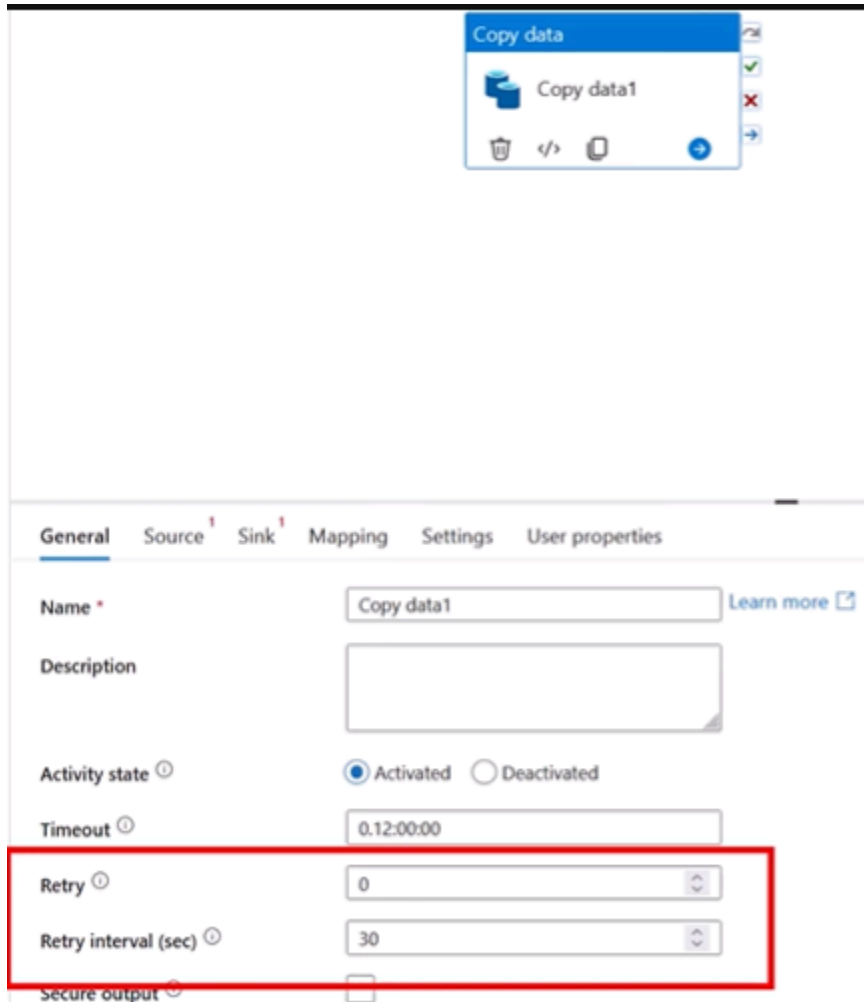


DP-203: 14 - Error Handling in ADF

Retry

1. First thing to handle the errors in ADF we can retrying our operation
2. Lets see it practically
3. Lets create a pipeline and add copy data activity



The screenshot shows the configuration for a 'Copy data' activity in Azure Data Factory. The 'General' tab is selected. The 'Name' field is 'Copy data1'. The 'Activity state' is 'Activated'. The 'Timeout' is '0.12:00:00'. The 'Retry' field is set to '0' and the 'Retry interval (sec)' field is set to '30'. These two fields are highlighted with a red box. The 'Secure output' checkbox is unchecked.

In the each activity

we have this retry option

4. Here if our pipeline encounter's an error...then it will wait for 30 sec and retry again for 3



The screenshot shows the configuration for a 'Copy data' activity in Azure Data Factory. The 'Retry' field is set to '3' and the 'Retry interval (sec)' field is set to '30'. A mouse cursor is pointing at the 'Retry interval (sec)' field.

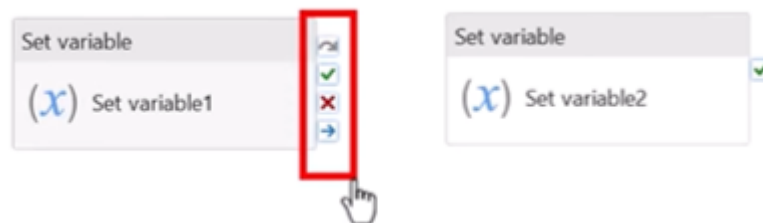
times

Conditional path

1. To understand this let us consider two Set variable activities inside the pipeline



2. Initially here in set variable 1...we have added a variable "myvar" which takes an integer and stores it
3. Similarly for the set variable 2 ..we did the same thing
4. Here SV2 only runs if the SV1 has successfully completed..if it fails then SV2 never



