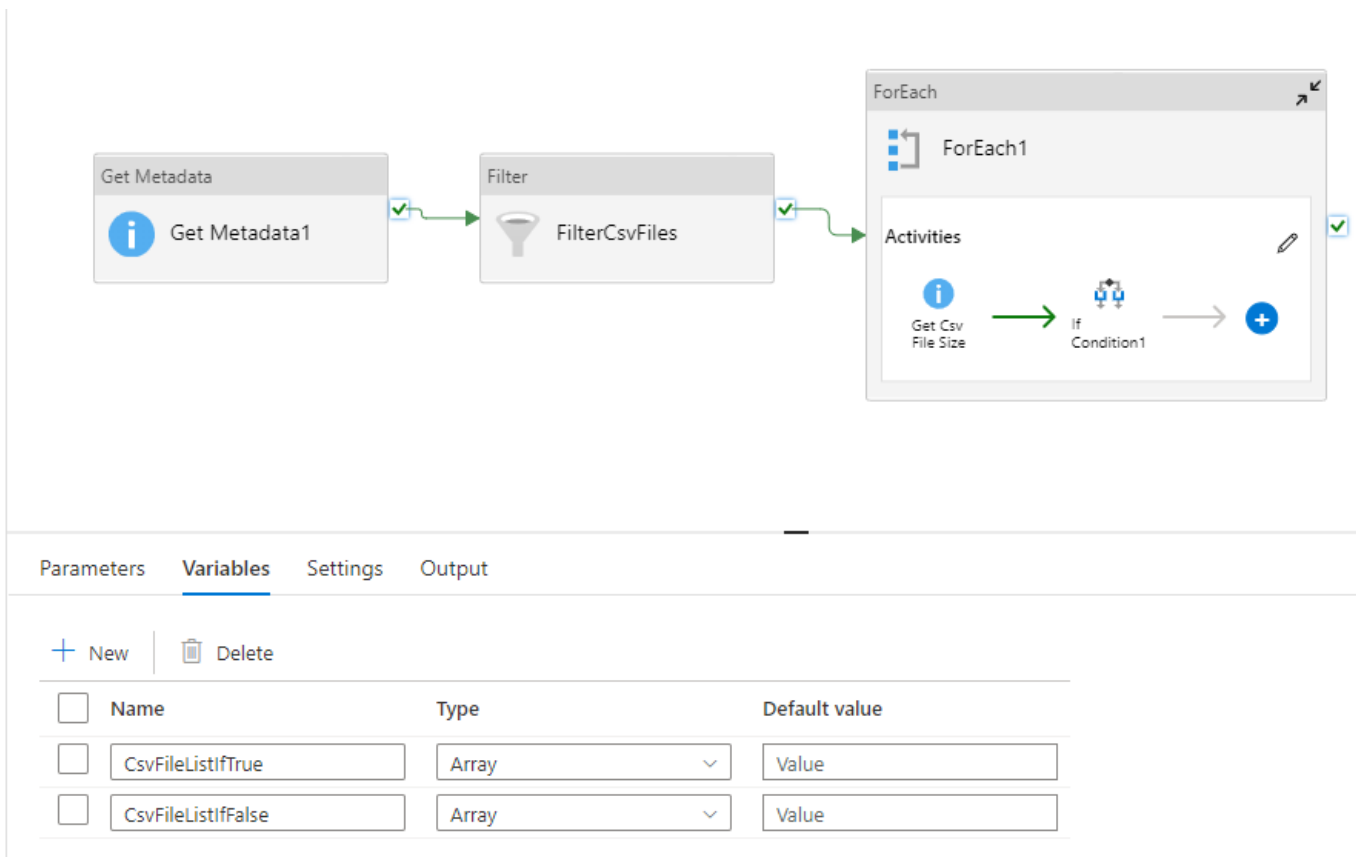
General **Settings** User properties

Name * CsvFileListIfFalse + New

Value @activity("Get Csv File Size").output.itemName

Disclaimer: Append Variables Only Supports Adding To 'Array' Type Variables

8. Publish the pipeline. Your pipeline look like as:



Exercise 2: Create a Logic App to send custom email from Azure Data Factory Pipeline.

