Customer Churn Azure Project

Note: The dataset provided for this assignment was already clean, with no missing values or Personally Identifiable Information (PII). However, for demonstration purposes, a filter condition was implemented to show how missing values can be handled using an Azure Data Factory (ADF) pipeline.

Project Objective

The goal of this project was to migrate data from an API source to an Azure SQL Database and establish an automated orchestration workflow using Azure Data Factory.

Technology Stack

- Resource Group
- Storage Accounts
- Azure Data Factory (ADF)
- Azure SQL Database

ADF Activities Used

- Lookup Activity
- Copy Data Activity
- Data Flow

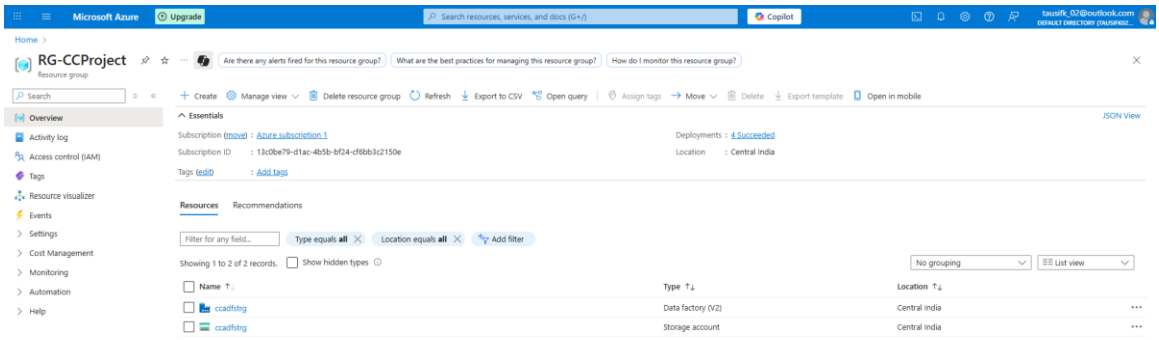
Project Workflow

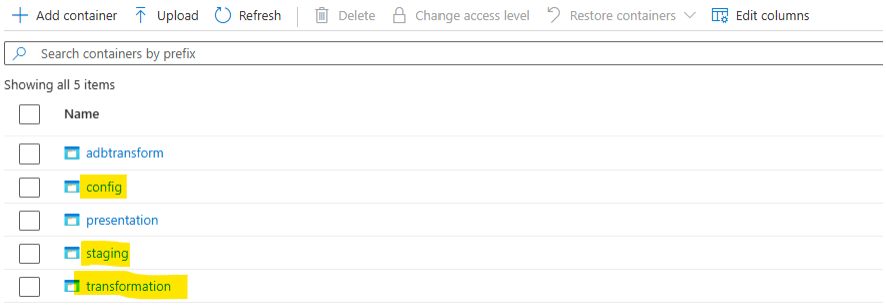
Step 1: Azure Resources Setup

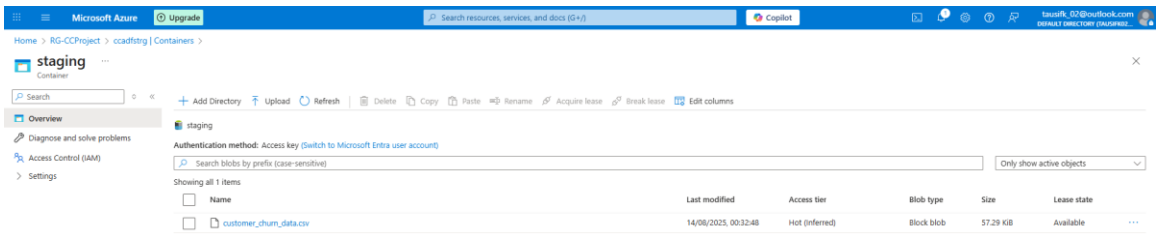
1. Created a Resource Group in Azure.
2. Added a Storage Account and Azure Data Factory instance under this resource group.
3. Within the Storage Account, created three containers:

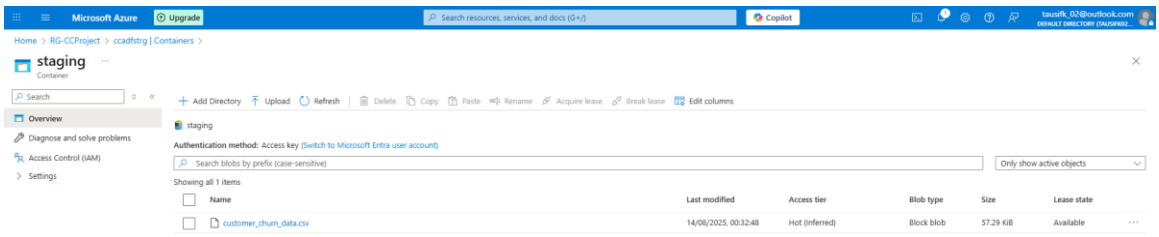
- config – Stores the file name from the API so the pipeline can automatically pick up new files.
- staging – Contains the raw CSV file loaded from the GitHub repository.
- transformation – Stores the transformed data in Parquet format for SQL loading.

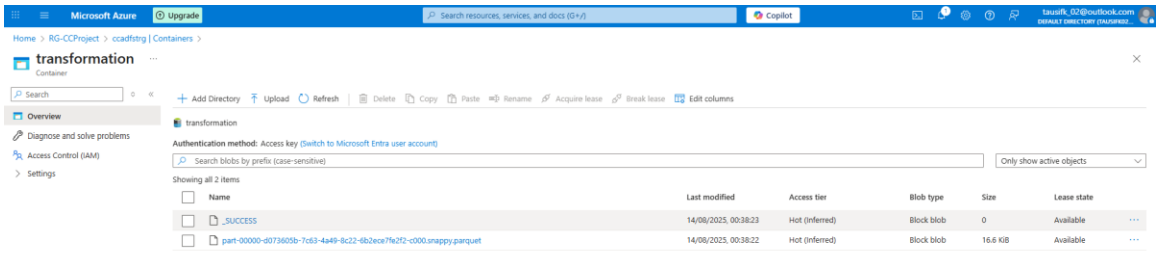
Screenshots for reference.:









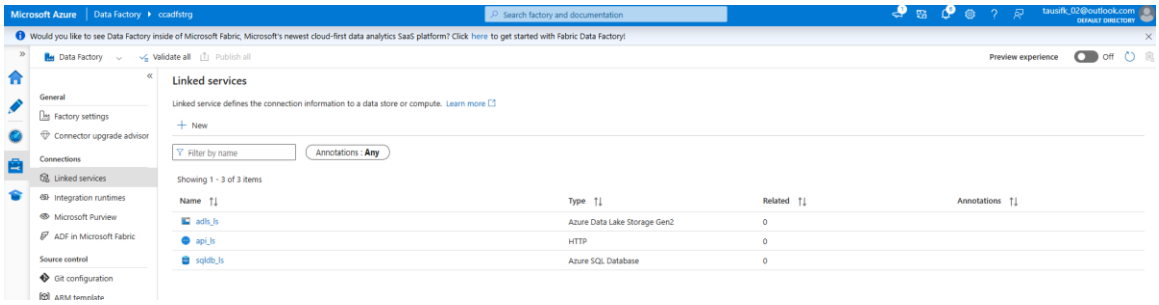


Step 2: Linked Services Configuration

Configured Linked Services in ADF for connections to:

- Azure Storage Account
- Azure SQL Database

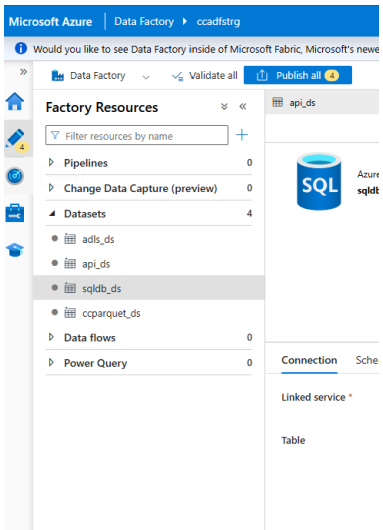
Screenshot for reference:



Step 3: Dataset Creation

Created datasets in ADF for:

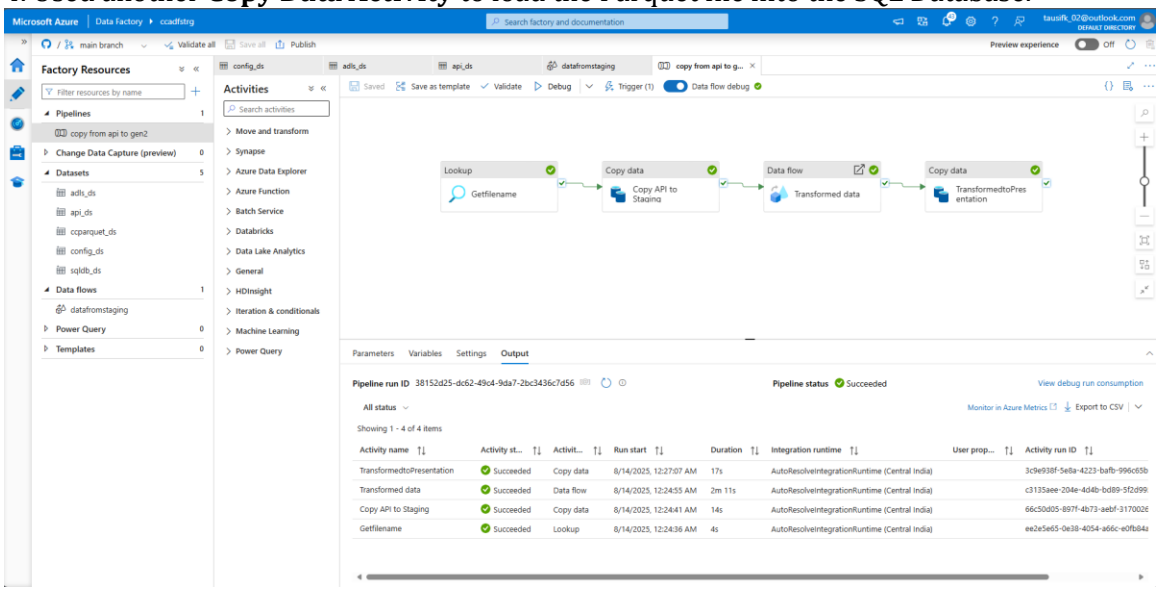
- The config file in the Storage Account
- The staging CSV file
- The transformed Parquet file
- The destination SQL Database table