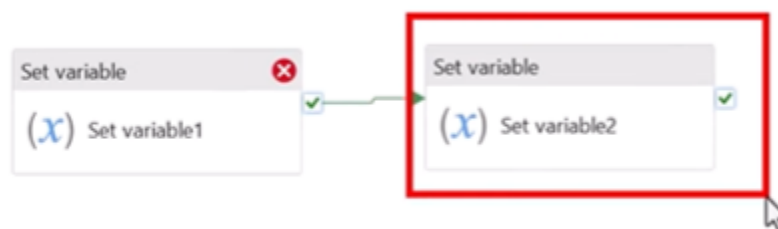
executes



because we have connected

its path via on success

5. Now let us change the "myvar" of SV1 to a string ...which gives an error and SV2 runs only when SV1 successfully executes



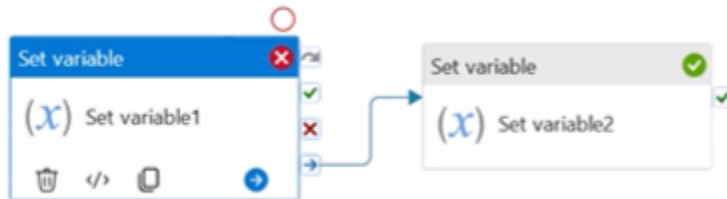
6. Here we can see SV2 has not run



7. here we have modifies the conditional path..to when SV1 fails then SV2 runs



8. We have 3 rd connector on Completion it says that after completing executing(even pass or fail) SV1 then move to SV2...



9. We have 4th connector on skip...basically it moves to 2nd activity if the first activity is skipped

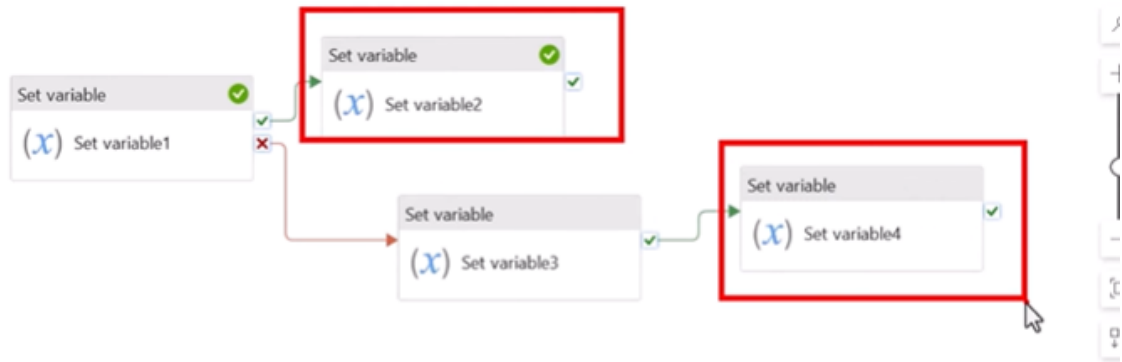
Pipeline Status

1. Here we have pipeline status as succeeded

| Activity name | Activity status | Activity type | Run start | Duration | Integration |
|---------------|-----------------|---------------|------------------------|--------------|-------------|
| Set variable2 | ✓ Succeeded | Set variable | 11/19/2023, 9:16:24 AM | Less than 1s | |
| Set variable1 | ✓ Succeeded | Set variable | 11/19/2023, 9:16:23 AM | Less than 1s | |

2. It only shows succeeded when all the leaf activity passes

3. Examples of leaf activities highlighted in squares



4. Suppose if the leaf activity is skipped...then it checks with the parent

Common Error Patterns

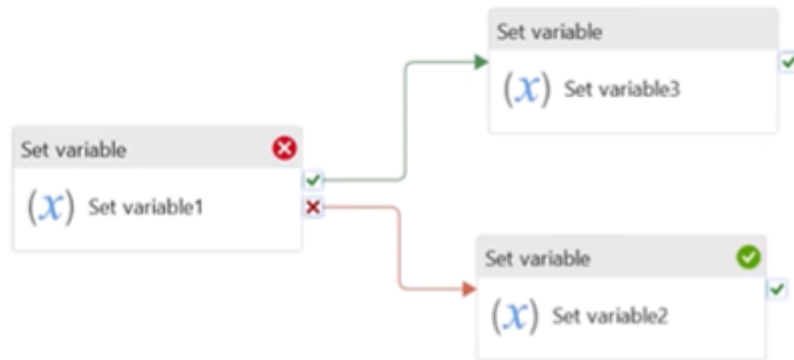
1. Try Catch method

Here we can make use of this connectors to perform try and catch



activity1 fails we can send the error to activity2 (SV2)

