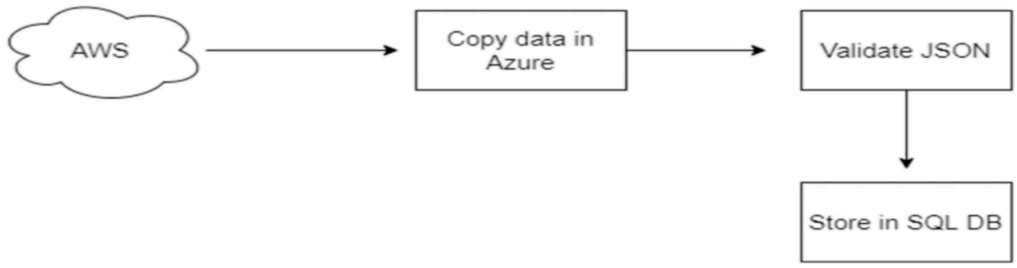
Project 1 Connected vehicles

General Motors is one of the leading heavy vehicle manufacturing company. To improve their services they are planning to rollout lot new features based on IoT.

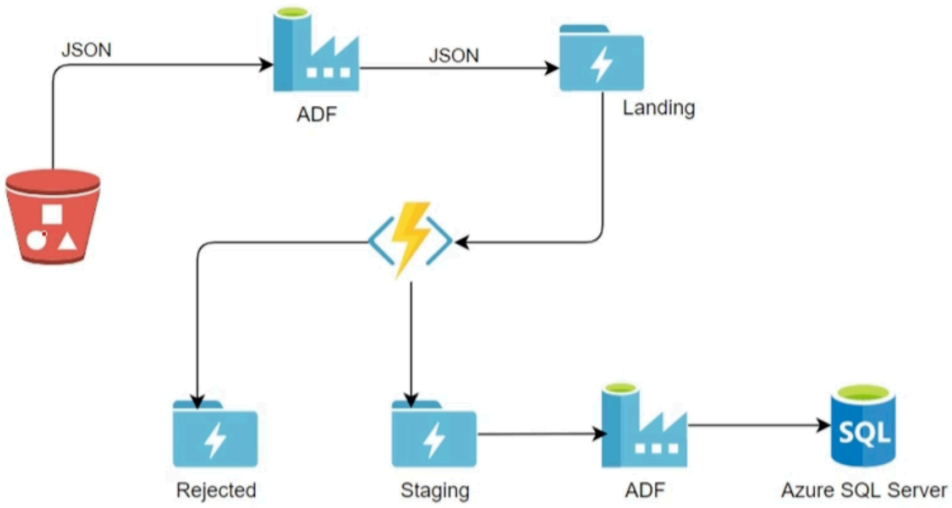
Vehicle has third party IoT device which will send the telemetry data (in JSON format) over the AWS cloud.

Data Engineer tasks:

1). Move data from third party AWS to General Motors Azure cloud
2). Validate the JSON (sometimes it could be incomplete or wrong JSON which need to be rejected)
3). After validation, store the data in SQL database



4). Design solution



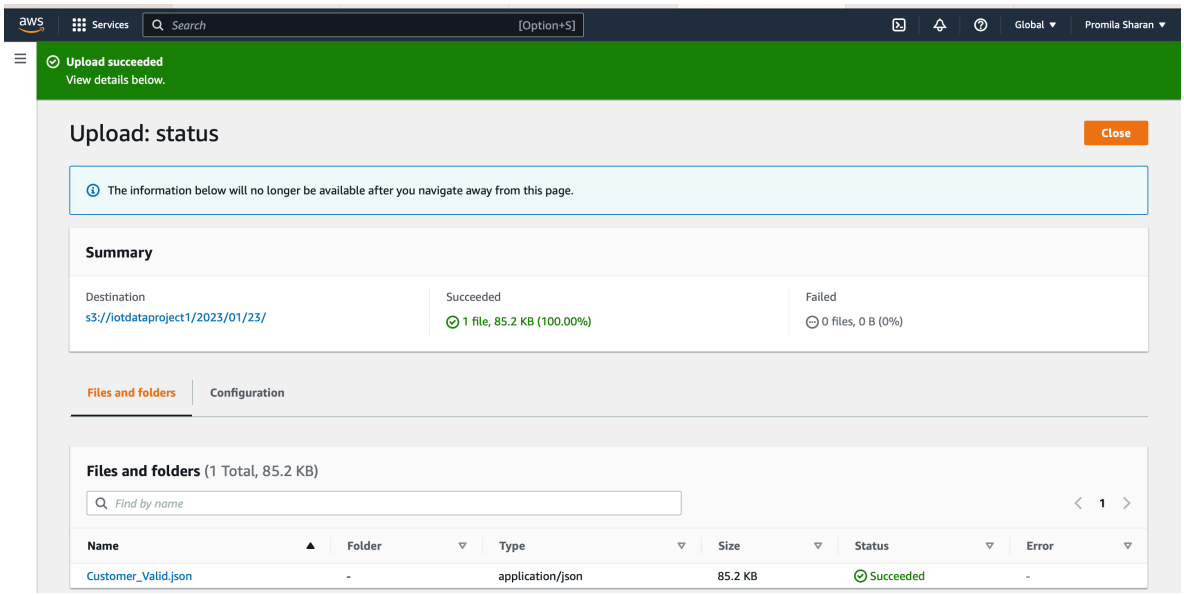
Architecture Diagram for Connected Vehicle Project

Input source: AWS cloud s3 storage bucket where JSON telemetry data is stored
We can use azure data factory for moving the data from one source to another source and move the data in landing folder

Then we can validate the data (JSON format) using azure function. We can use storage based trigger on the Azure function. As soon as file arrive in landing folder, azure function will trigger automatically. This azure function will read the file and just validate whether its in proper JSON format. If it is in valid format then it'll be moved in Staging folder, otherwise in Rejected folder. Once we have data in Staging folder, we can copy this data from Staging Folder to Azure SQL database using Azure Data Factory

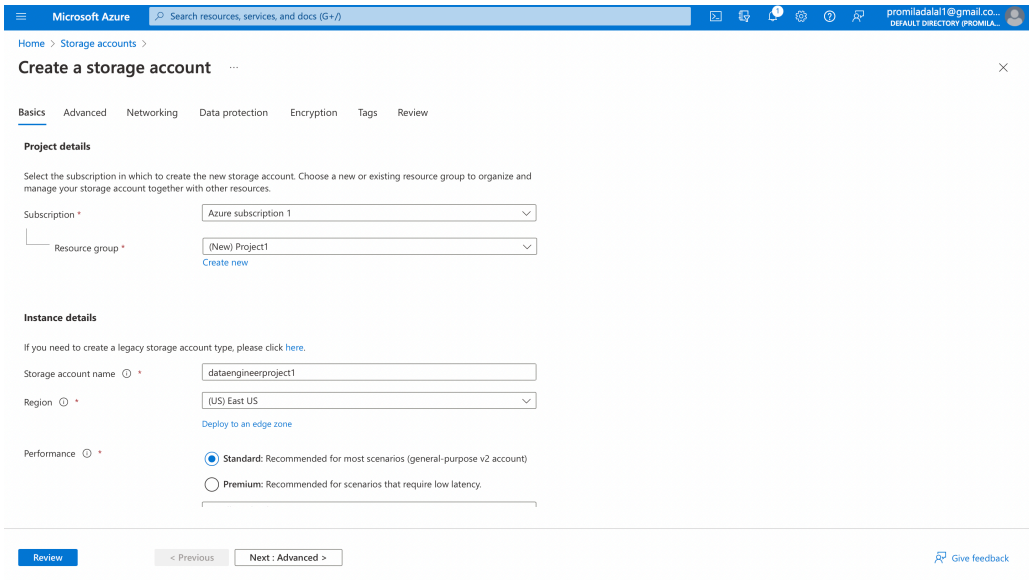
Implementation of Project:

- Create a free AWS account ((https://portal.aws.amazon.com)
- Create an S3 bucket in AWS to store data from connected vehicles' IoT devices
- Upload a sample data file into the S3 bucket, following a specific folder format for date organization
- The data will be sent on a daily basis and saved in the S3 bucket with a year, month, and day format



- The goal is to create an ADF to pull data from this S3 bucket.