The task for this exercise are as follows:

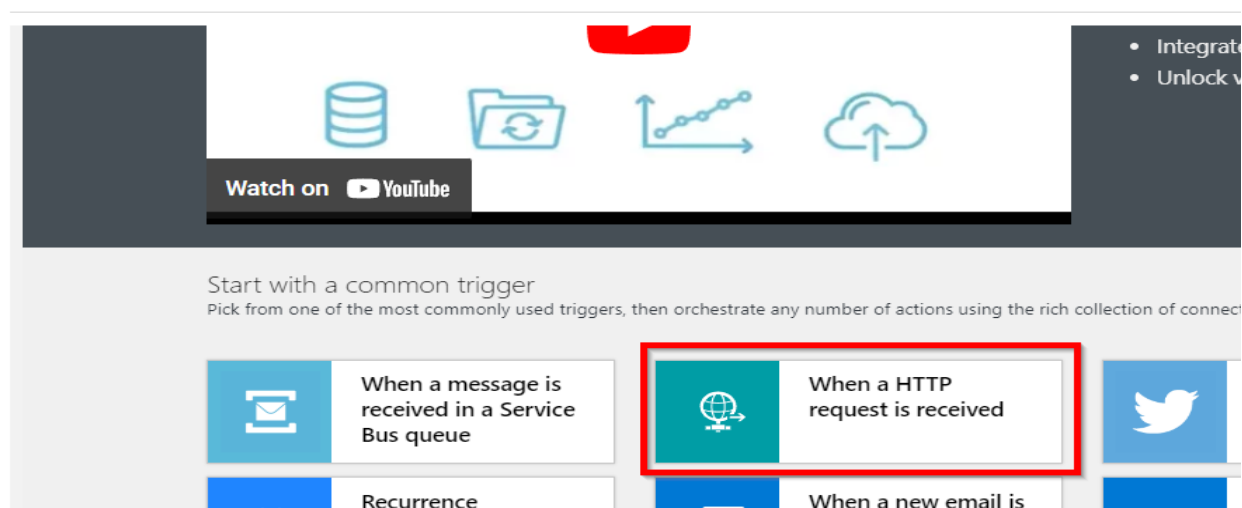
1. Create and Configure the Logic App to send the email.
2. Configure the pipeline to call the Logic App.

Task 1: Create Logic App

1. Search for **Logic Apps** in search bar at the top, click on Create and provide the following details.
 - a. **Subscription** → Azure Pass - Sponsorship
 - b. **Resource group Name** -> adfxx-rg
 - c. **Logic App name** → EmailLogicApp-xx
 - d. **Region** → Same as your Resource Group
 - e. **Enable log analytics** -> No
 - f. **Plan Type** → Consumption
2. Leave the other settings to Default and click **Review + Create** and then **Create**.
3. Once deployed click on **Go To Resource** and click on **When a HTTP request is received**. Or inside the logic app select **Logic App Designer** from the left-hand side menu under development Tools section.

[Home](#) > [EmailLogicApp34x](#) >

Logic Apps Designer ...



4. If coming from Logic App Designer then under Choose and Operation, Search for **Request**, Select the **Request** and then Select **When a HTTP Request is received**.
5. Click Use Sample payload to generate schema and paste the following code and click on Done.

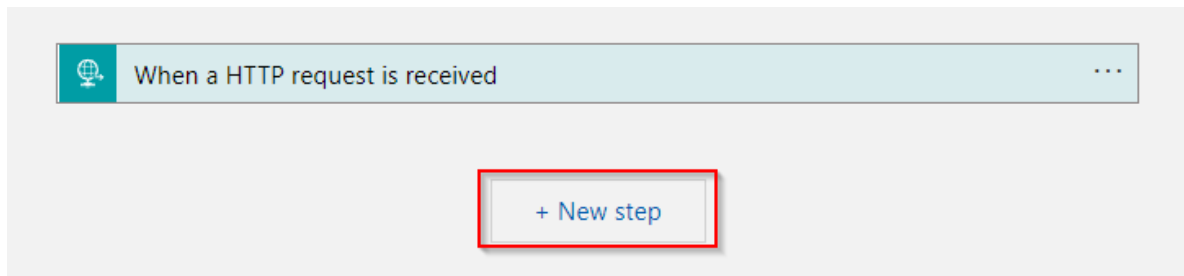
```
{  
  "InvalidFileList" : [],  
  "ValidFileList" : [],  
  "PipelineRunId" : [],  
  "PipelineName" : "",  
  "ADFName" : ""  
}
```