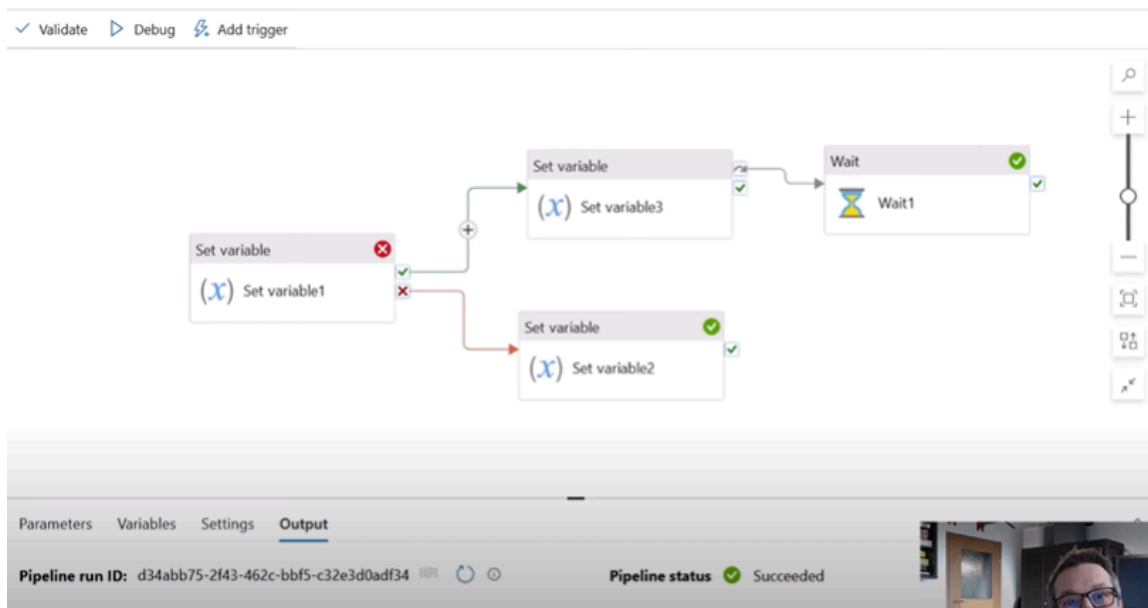
here if



2. DO IF ELSE

Here if activity1 (SV1) fails then it executes Activity2 (SV2) else it executes Activity3(SV3)

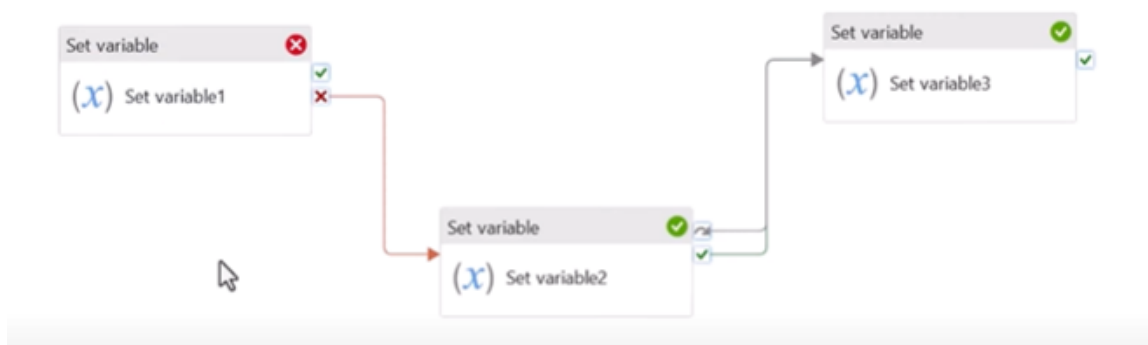
3. DO IF SKIP ELSE



Here SV3 is getting skipped and it resulted in Pipeline status Failed ...as it is leaf activity and parent activity failed

So to avoid pipeline failed status we added a dummy activity which executes when leaf activity is skipped

4. TRY CATCH PROCEED



Here our first activity failed and 2nd activity handles this error and proceeds with 3rd activity

2nd scenario



Here the first activity passes and 2nd activity gets skipped as it only runs when first activity fails...and we have connected SV2 and SV3 using skipped connector ..as SV2 is skipped SV3 executes