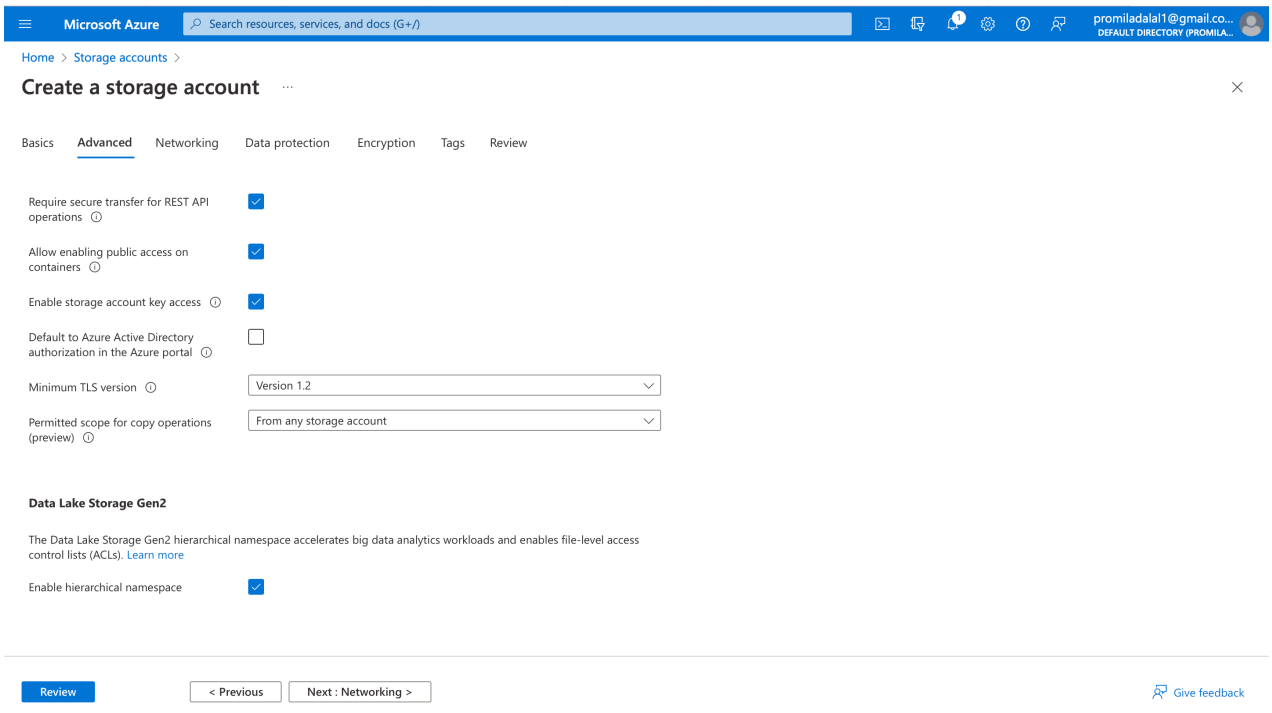
1. Create Azure data storage account



2.Select

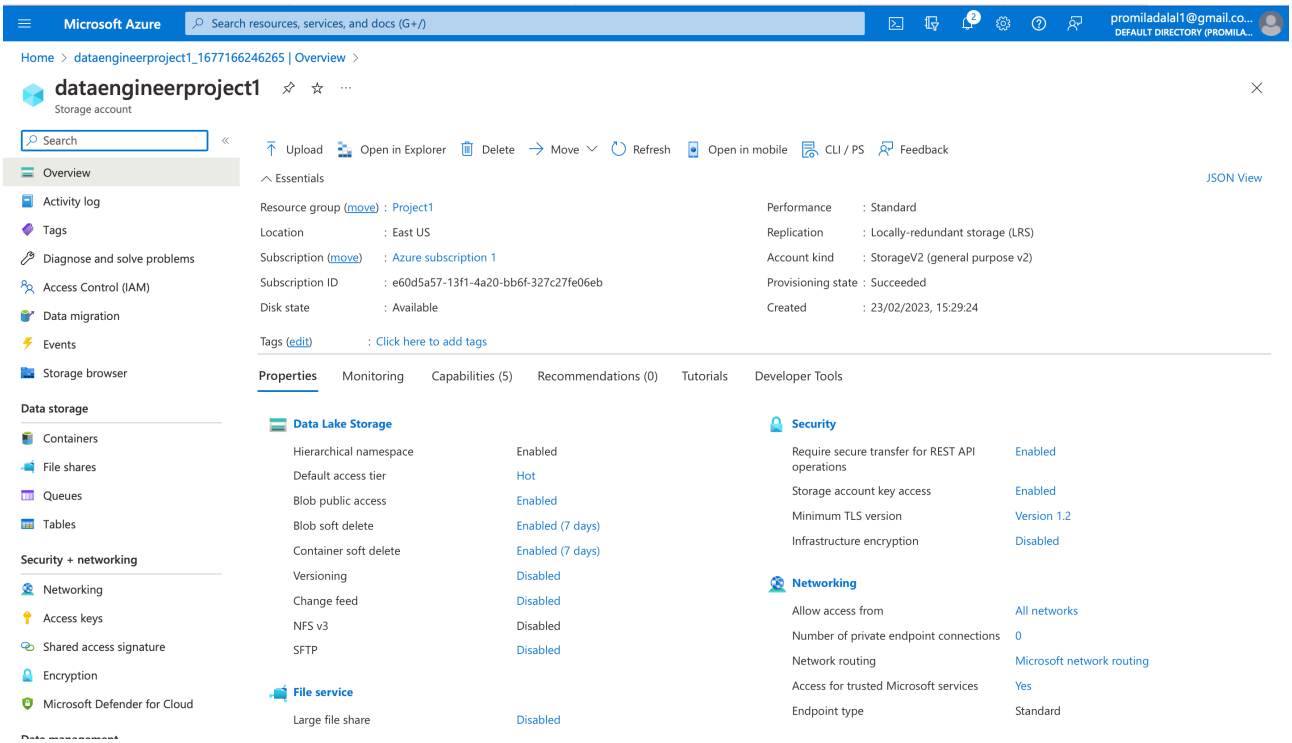
subscription and resource group.

3. Enter unique storage account name and select standard performance.



If we enable hierarchical namespace then this account is converted to adls gen2 account.

4. Click "Review + Create" and wait for storage account creation.
5. Click on "Go to resource" to access storage account.
6. In the storage account go to the “Data Lake Storage”

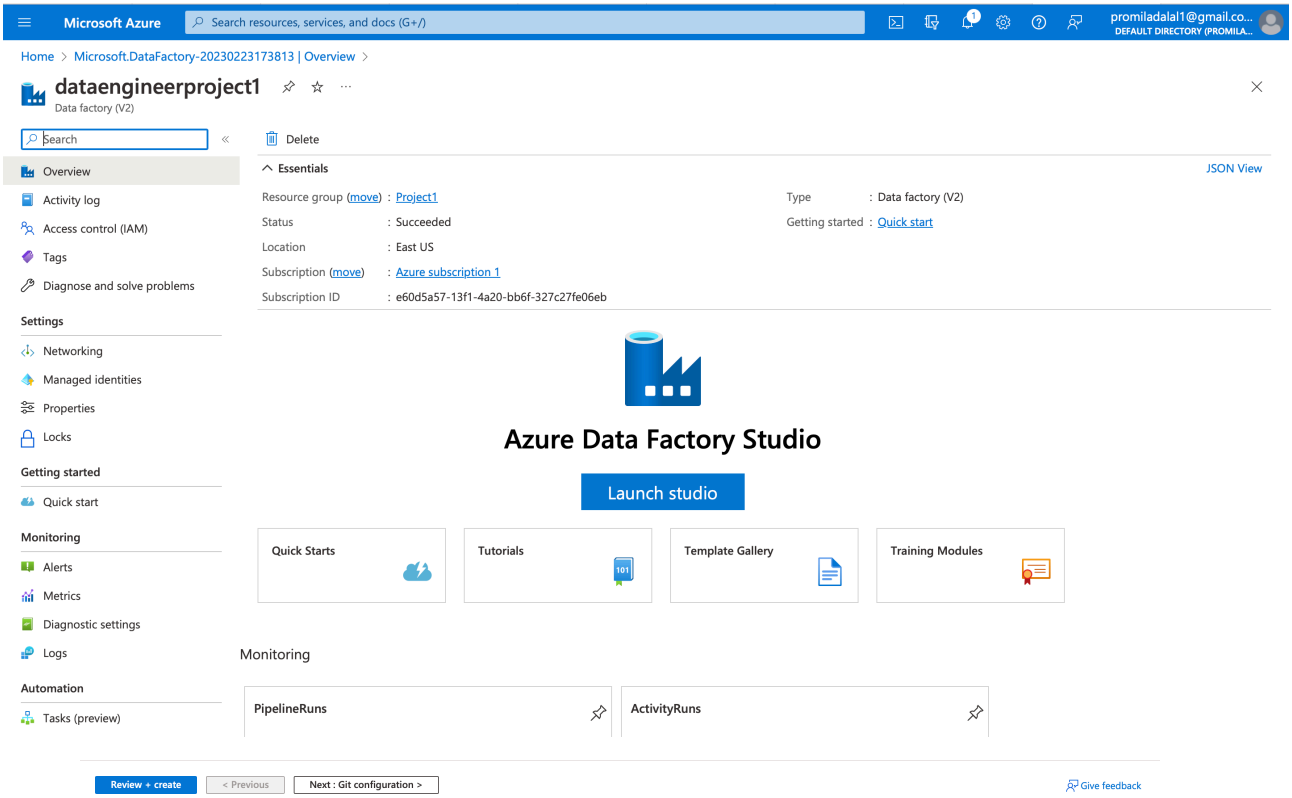


7. Create a container named "input" inside the storage account.

8. Create a folder named "landing" inside the "input" container.
9. "landing" folder will be the landing folder for all files transferred from AWS.

Create Azure Data Factory Account Setting Up for Data Transfer:

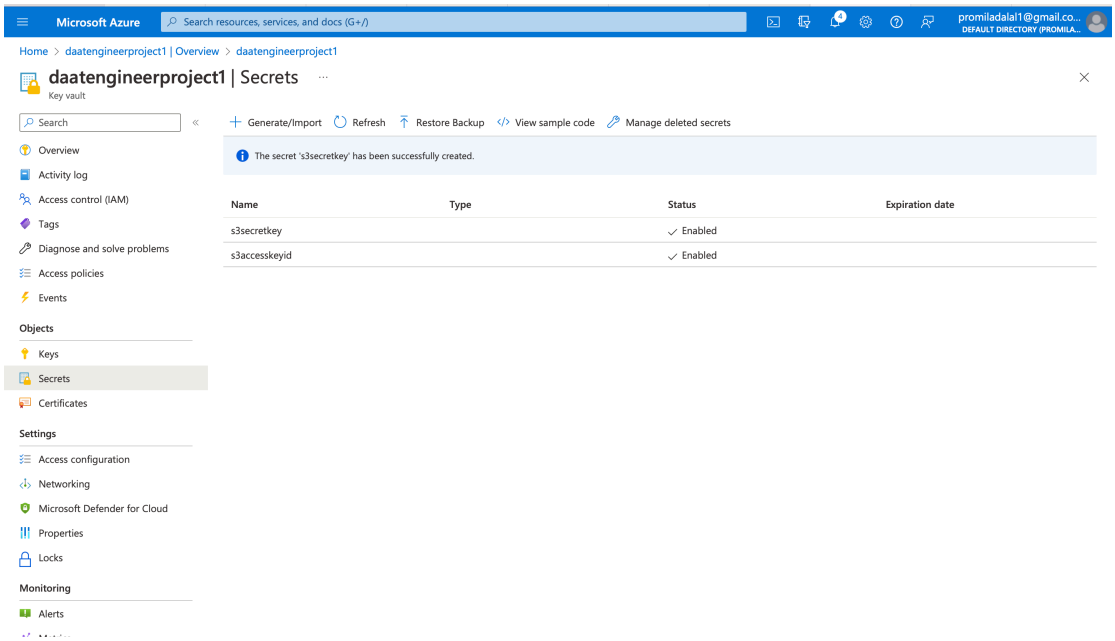
1. Creating a new ADF account in the Azure portal
2. Selecting subscription, resource group, region, and giving a suitable name to the account
3. Skipping Git configuration for now and keeping networking settings default
4. Reviewing the settings and creating the ADF account
5. Accessing the ADF account and opening up the Azure Data Factory studio to create pipelines



Create Azure Key Vault & Add the secretes: (Create a new Azure Key Vault to store Access Keys and Access ID.Use the Azure Key Vault to manage and store credentials and secrets):

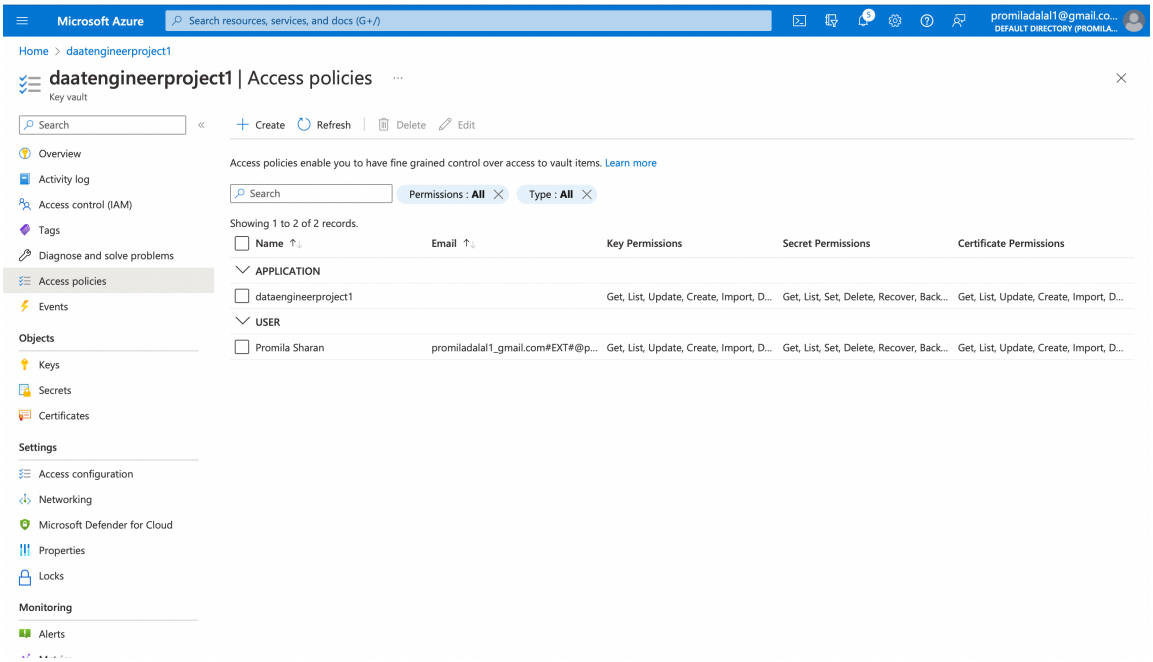
- Store all the secrets within the Azure Key Vault to allow easy access to Azure portal.
- Create a new Azure Key Vault by selecting a subscription and resource group and giving it a unique name.
- Keep all default settings and create the Azure Key Vault.
- Once the Azure Key Vault is created, store the other credential in the secrets of the Azure Key Vault.
- Go to the secrets and click on Generate.
- Copy the Access Key and Access Secret Key for AWS (go to AWS Services —> click on IAM —> click on “Manage Access Keys” —> Create New Access Keys —> copy the Access Key ID —> go to KeyVault & paste the Access Key ID in value -> & Secret Access Key.
- Create a secret for each key by pasting the Access Key and Access Secret Key in the Azure Key Vault.