The screenshot shows the configuration pane for the 'When a HTTP request is received' trigger. The title bar is light blue with a globe icon and the text 'When a HTTP request is received'. Below the title bar, there is a section for 'HTTP POST URL' with a text box containing 'URL will be generated after save' and a copy icon. Below this is a section for 'Request Body JSON Schema' with a large text area. At the bottom of the pane, there is a button labeled 'Use sample payload to generate schema' which is highlighted with a red rectangle. Below the button is a dropdown menu labeled 'Add new parameter'.

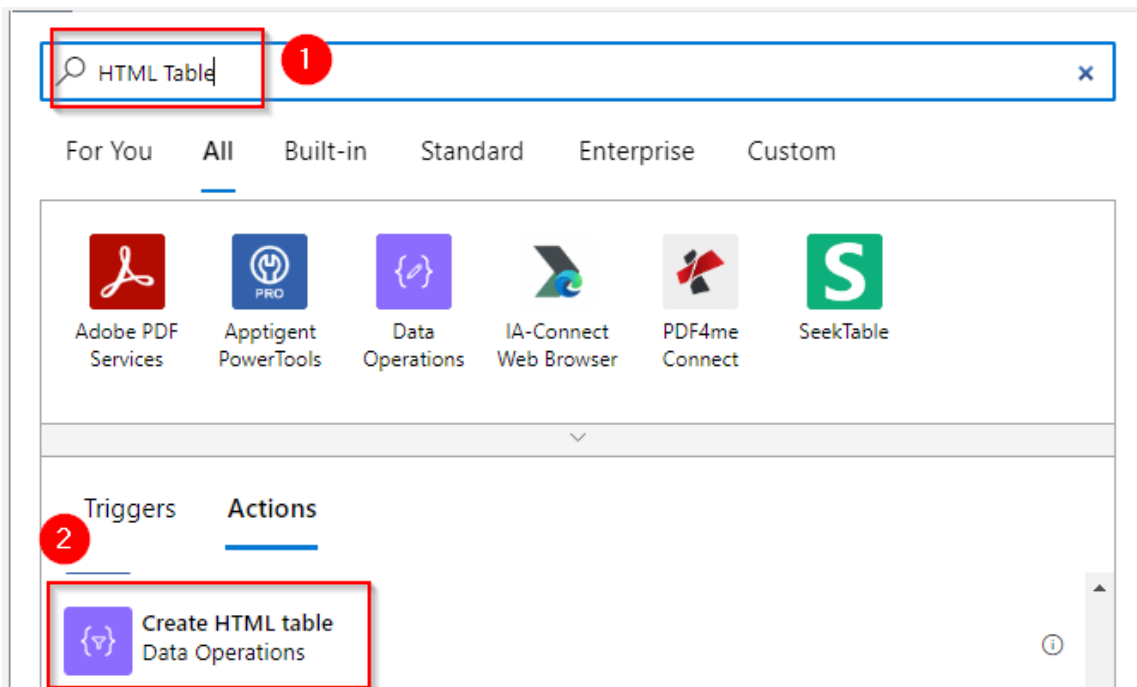
6. Click on Add new parameter, For Method and Select POST.

The screenshot shows the 'Add new parameter' dialog box. It has a title bar with the text 'Use sample payload to generate schema'. Below the title bar, there is a dropdown menu for 'Method' with 'POST' selected, highlighted with a red rectangle and a red circle labeled '1'. To the left of the dropdown menu is a red circle labeled '2'. Below the dropdown menu is a text box labeled 'Add new parameter' with a dropdown arrow, also highlighted with a red rectangle.