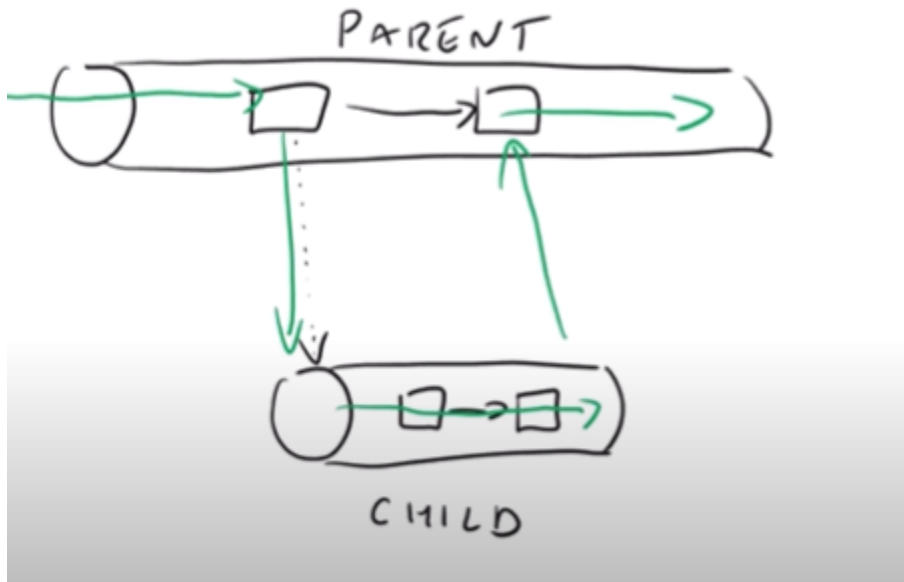
5. Generic Error Handling

This Pattern is most confusing...refer video

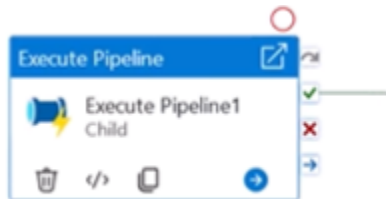
Parent Child Scenario

1. Assume we created two pipelines ..

2. Child has two activities and parent has 2 activities



3. When we execute the parent ...it goes to the child...execute all child activities and then execute remaining parent activities
4. Here we created child pipeline and added two activities of Set Variable (SV)
5. And created parent pipeline and inside that we have executed child using



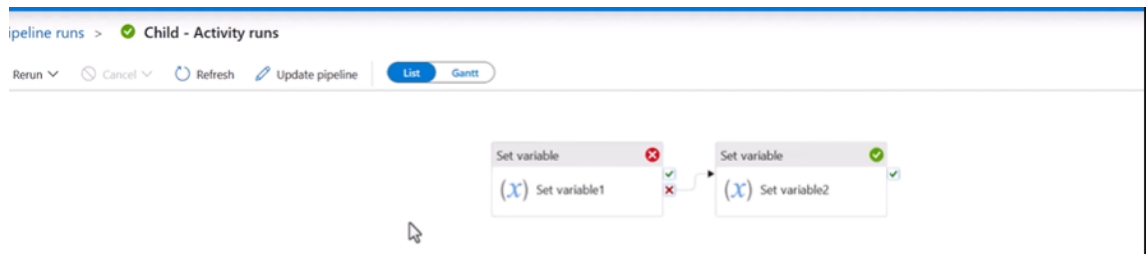
and created wait activity inside parent pipeline

6. To check the pipeline statuswe can see pipeline runs inside manage
7. Later we introduce a bug in child pipeline ...now when we execute parent pipeline

| Microsoft Azure Data Factory tybuladf | | | | | | |
|--|------------------------|------------------------|----------|-----------------------|-----------|--|
| Pipeline runs | | | | | | |
| <div> Triggered Debug Return Cancel options Refresh Edit columns List Gantt </div> | | | | | | |
| <div> Filter by run ID or name Local time: Last 24 hours Pipeline name: All Status: All Runs: Latest runs Triggered by: All </div> | | | | | | |
| Showing 1 - 4 items | | | | | | |
| Pipeline name | Run start | Run end | Duration | Triggered by | Status | |
| Child | 11/19/2023, 9:40:22 AM | 11/19/2023, 9:40:24 AM | 3s | 8047a205-eb3d-4f73... | Failed | |
| Parent | 11/19/2023, 9:40:21 AM | 11/19/2023, 9:40:25 AM | 5s | Manual trigger | Failed | |
| Child | 11/19/2023, 9:39:26 AM | 11/19/2023, 9:39:28 AM | 3s | d0ecae8c-24f0-40fd... | Succeeded | |
| Parent | 11/19/2023, 9:39:24 AM | 11/19/2023, 9:39:31 AM | 7s | Manual trigger | Succeeded | |