- Assign access policies to the Azure Key Vault by clicking on Add Access Policy and



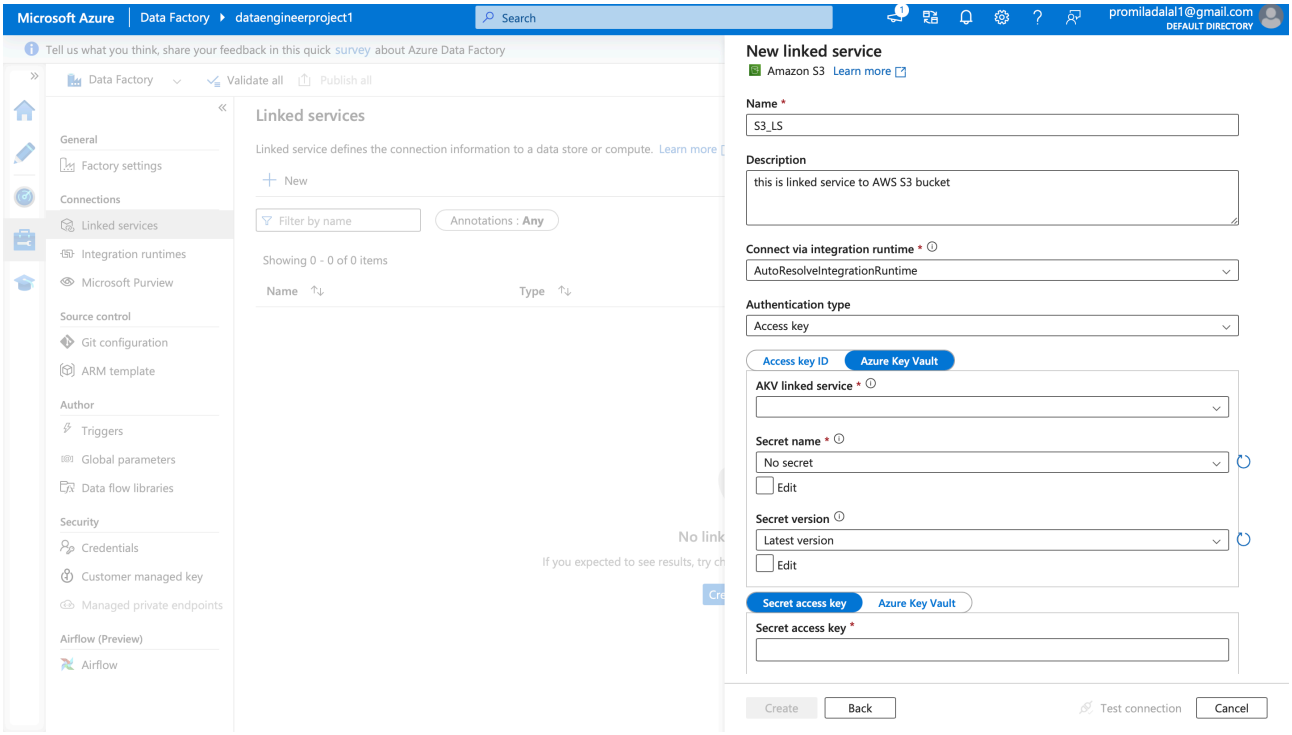
selecting the Azure Data Factory account as the principal.

- Verify that the keyboard is accessible by the user and the Azure Data Factory account.
- Use the stored Access Key and Access Secret Key to connect to AWS S3 account in a pipeline.

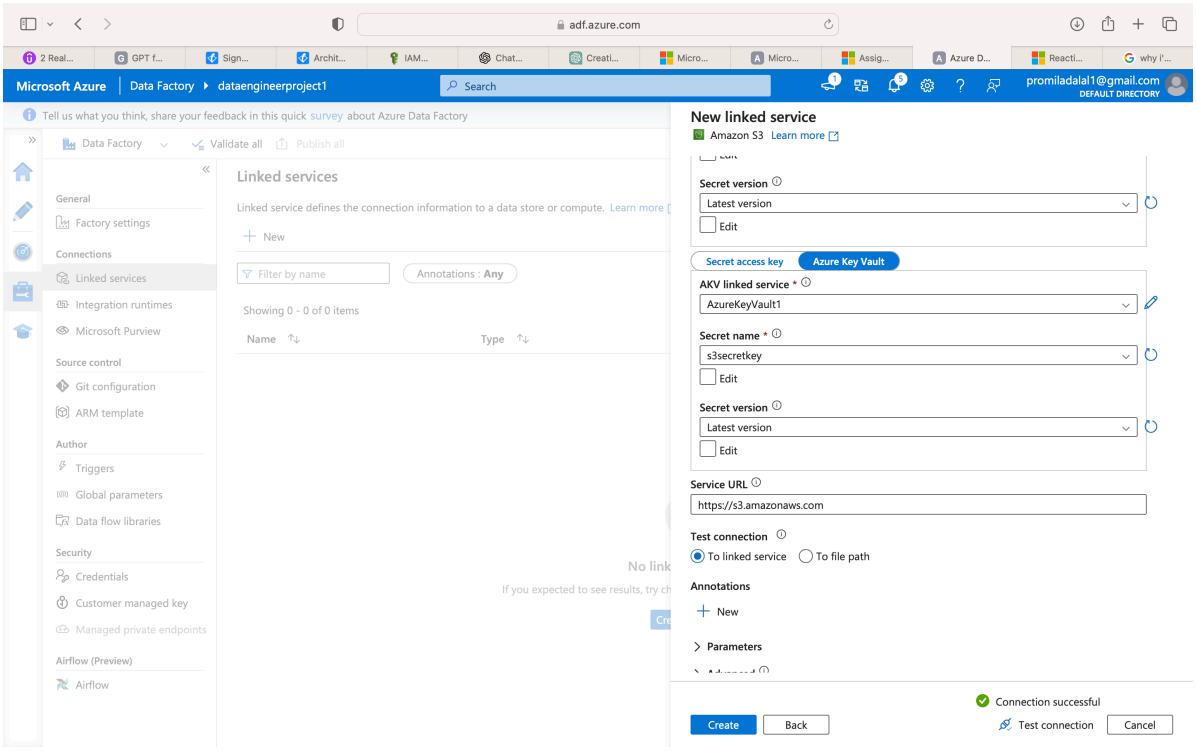


Create pipeline for moving the data from AWS s3 bucket to landing folder in azure & add trigger to run on daily basis:

1). In Azure Data Factory, go to the author tab and clicked on “Linked services” for creating link service for connecting to AWS S3 (where data is stored by the third party device; Source of the data).



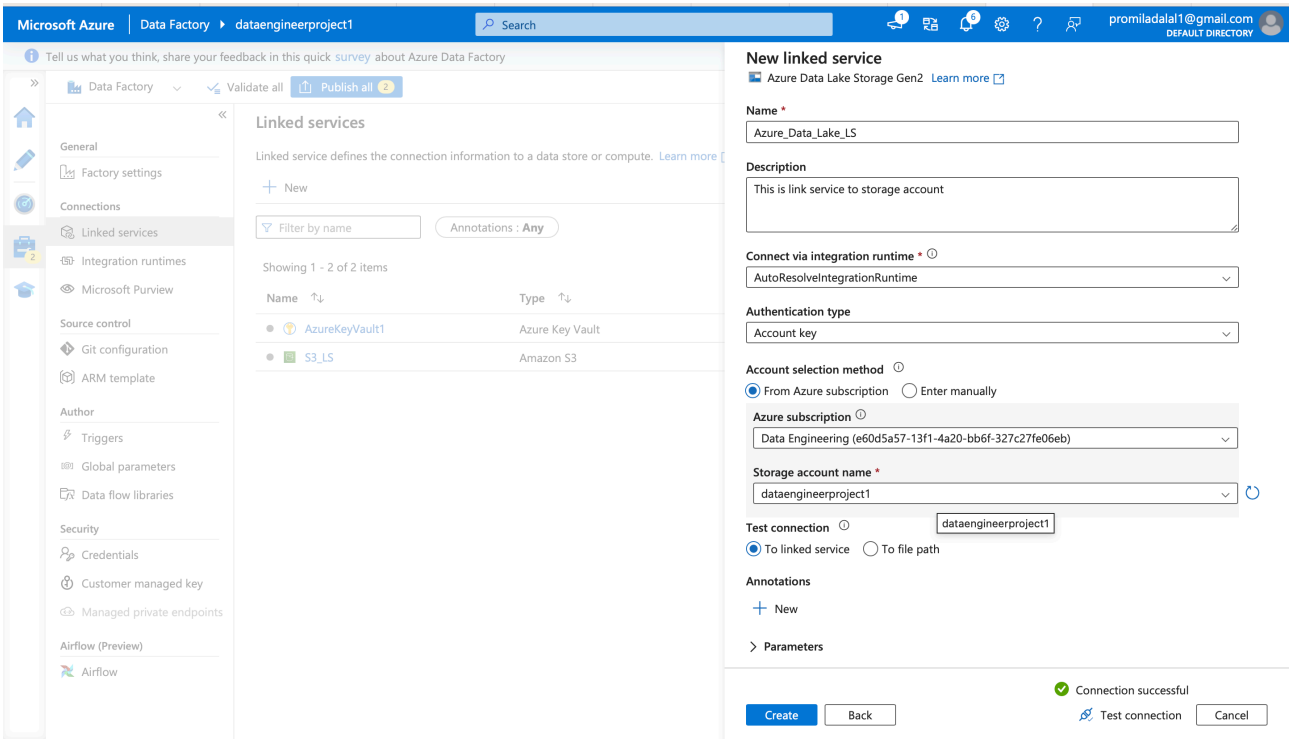
In this we’ll use the credentials using Azure Key Vault where we stored secret key & secret key id. But before that we need to creat AKV linked service which we can utilise also for creating linked