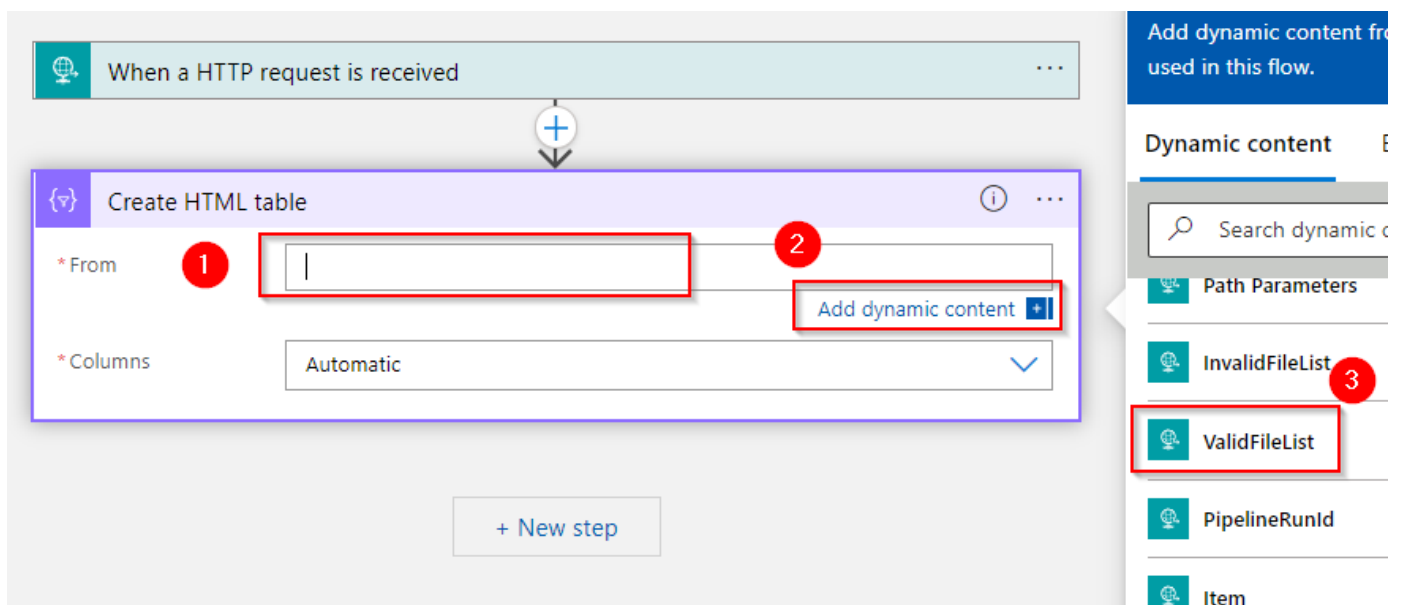
7. Click on the Plus icon to insert a new Step and Click on Add an Action.



8. Under choose an operation, search for HTML Table. Select the Create HTML Table and rename it to **Create HTML Table Valid File**