we see the both pipelines failed

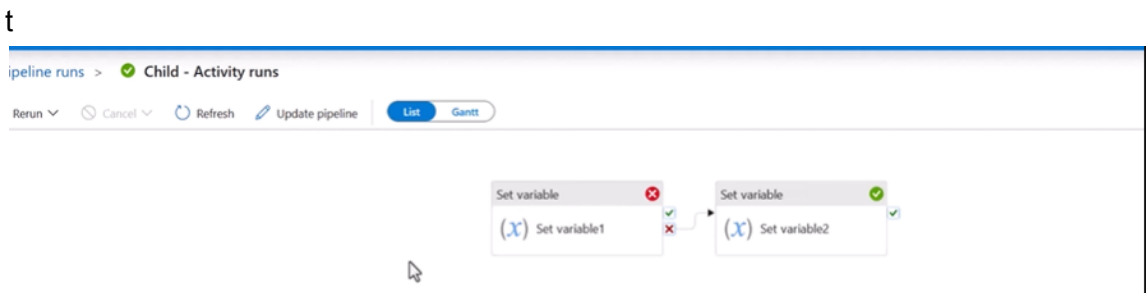
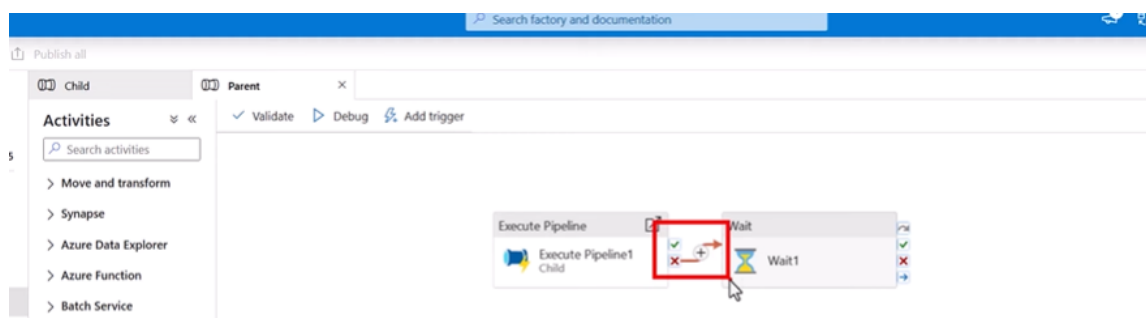
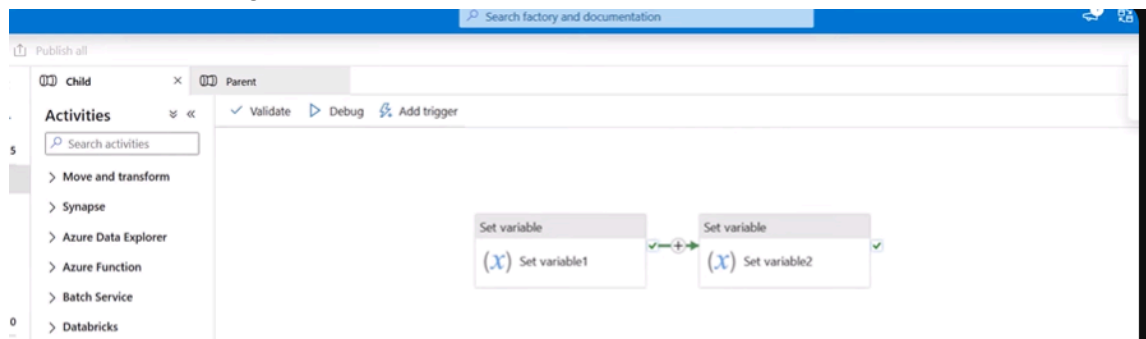
8. And if we handle the bug in the



and run the parent pipeline then we get success

9. Another scenario....

10. Here we do not handle the bug in child pipeline and handle the bug in parent pipeline...then we get..child as failed and parent as succeeded