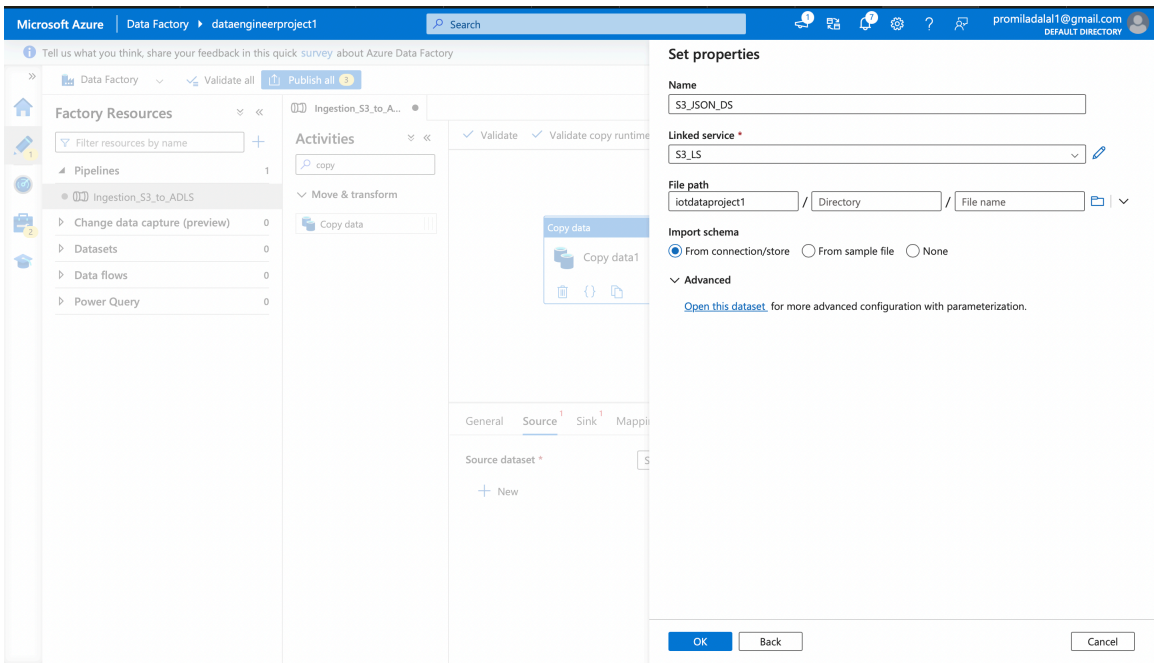
service for connecting to ADLS. We'll test connection, if it''s successful then we can see it at the bottom right corner "Connection successful"

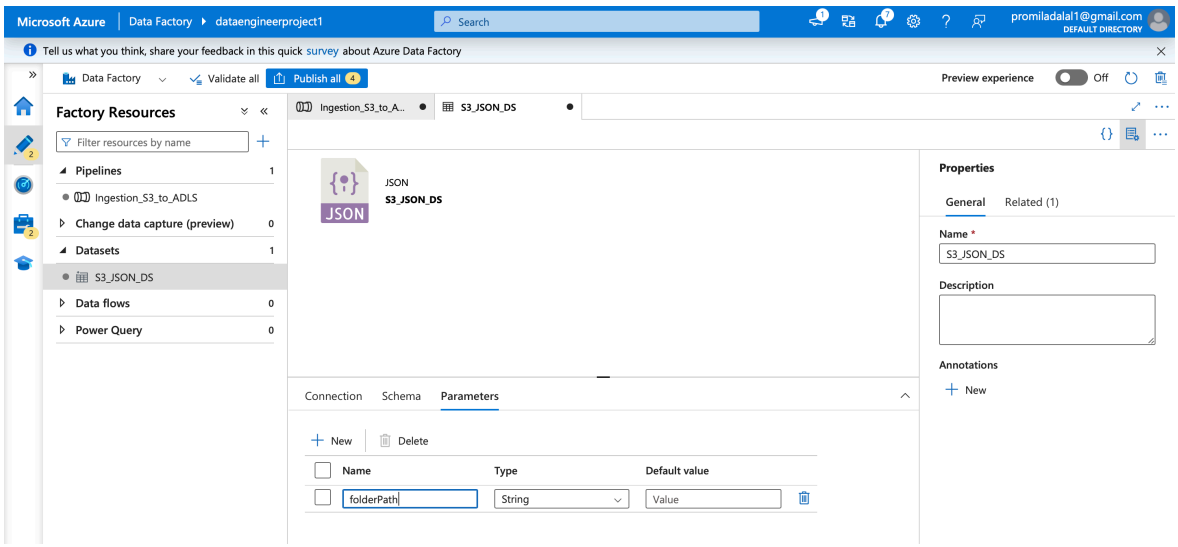
2). Connect to Azure Data Lake Gen2 Storage

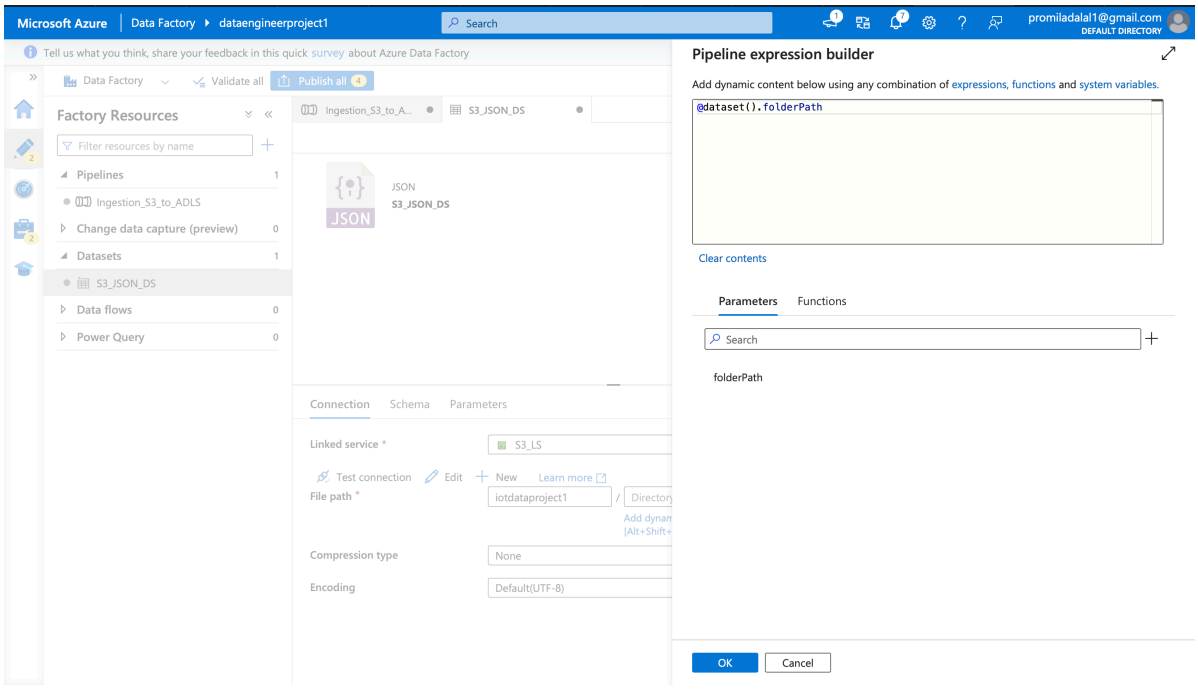
In a similar way we'll create linked service for making connection of Azure Data Factory with Azure Data Lake



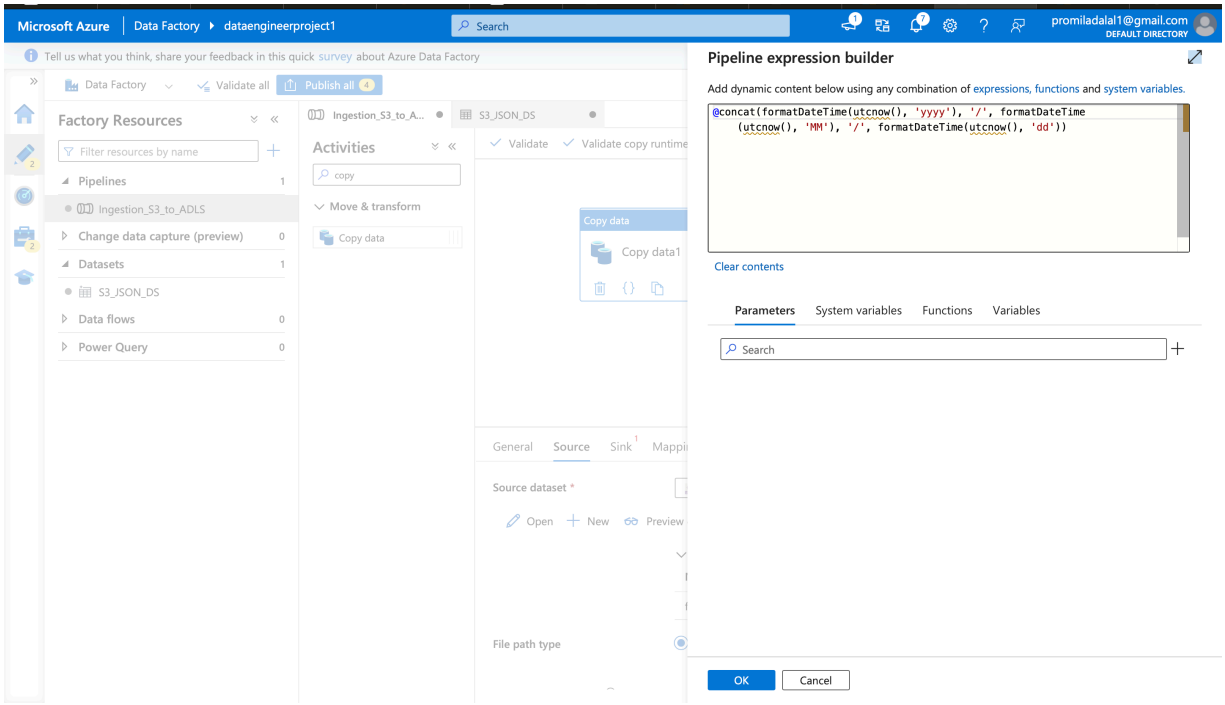
3). Create pipeline for pulling the data from AWS S3 bucket to Azure Data Lake Gen2 . We'll name the pipeline "Ingestion_S3_to_ADLS". For this we'll use Copy activity. We need to provide source & sink. When we click on source (at this moment we don't have any dataset available so e click on "+" for creating the dataset and the source side. We know that Amazon S3 as our source, so we have to create the dataset for Amazon S3, so select S3 and file format to JSON. Then we have to select the link service, which we had created so select that. When we click on browse-it will allow us to select the root folder that is in S3 bucket, select root folder. In our case, data folder is in format YYYY-MM-DD hierarchy. If folder path is changing on daily basis and we need to run this pipeline on daily basis as well then we have to make a pipeline in such a way that'll dynamically pull the data from dynamically selected foldersn the If we want to make a pipeline that'll dynamically pull the data form dynamically selected foldersPath. We can do this by parametrising the dataset
Go to the parameter & give the name"folderPath" as dynamic parameter —> go to the connection —> Directory & add dynamic content—>select parameter which is folderPath


Microsoft Azure
Data Factory
dataengineerproject1
Search
promiladalal1@gmail.com
DEFAULT DIRECTORY
Tell us what you think, share your feedback in this quick survey about Azure Data Factory
Data Factory
Validate all
Publish all
Factory Resources
Filter resources by name
Pipelines
Ingestion_S3_to_ADLS
Change data capture (preview)
Datasets
Data flows
Power Query
Activities
copy
Move & transform
Copy data
Validate
Validate copy runtime
Copy data1
General
Source
Sink
Source dataset *
New
Set properties
Name
S3_JSON_DS
Linked service *
S3_LS
File path
iotdataproject1
Directory
File name
Import schema
From connection/store
From sample file
None
Advanced
Open this dataset for more advanced configuration with parameterization.
OK
Back
Cancel