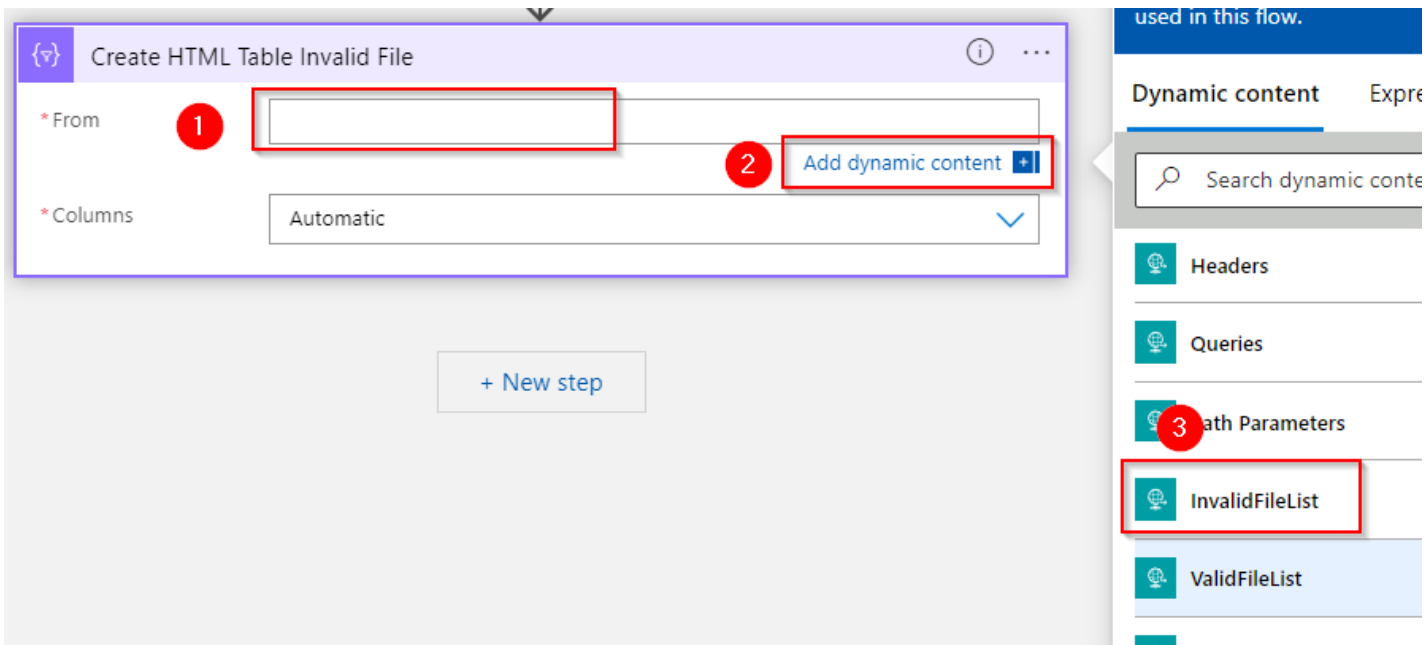


9. In Create HTML Table Valid File, do as show below.



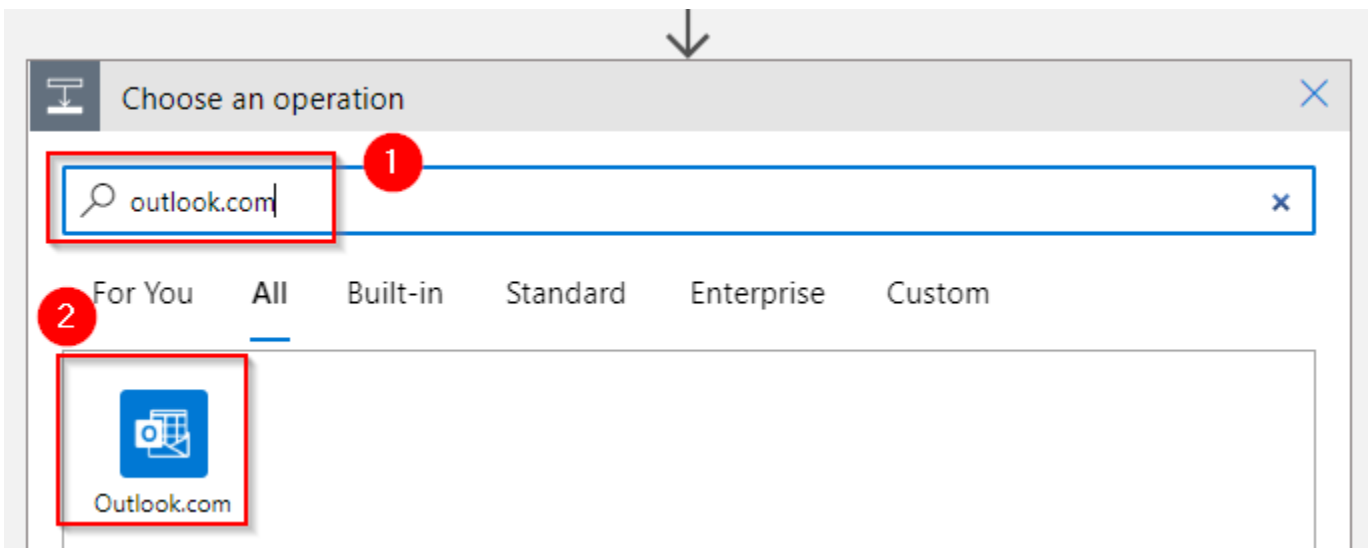
10. Click on the Plus icon to insert a new Step.