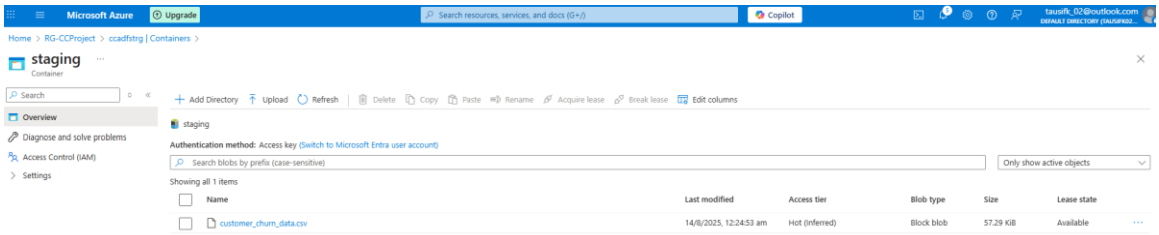


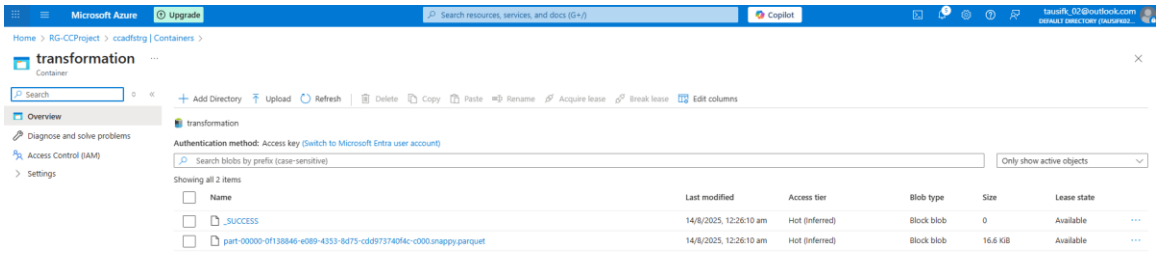
Step 4: ETL Pipeline Development

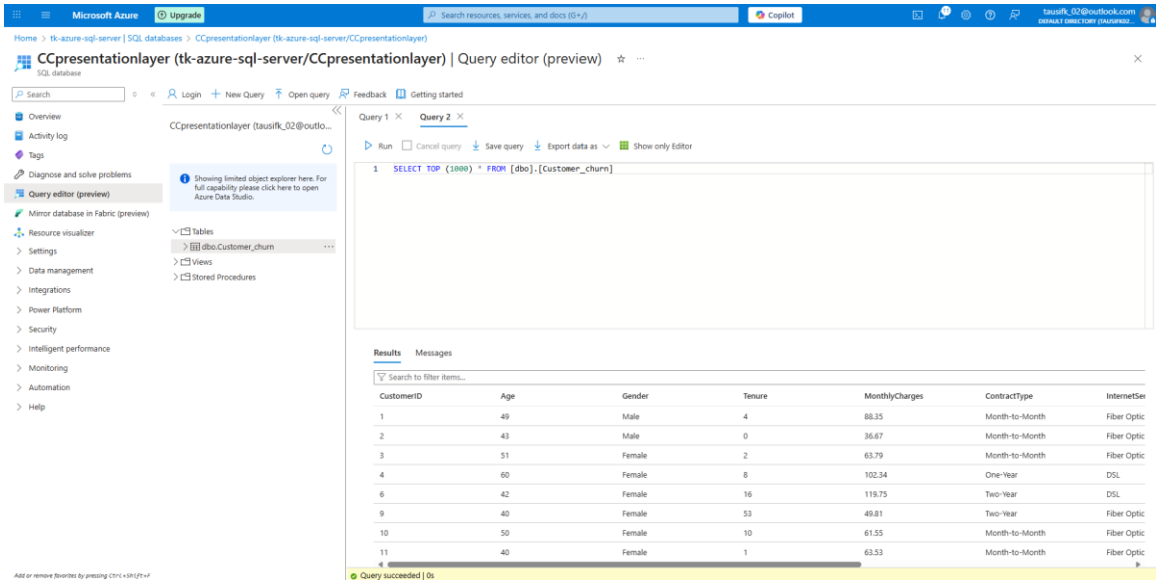
Pipeline Workflow:

1. The config file containing the file name is passed through a Lookup Activity to the Copy Data Activity.
2. In **Data Flows**, filtered out rows where “InternetService” = "None".
3. Saved the filtered data as a Parquet file in the transformation folder.
4. Used another **Copy Data Activity** to load the Parquet file into the SQL Database.