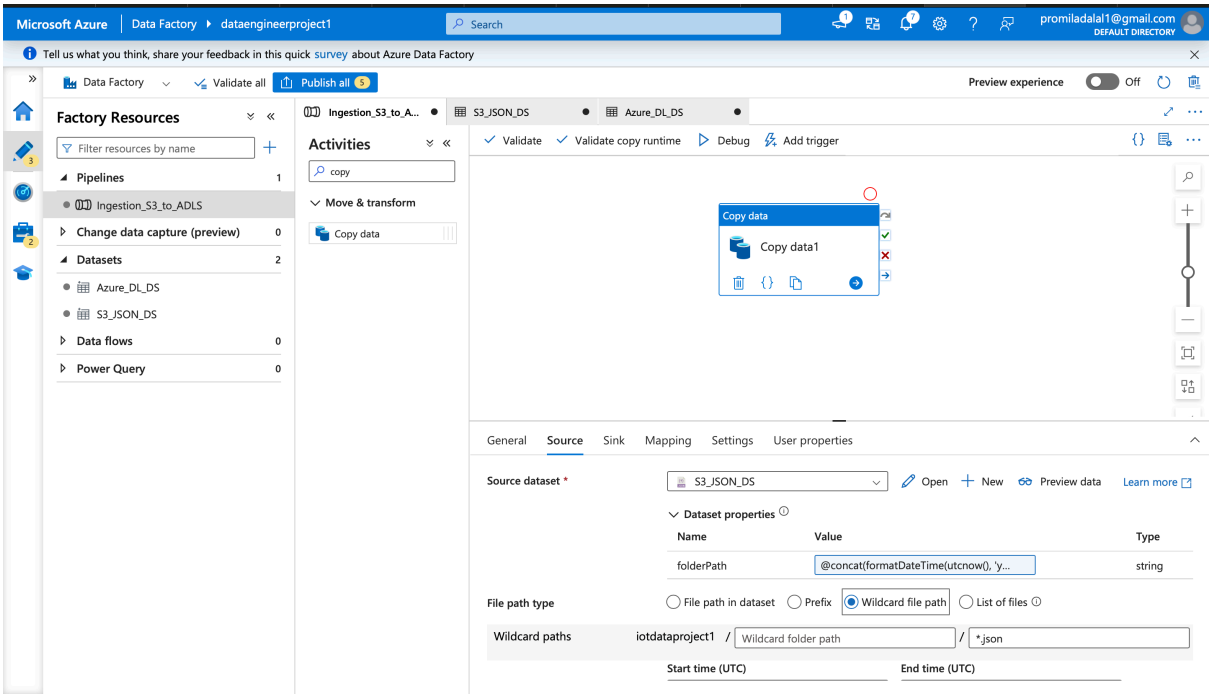

Microsoft Azure
Data Factory
dataengineerproject1
Search
promiladalal1@gmail.com
DEFAULT DIRECTORY
Tell us what you think, share your feedback in this quick survey about Azure Data Factory
Data Factory
Validate all
Publish all 4
Preview experience
Off
Factory Resources
Filter resources by name
Pipelines
Ingestion_S3_to_ADLS
Change data capture (preview)
Datasets
S3_JSON_DS
Data flows
Power Query
Ingestion_S3_to_A...
S3_JSON_DS
JSON
S3_JSON_DS
Properties
General
Related (1)
Name *
S3_JSON_DS
Description
Annotations
New
Connection
Schema
Parameters
New
Delete
Name
Type
Default value
folderPath
String
Value


Microsoft Azure
Data Factory
dataengineerproject1
Search
promiladalal1@gmail.com
DEFAULT DIRECTORY
Tell us what you think, share your feedback in this quick survey about Azure Data Factory
Data Factory
Validate all
Publish all 4
Factory Resources
Filter resources by name
Pipelines
Ingestion_S3_to_ADLS
Change data capture (preview)
Datasets
S3_JSON_DS
Data flows
Power Query
Ingestion_S3_to_A...
S3_JSON_DS
JSON
S3_JSON_DS
Connection
Schema
Parameters
Linked service *
S3_LS
Test connection
Edit
New
Learn more
File path *
iotdataproject1
Directory
Compression type
None
Encoding
Default(UTF-8)
Pipeline expression builder
Add dynamic content below using any combination of expressions, functions and system variables.
@dataset().folderPath
Clear contents
Parameters
Functions
Search
folderPath
OK
Cancel

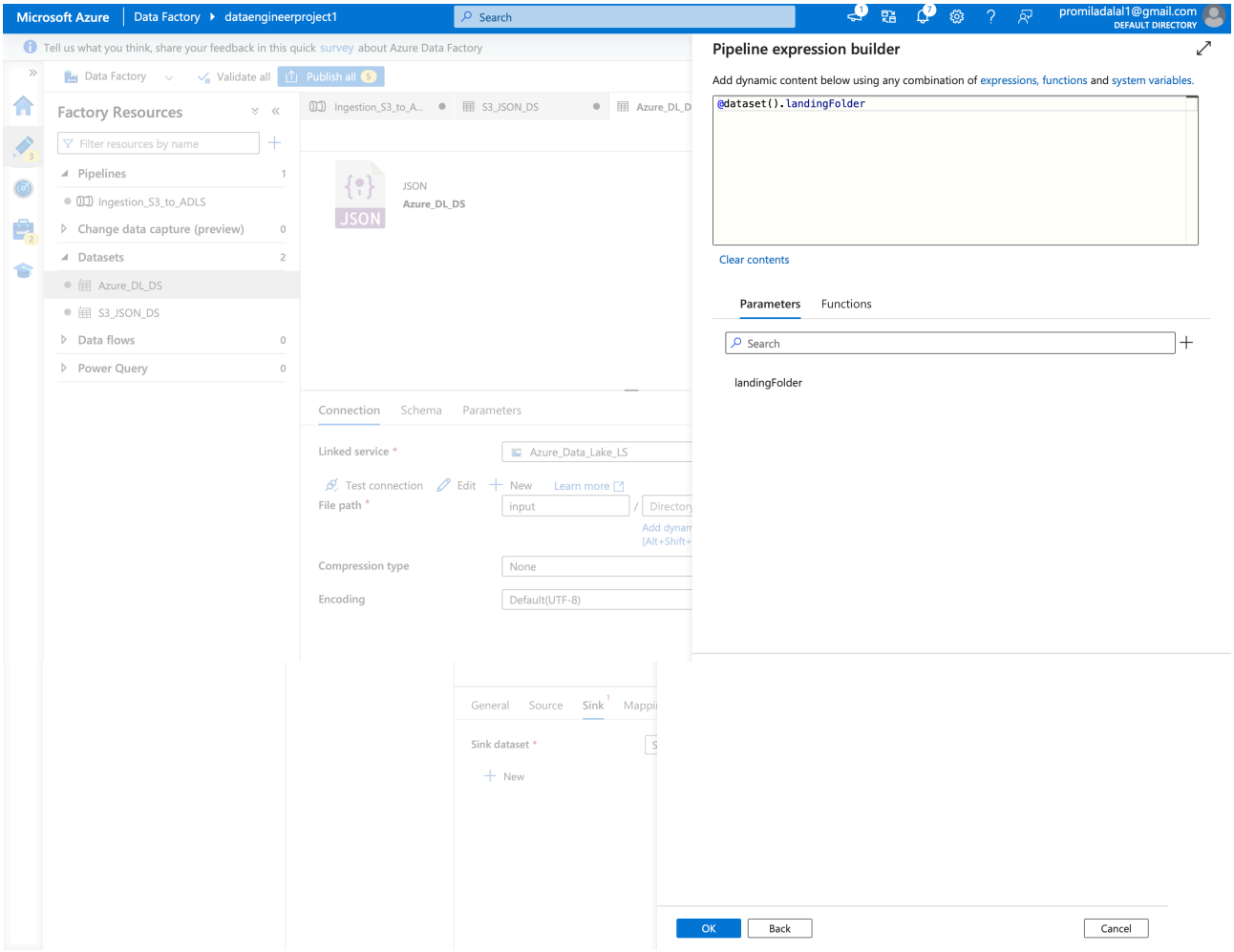
Whole path will be created dynamically based on parameter value which we pass in folder path. Now we'll go the pipeline and we can see that in source we're getting the option of folderPath. Now we know that we have to create the folder path in YYYY-MM-DD format so for that we'll use formatDateTime function. For ex. If we give this function:

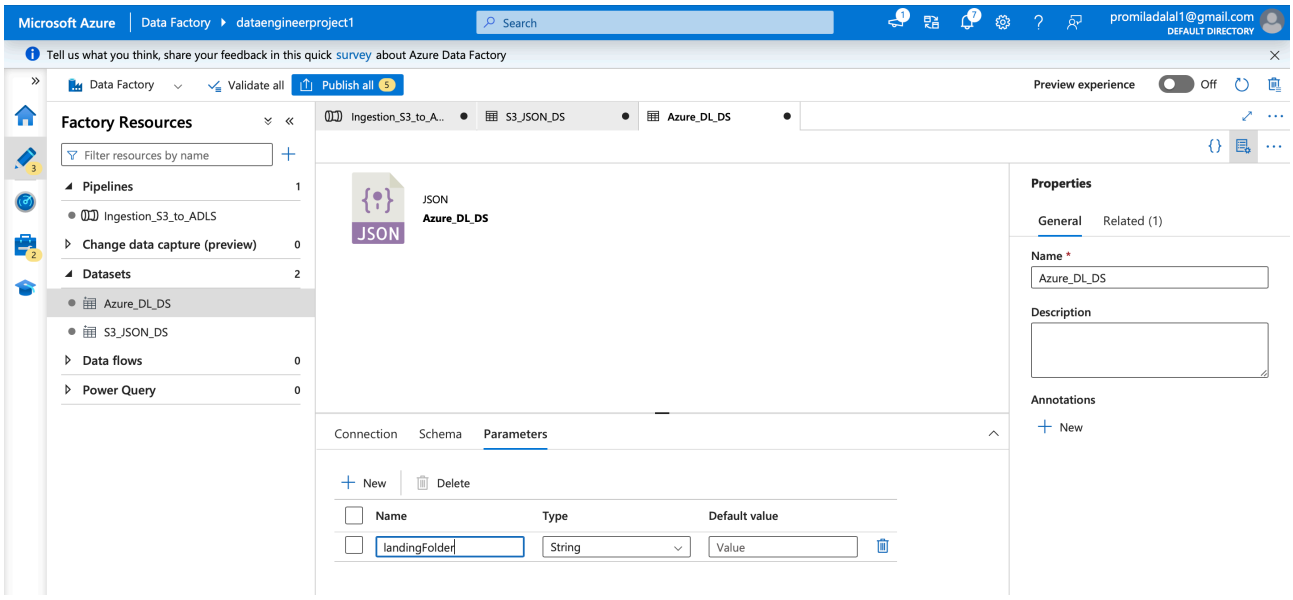


It'll extract yyyy MM & DD from the date. We'll use this to create our dynamic folder and then we'll use the concat function to concatenate all of them.