11. Under choose an operation, search for HTML Table. Select the Create HTML Table and rename it to **Create HTML Table Invalid File**

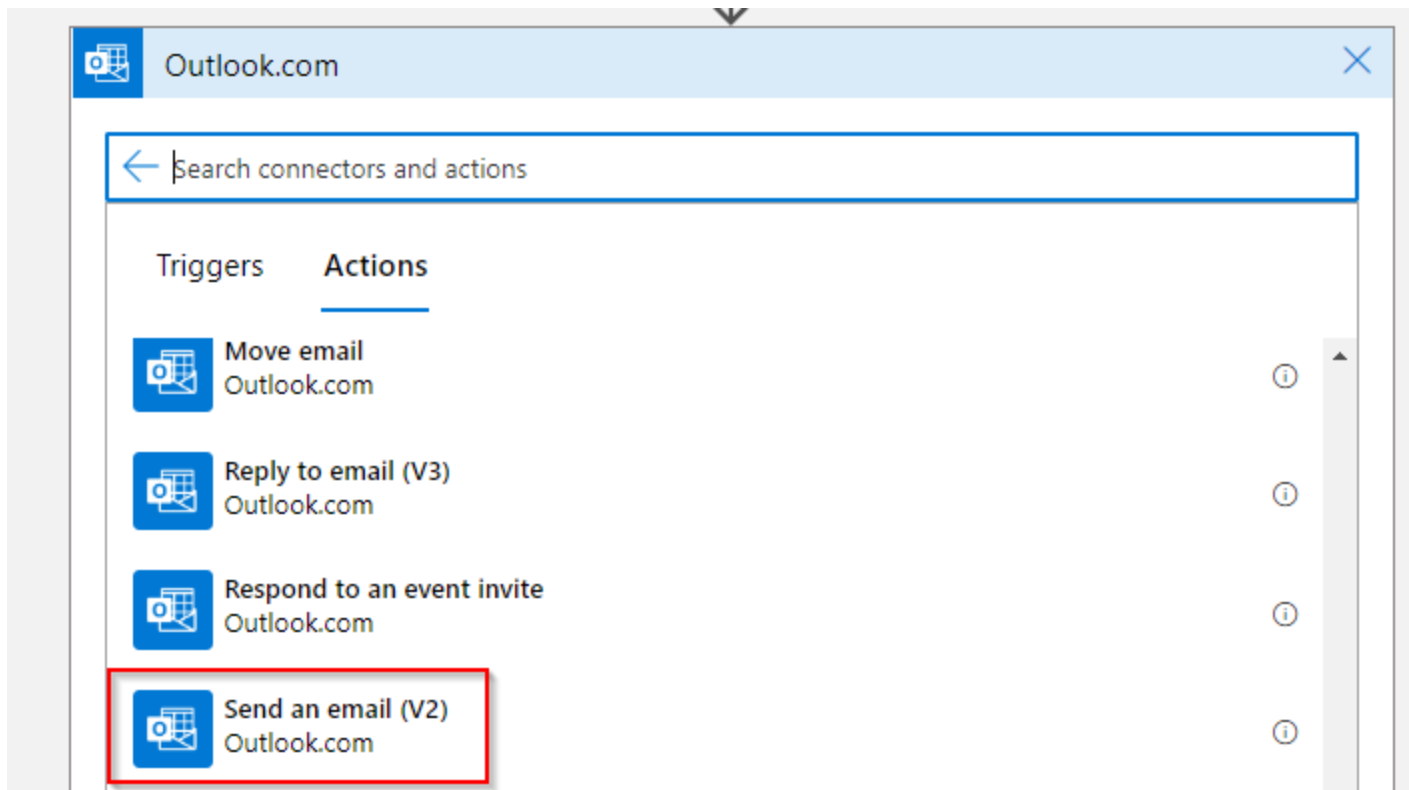


12. Click on the Plus icon to insert a new Step.

13. Under Choose an operation, Search for Outlook.com and select it.



14. Under Actions, Select **Send an email (V2)** and Sign in using an Outlook Account. (Create one if you don't have)



15.After Signing In, Click on Save.

16.Enter the following Details in Parameters.


To: Your Outlook Mail ID

Subject: Valid-Invalid Files – ADF Status

Body:


Hi,


Valid-Invalid File Status Update Message From ADF.