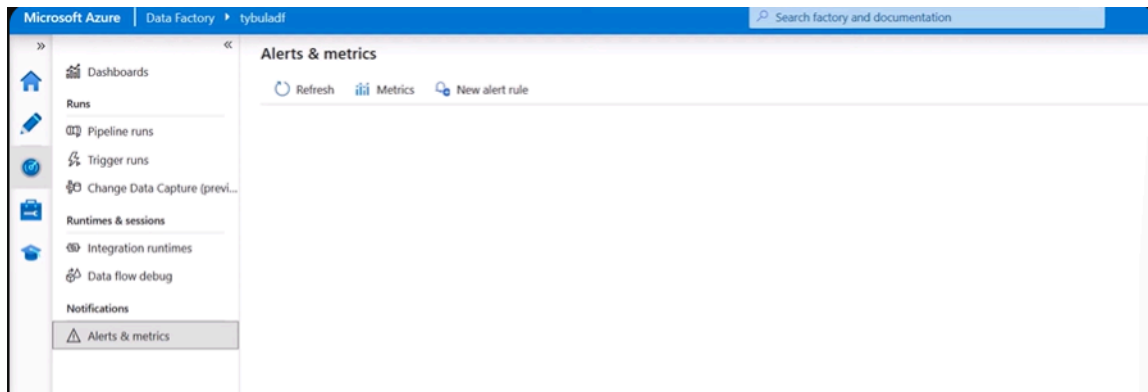


11. So by this we can track the error's and handle them in our ADF

Alerts

1. Here alerts will notify us when there are any issues in our ADF

2. To setup alerts we go to



3. Next we will create a new alert rule

The screenshot displays the 'New alert rule' configuration form. It includes the following fields and options: 'Alert rule name' with the value 'NewAlert'; an empty 'Description' text area; 'Severity' set to 'Sev2' in a dropdown menu; 'Target criteria' with an 'Add criteria' button; a notification message: 'There will be a monthly rate for the configured criteria. Learn more about Pricing'; 'Configure Email/SMS/Push/Voice notification' with a 'Configure notification' button; and 'Enable rule upon creation' which is a toggle switch currently turned 'On'.

here severity is

level of issue

4. Next we have to add criteria on when can this alert trigger

5. So here we can choose any issue

Add criteria

Select one metric to set up the alert condition.

Metrics ↑↓

Cancelled SSIS package execution metrics

Cancelled trigger runs metrics

Copy available capacity percentage of MVNet integration runtime

Copy capacity utilization of MVNet integration runtime

Copy waiting queue length of MVNet integration runtime

Elapsed Time Pipeline Runs Metrics

External available capacity percentage of MVNet integration runtime

External capacity utilization of MVNet integration runtime

External waiting queue length of MVNet integration runtime

Failed activity runs metrics

Failed pipeline runs metrics

- Next we have to select the pipelines and failure type

The screenshot displays a monitoring dashboard. At the top, a line graph shows a metric over time from 3AM to 8AM. The y-axis ranges from 0 to 2.0. The line is at 0 until 7AM, then rises linearly to 2.0 at 8AM. Below the graph, the word "Total" is displayed. A text instruction reads: "Selecting the dimension values will help you filter to the right time series." Below this, there are two dropdown menus: "Name" with "5 selected" and "FailureType" with "3 selected". An "Alert logic" section contains a "Condition" dropdown set to "Greater than", a "Time aggregation" dropdown set to "Total", and a "Threshold count" input field set to "0". A list of failure types is shown with checkboxes: "Select all", "UserError", "SystemError", and "BadGateway", all of which are checked. At the bottom, there is an "Evaluate based on" section with a "Period" dropdown. Navigation buttons "Add criteria", "Back", and "Cancel" are at the bottom.

| Dimension | Values |
|-------------|------------|
| Name | 5 selected |
| FailureType | 3 selected |

Alert logic

Condition * ⓘ
Greater than

Time aggregation * ⓘ
Total

Threshold count * ⓘ
0

Evaluate based on

Period * ⓘ

☒ Select all
☒ UserError
☒ SystemError
☒ BadGateway

- If threshold is greater than 0...then this alert gets trigger
- After creating this criteria...we have to configure the notification on how to notify

9. So if any of our pipelines fails..then we get an email alert

Alert Activated Because:

| | |
|------------------|--|
| Metric name | PipelineFailedRuns |
| Metric namespace | factories/tybuladf |
| Dimensions | ResourceId = /SUBSCRIPTIONS/1C461DD7-4FB5-4D22-B200-E052F75B9C6F/RESOURCEGROUPS/DP-203/PROVIDERS/MICROSOFT.DATAFACTORY/FACTORIES/TYBULADF Name = Child FailureType = UserError |
| Time Aggregation | Total |
| Period | Over the last 1 mins |
| Value | 2 |
| Operator | GreaterThan |
| Threshold | 0 |
| Criterion Type | StaticThresholdCriterion |

[See in the Azure portal >](#)

You're receiving this notification as a member of the alert action group.
[Unsubscribe](#) from emails directed to this group.