In

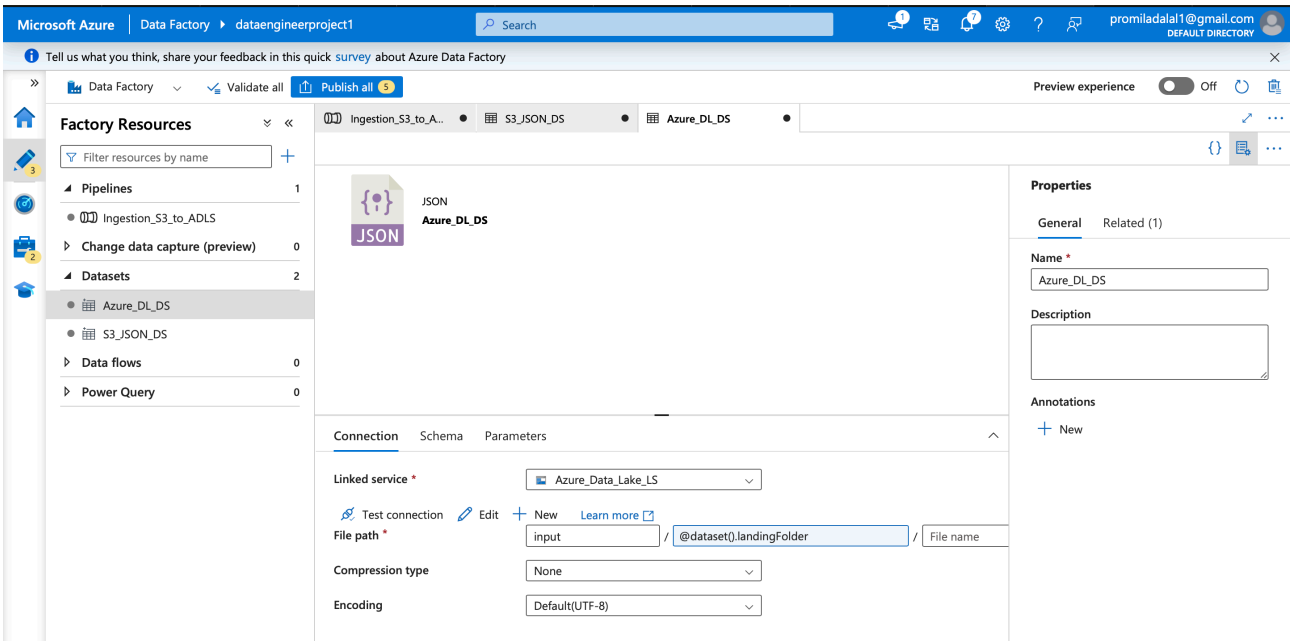


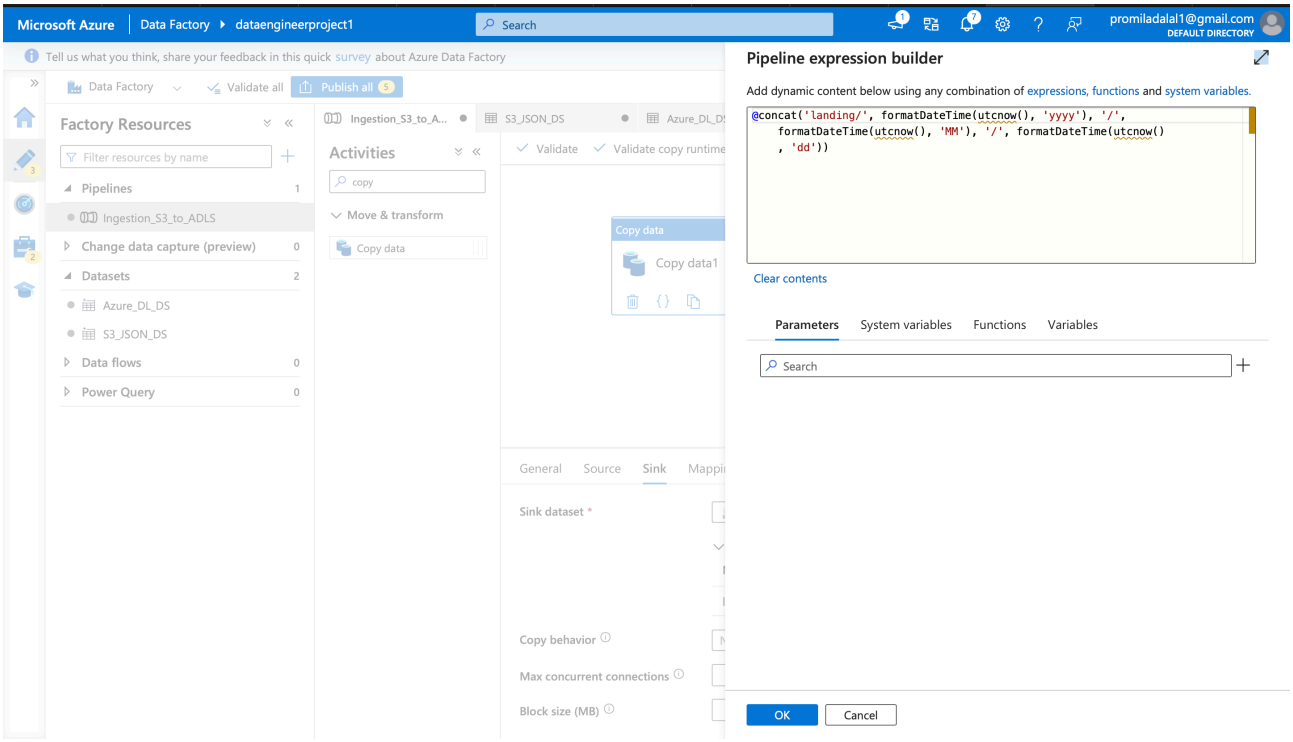


the source, if we have to pull multiple files then we'll select wildcard file path. It'll pull all files which are in json format

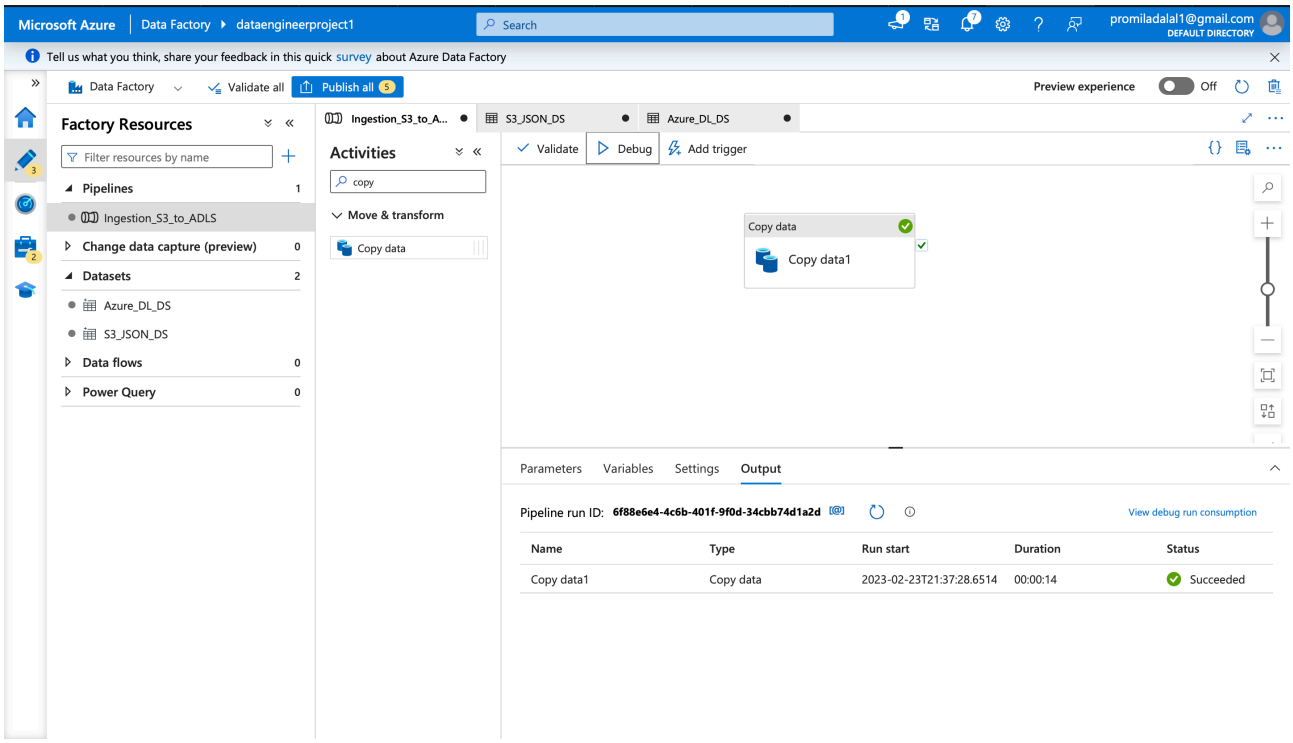
Now go to the "sink" (which is ADLS)—> create sink dataset—> in new dataset search for Azure Data Lake Storage Gen2 —> select the file format JSON —> select the name & select link service —> create folder structure—> just select input folder for dynamic creation of folder (same procedure as we followed for source)

So whatever value we give to this dynamic parameter, our file path will be selected accordingly For creating the folder structure in the similar pattern like YYYY-MM-DD, we'll use concat function. First we'll give the path as 'landing'/yyyy/mm/dd

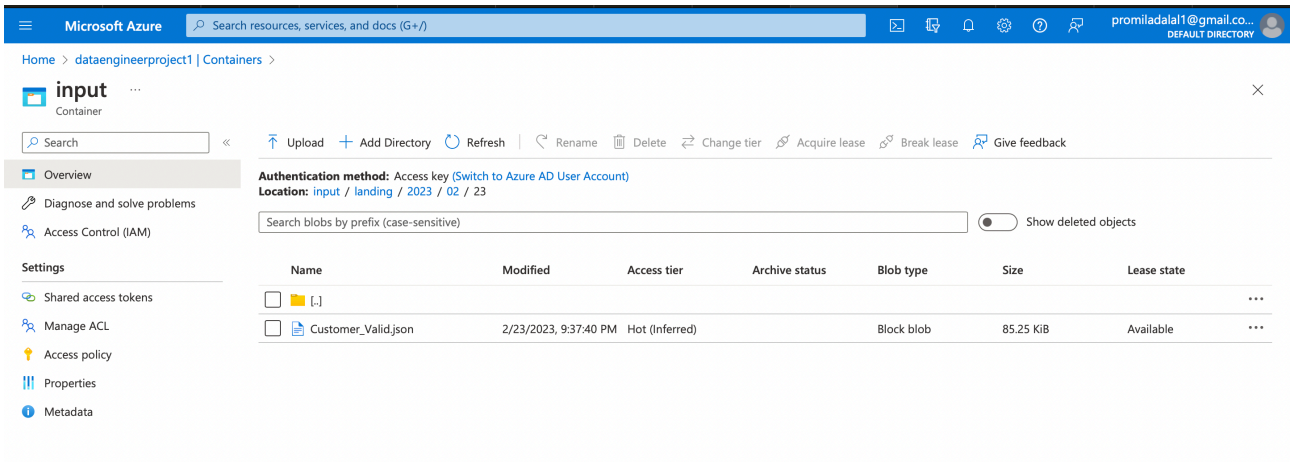




Now, our pipeline is almost ready, so we'll click on "debug". It'll only pull the data from the current date. We can see pipeline run is successful. We can check whether our data have been copied from the S3 bucket to Azure data lake storage or not. We can see that data has been copied from the AWS