Name:  ADFName x

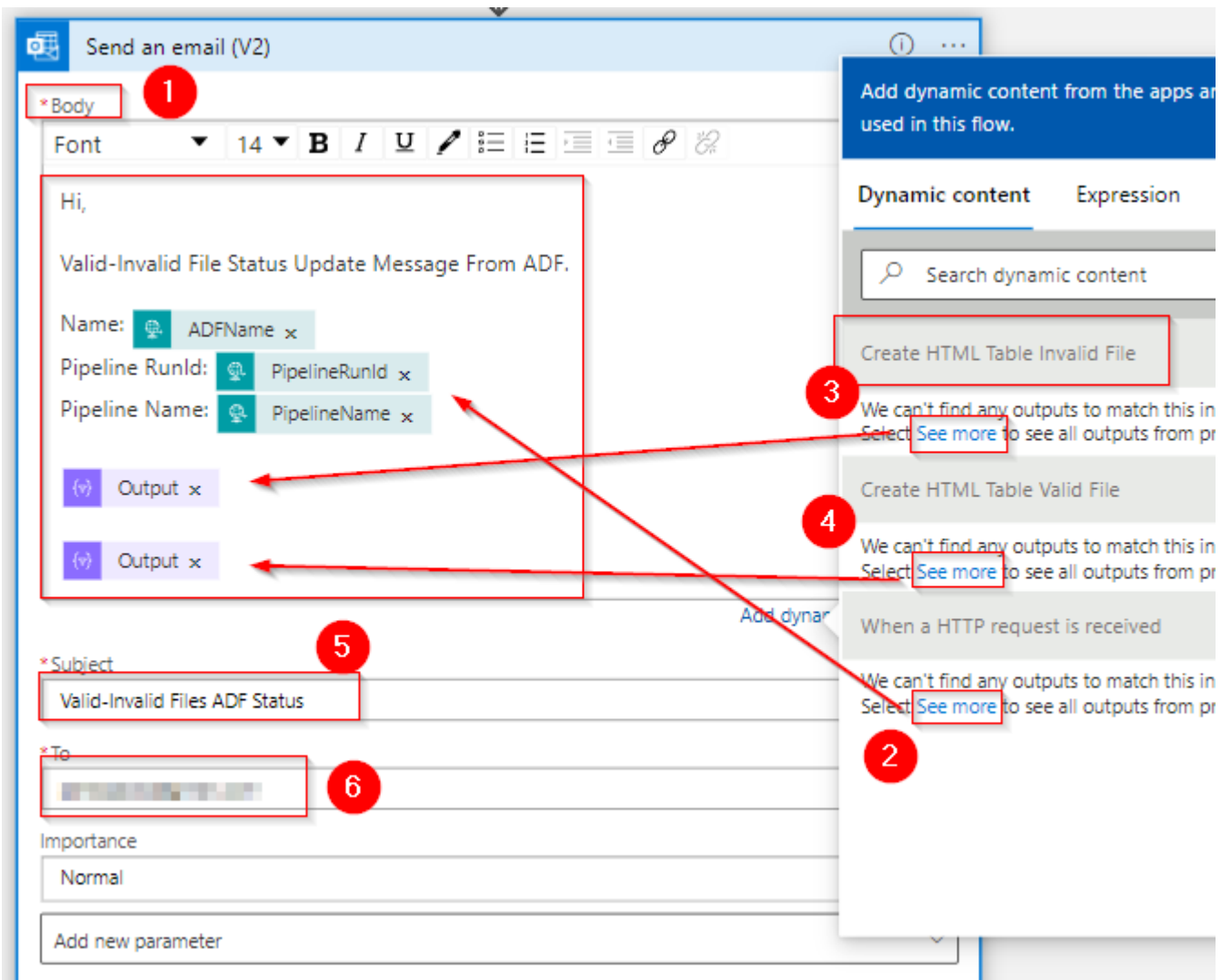
Pipeline RunId:  PipelineRunId x

Pipeline Name:  PipelineName x

 Output x

 Output x

Type the message body, drag and drop the relevant field as show above from the dynamic content section to its proper place (see below image).



17. Click on Save.

18. Go back to the first step, **When a HTTP request is received**. Open it and can copy the **HTTP Post URL** and paste it in a notepad.