[Facebook](#) [Twitter](#) [YouTube](#) [LinkedIn](#)

[Privacy Statement](#)

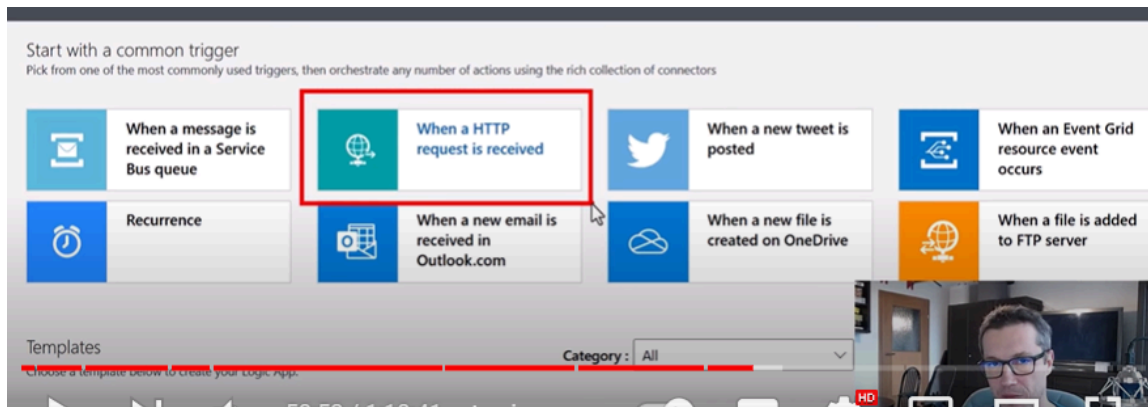
Microsoft Corporation, One Microsoft Way, Redmond, WA 98052
Microsoft

10. It will be very useful if we have 1000's of pipeline

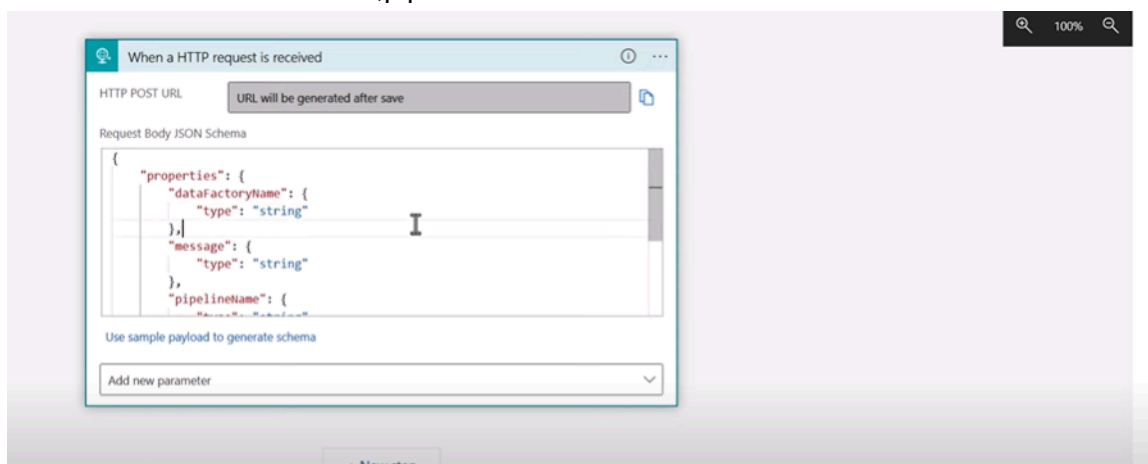
Logic App

1. Here ADF does not have activity to send email alert
2. So we have to use other Azure Service to send email alert
3. Next we create a logic app with defaults

4. Then we have to choose when should our logic app triggers



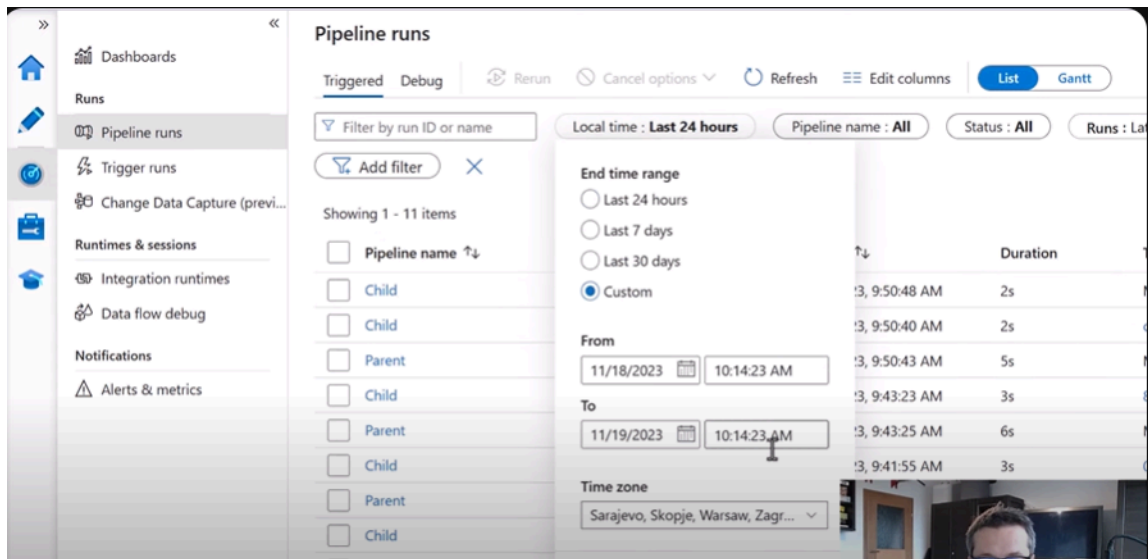
5. Next we define the adf name, pipeline name