S3 bucket to Azure Data lake get 2 container

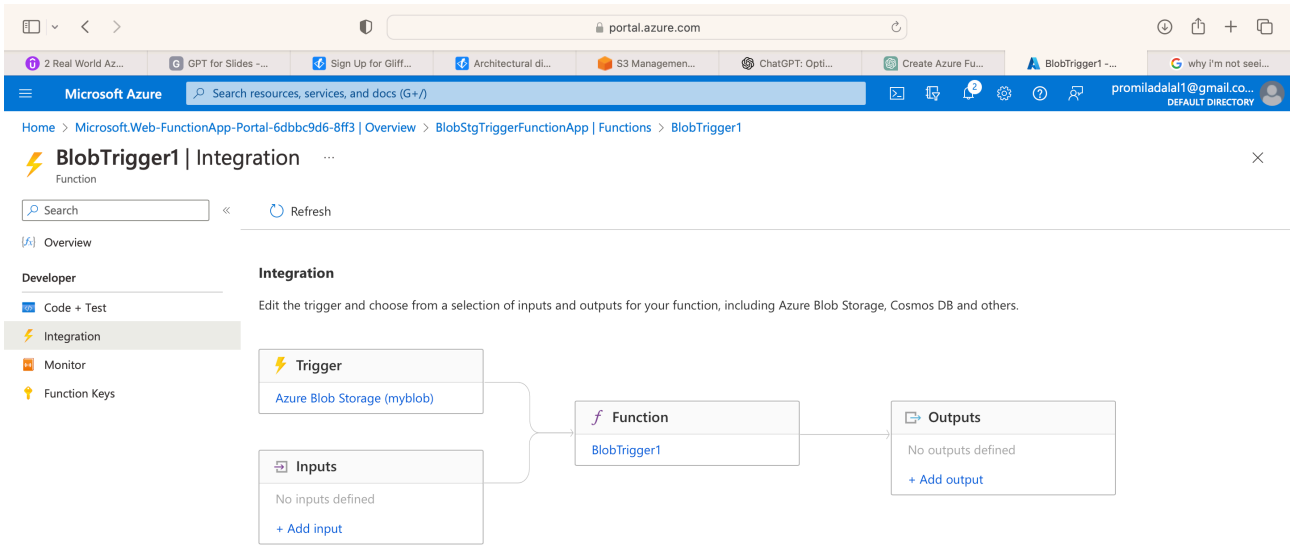


Create Azure Function with blob storage logic:

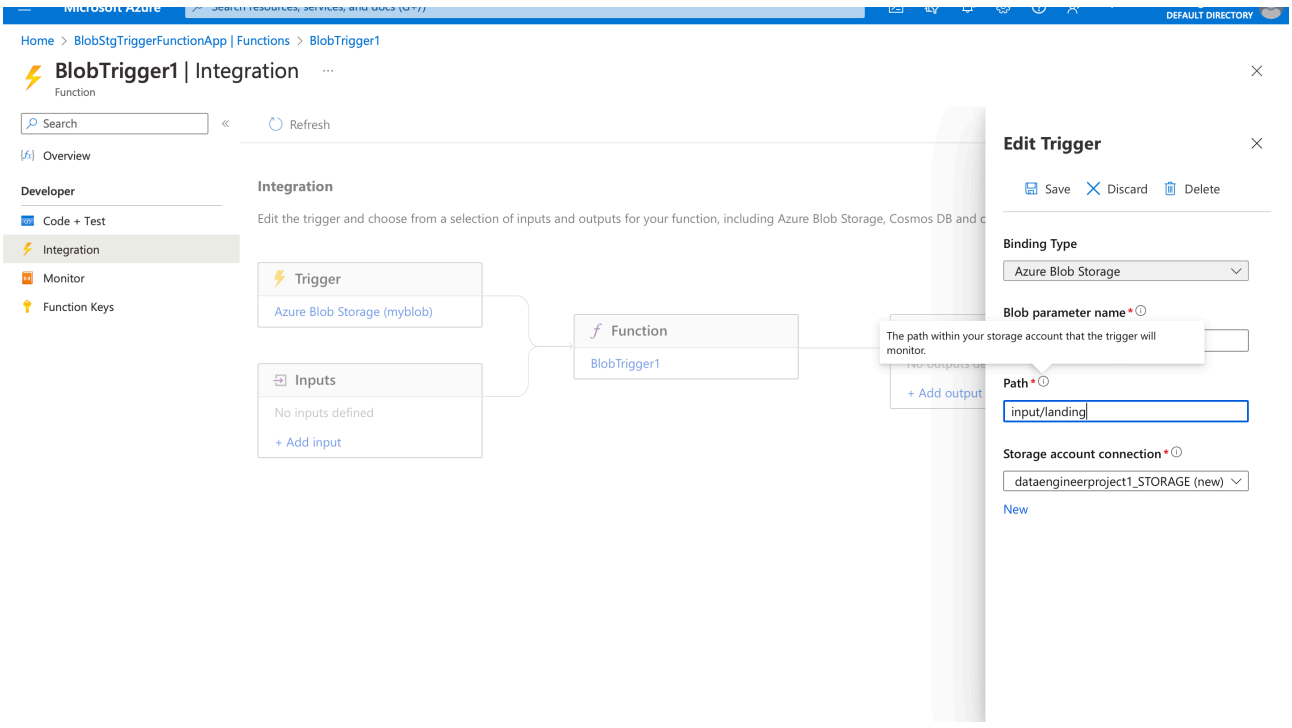
To create the azure function which'll validate the json & it'll going to run as soon as file is landed up in landing folder from amazon s3, it'll be triggered up. This ADF'll be run on daily basis & this function'll decide whether incoming file should move to staging folder or rejected folder.

1). Creation of Function app

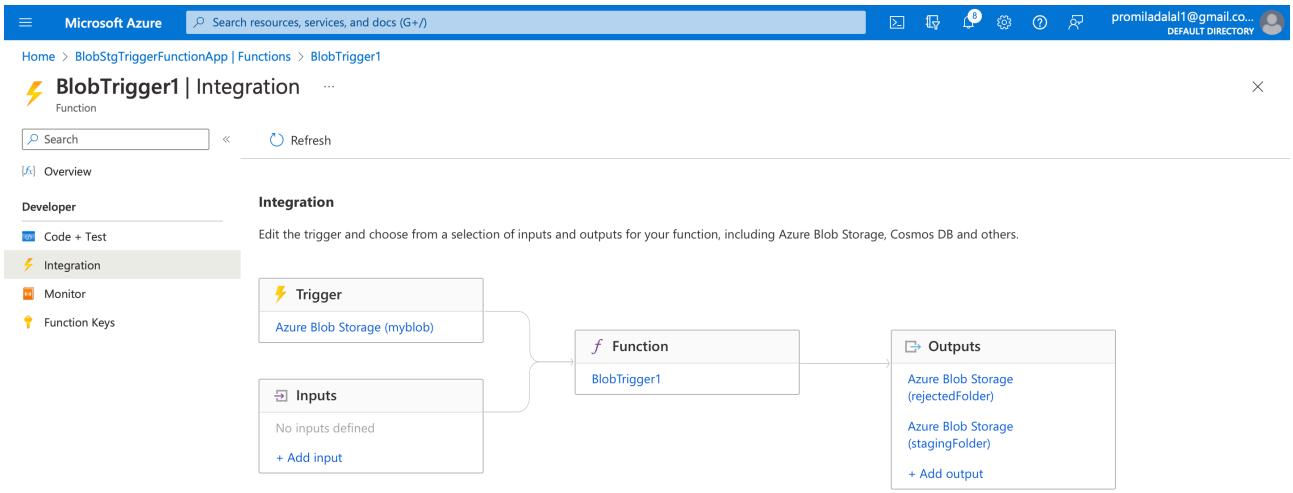
2). Creation of blob storage trigger function:



3). Setting up trigger path

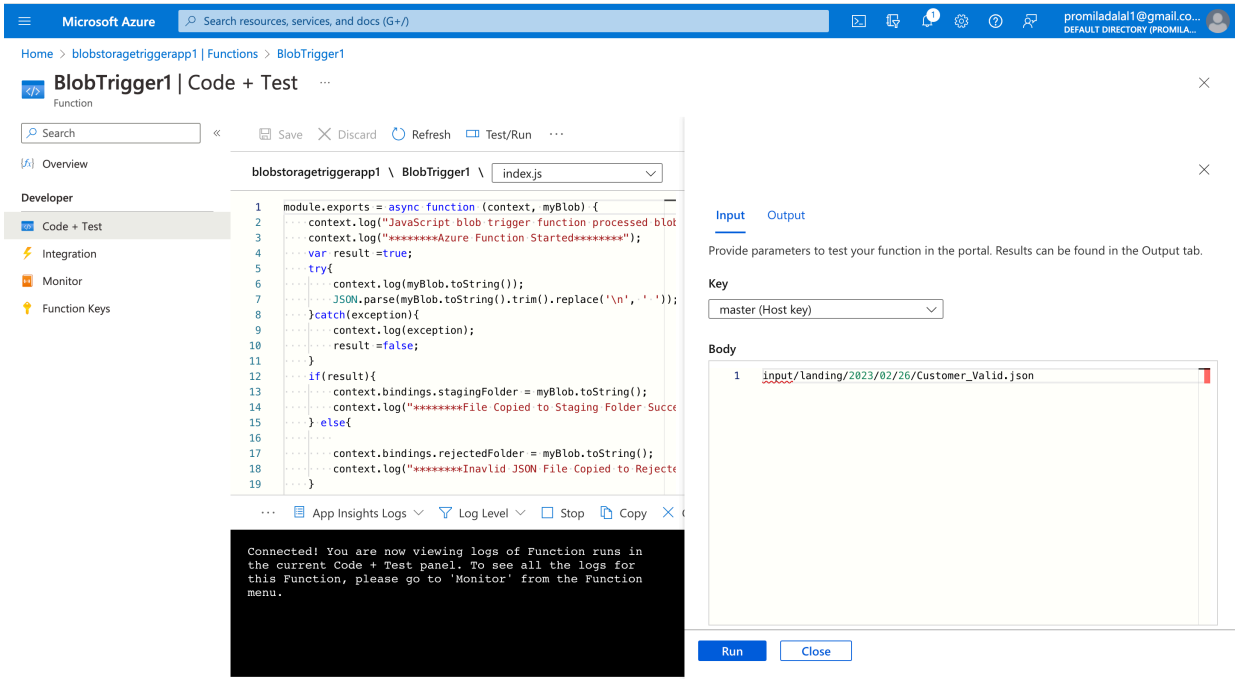


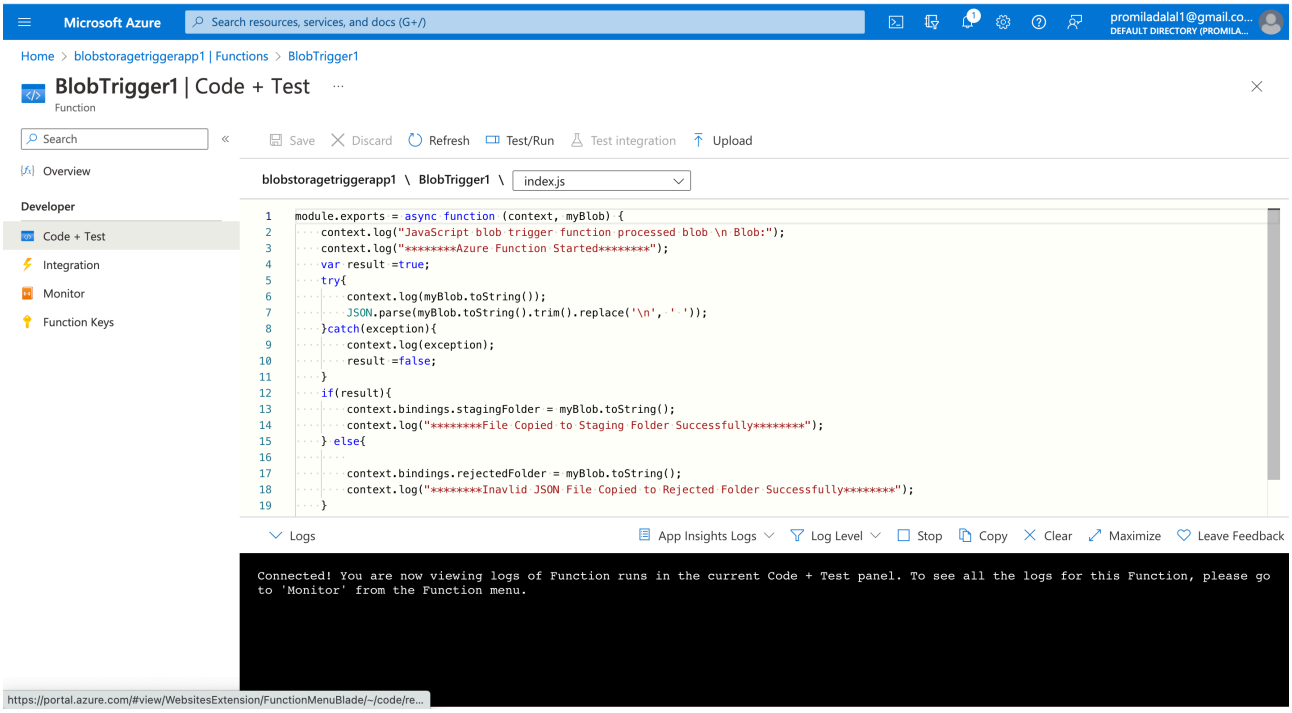
4). Create folders for rejected & staging folders



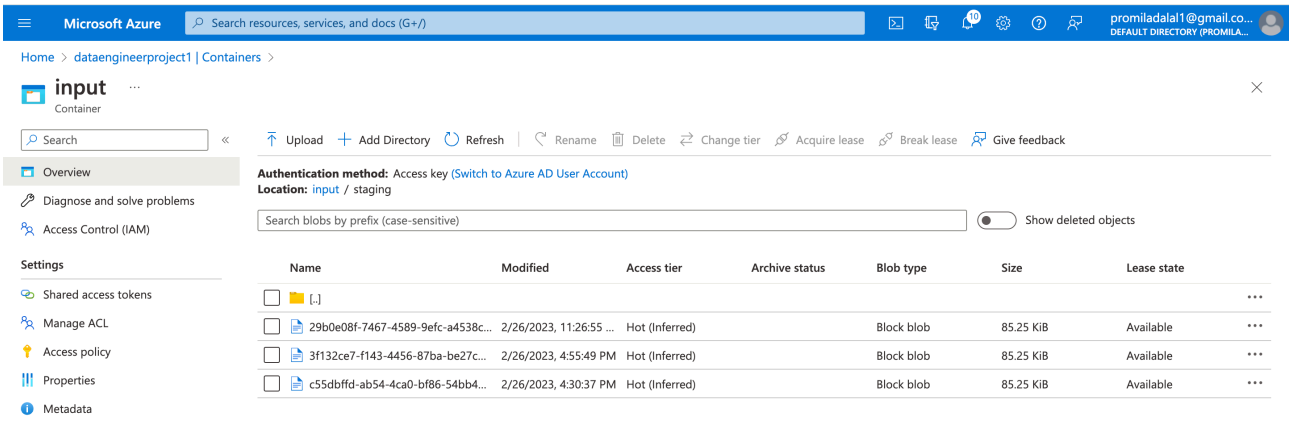
5). code+test

Click on Test/run & mention the folder path (i.e., landing in our case)



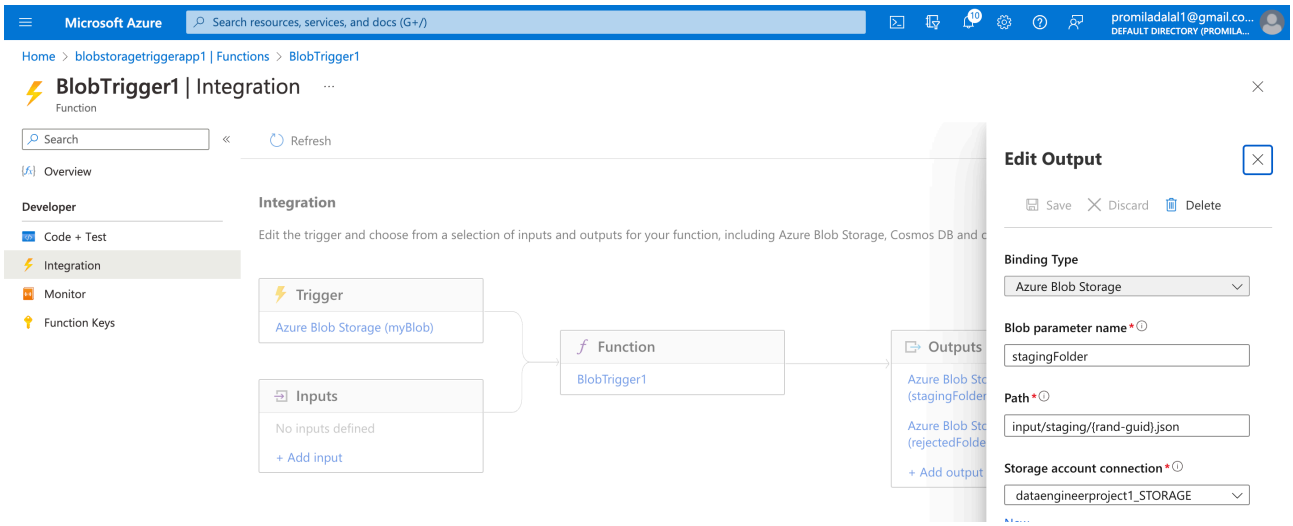


We can see that file copied to staging folder successfully



Add JSON format in the Azure function

-in 'Path' add .json for both stagingFolder & rejectedFolder outputs



Create Azure SQL Server & Database

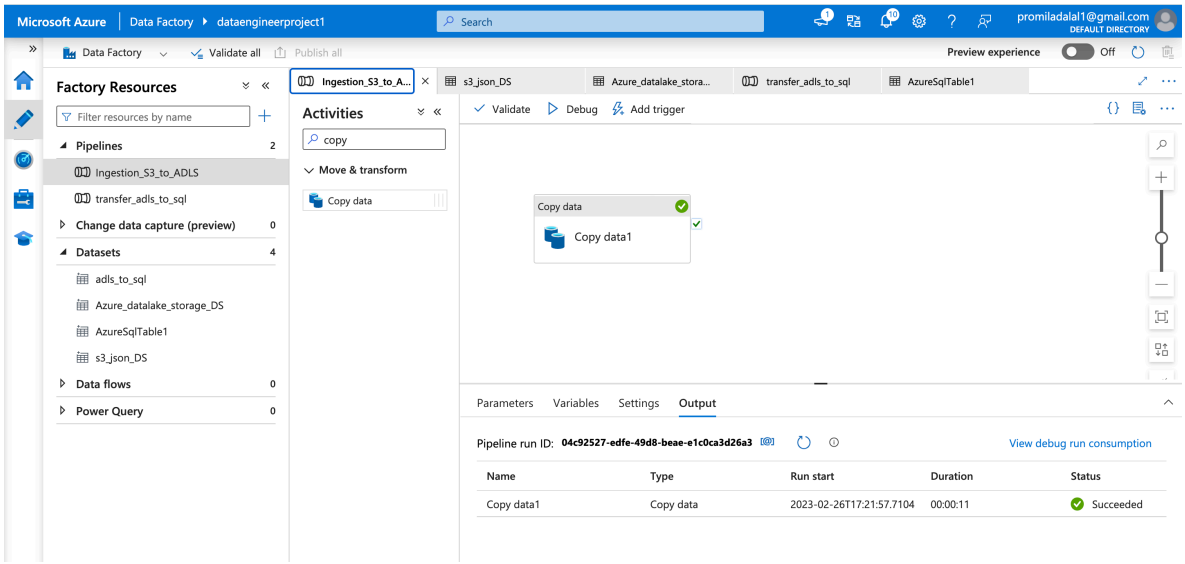
We'll use Azure data factory for creating a pipeline that will use copy activity for moving data from Azure data lake storage Gen2 to Azure SQL db. So for this purpose we have created SQL database & SQL server.

Create Pipeline for moving data from ADLS to Azure SQL DB:

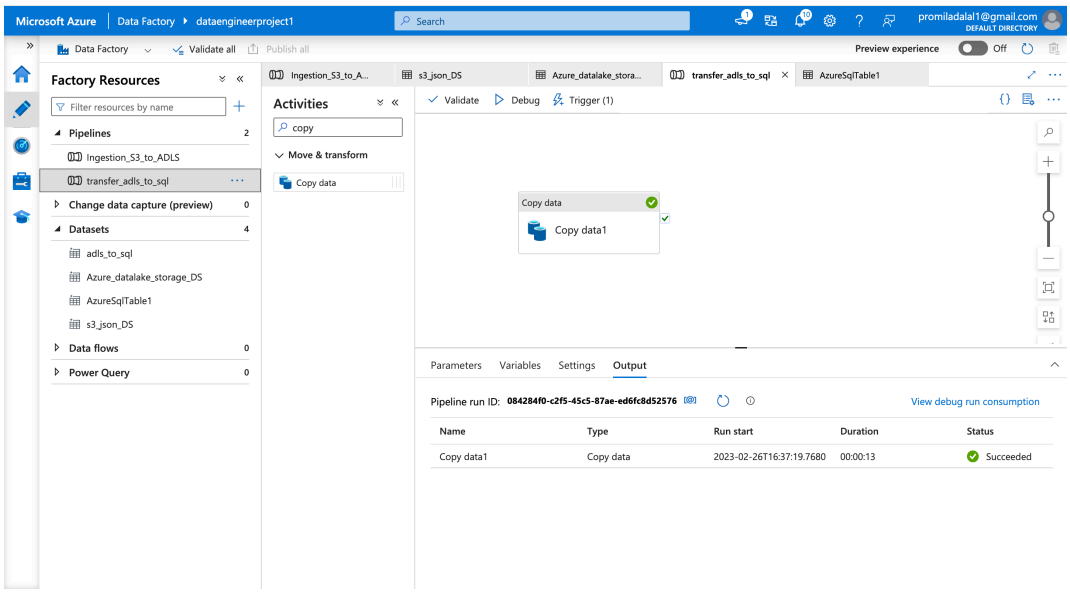
- Most of the steps are similar to the steps we utilised for creating pipeline for copying data from amazon s3 bucket to ADLS (with few exception)

End-to-end project execution:

Pipeline 1: for copying data from amazon s3 to ADLS (succeeded)



Pipeline 2: for copying data from adls to SQL DB



We can see that data has been copied to Azure SQL database