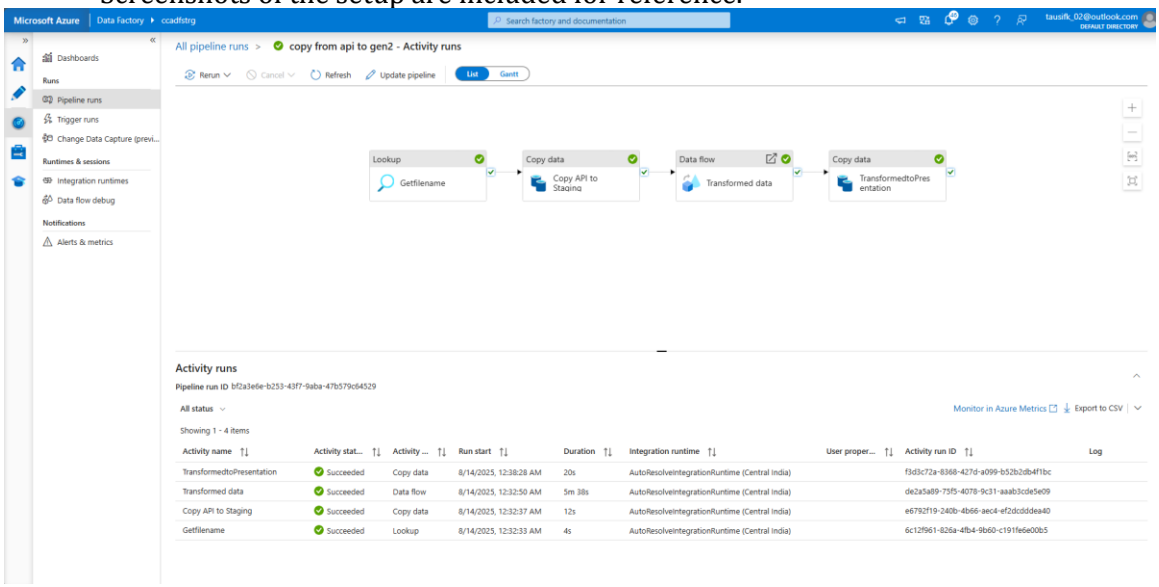


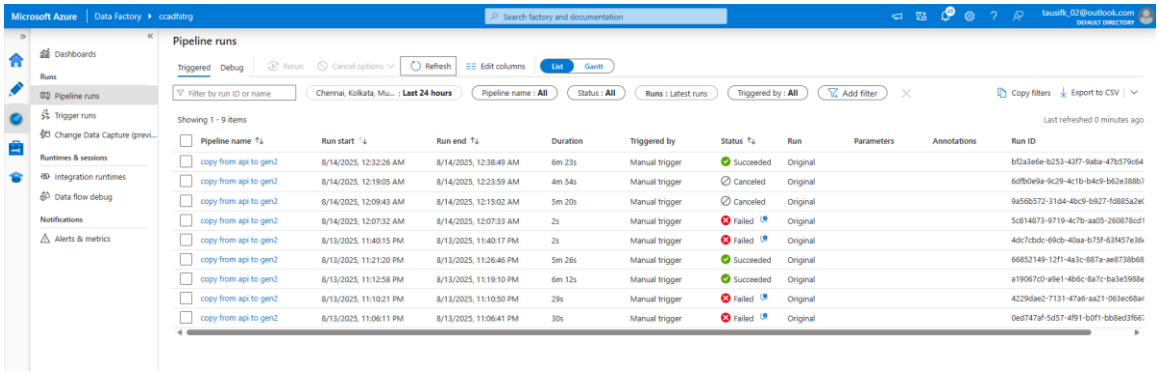




Step 5: Orchestration and Automation

- Set a trigger so the pipeline runs automatically at scheduled times. *As per assignment it is to be scheduled for every hour.*
- Screenshots of the setup are included for reference.





Outcome

This project shows how to:

- Move data from an API to Azure SQL Database automatically.
- Handle files in a dynamic way using a config file.
- Apply simple cleaning steps in ADF.
- Orchestrate the whole process in one workflow.