Task 2: Configure the pipeline to send the email.

1. Go to the **Azure Data Factory** workspace and open the pipeline if not open already.
2. Add **Web Activity** to the pipeline from Activities canvas under General section. Connect **ForEach** activity to **web activity** on success.
3. Under **setting** tab of **Web Activity** provide the following details.

URL: past the **HTTP Post URL** copied from Logic App.

Method: **Post**

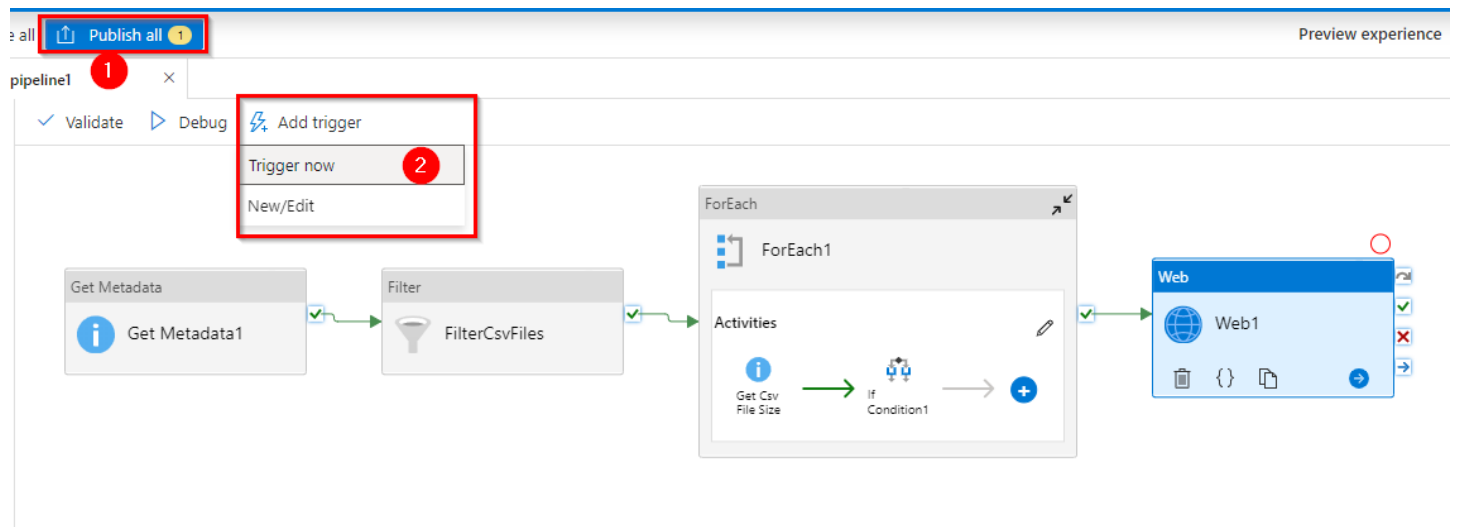
Body:

```
{  
  "InvalidFileList" : @{{variables('CsvFileListIfFalse')}},  
  "ValidFileList" : @{{variables('CsvFileListIfTrue')}},  
  "PipelineRunId" : "@{pipeline().RunId}",  
  "PipelineName" : "@{pipeline().Pipeline}",  
  "ADFName" : "@{pipeline().DataFactory}"  
}
```

The screenshot shows the 'Settings' tab of a Web Activity in Azure Data Factory. The interface includes tabs for 'General', 'Settings', and 'User properties'. The 'Settings' tab is active. The following fields are visible and annotated with red circles and numbers:

- 1**: Points to the 'Settings' tab.
- 2**: Points to the 'URL' field, which contains a placeholder URL ending in 'logic.azure.com'. A warning icon and message are displayed below the field: 'Information will be sent to the URL specified. Please ensure you trust the URL entered.'
- 3**: Points to the 'Method' dropdown menu, which is set to 'POST'.
- 4**: Points to the 'Body' text area, which contains the JSON payload: `{ "InvalidFileList" : @{{variables('CsvFile...`
- The 'Authentication' dropdown menu is set to 'None'.

4. Publish the pipeline and Run it.



5. When run successfully check the inbox of the provided mail.