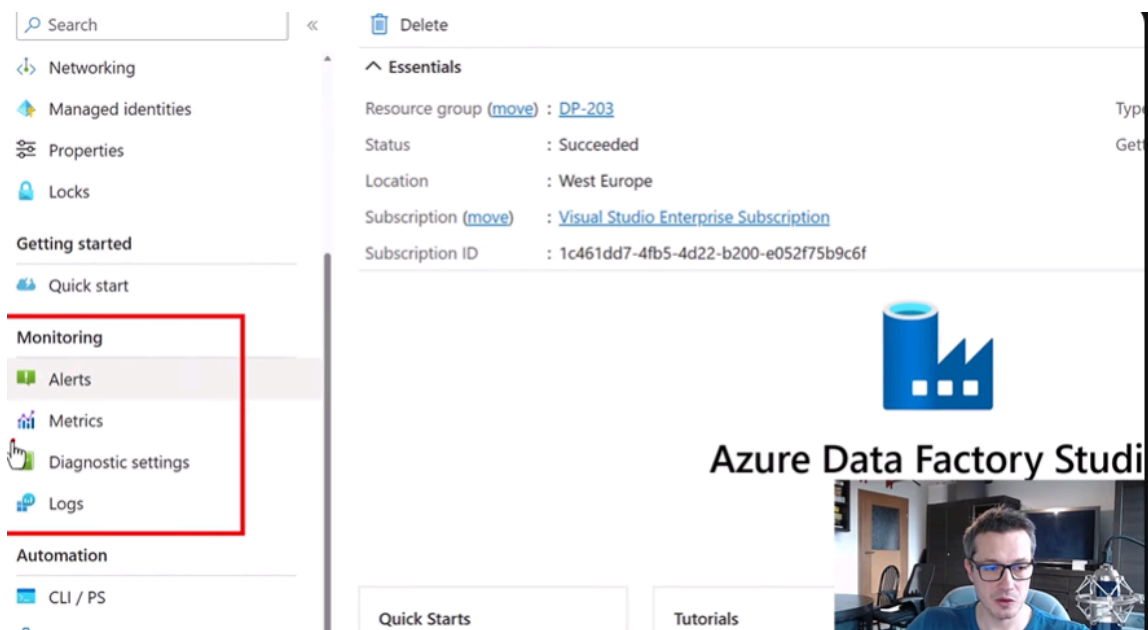
6. Next we need to use logic app connector and connect with outlook mail...now this email will send the alert to use
7. So when we use logic app...we can create custom email template and send it via alerts to the user's..if there's any issue(errors) with ADF pipelines

Log Analytics Approach

1. In ADF our pipelines logs are only stored for a max period of 45days



2. So to analyze the logs we might need to store them for a longer period of time...to perform any analysis
3. To store the logs for longer periods..we can use diagnostic settings
4. So here we can make use of log analytic workspace to store logs for longer period of time
5. Practical
6. Search log analytics workspace and create one
7. In the monitor tab we can see



diagnostic setting

8. Next we have to choose the type of logs that log analytics stores and the destination(log analytics workspace)

Diagnostic setting ...

 Save  Discard  Delete  Feedback

A diagnostic setting specifies a list of categories of platform logs and/or metrics that you want to collect from a resource, and one or more destinations that you would stream them to. Normal usage charges for the destination will occur. [Learn more about the different log categories and contents of those logs](#)

Diagnostic setting name * ✓

Logs

Category groups ⓘ

☐ allLogs

Categories

☐ Pipeline activity runs log

☐ Pipeline runs log

☐ Trigger runs log

Destination details

☐ Send to Log Analytics workspace

☐ Archive to a storage account

☐ Stream to an event hub

☐ Send to partner

Destination details

☒ Send to Log Analytics workspace

Subscription

Visual Studio Enterprise Subscription

Log Analytics workspace

LogWorkspace (westeurope)

Destination table ⓘ

Azure diagnostics

Resource specific

☐ Archive to a storage account

9. Also we can choose the retention period for this logs inside log analytic workspace ...default is 30days...can be extended upto 730days.
10. And logs inside log analytics workspace can be queried using KQL language
- 11.

