**Automated Employee Data Processing Using Azure Services**

**1. Introduction**

This document explains the development of an automated system to process employee data using Azure services. The system reads employee CSV files uploaded to Azure Blob Storage, validates the data using Azure Functions, stores valid records in Azure SQL Database, and sends notifications using Logic Apps. Monitoring is performed using Azure Monitor and Application Insights.

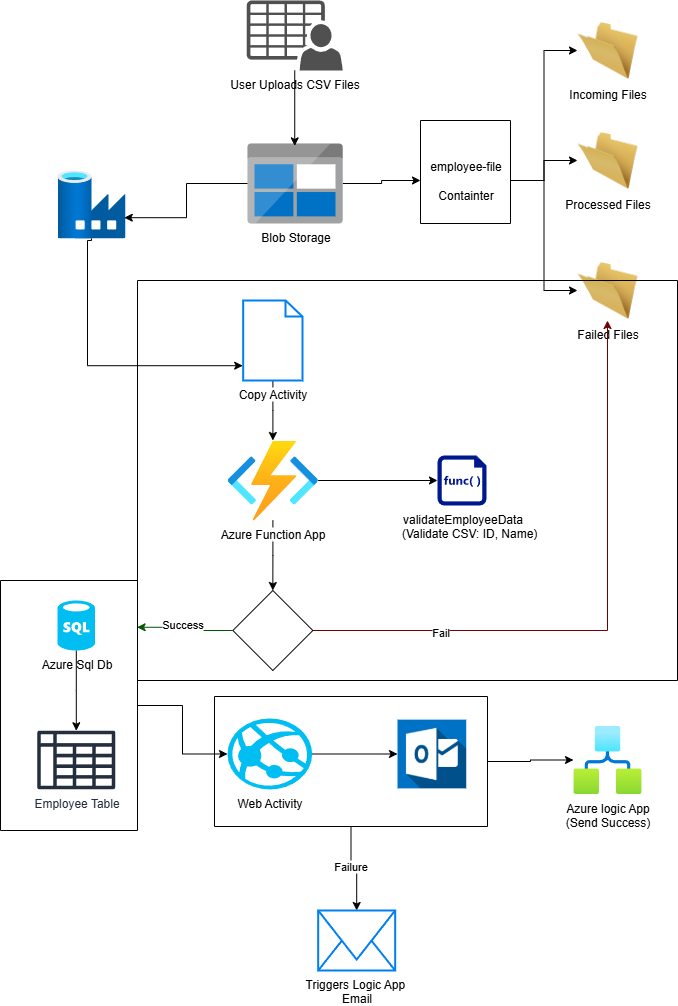
**2. Problem Statement**

A company receives daily employee data files in CSV format. The goal is to automate the processing of these files: validate the data, insert valid records into the database, and notify the relevant team about success or failure.

**3. Azure Services Used**

| **Azure Service** | **Purpose** |
| --- | --- |
| Azure Blob Storage | Store incoming CSV files |
| Azure Data Factory | Orchestrate ETL and validation pipeline |
| Azure Function | Validate CSV data |
| Azure SQL Database | Store valid employee records |
| Azure Logic Apps | Send email notifications |
| Azure Monitor | Track execution logs and monitor failures |
|  |  |

**4. Architecture Diagram**



**Figure:** Architecture of automated employee CSV processing using Azure.

**5. Step-by-Step Implementation**

**Task 1: Data Setup**

* Created a Blob container employee-files with folders: incoming/, intermediate/, processed/, and failed/
* Created Azure SQL Database with table Employees:

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

Name NVARCHAR(100),

Department NVARCHAR(100),

Salary FLOAT

);

* Uploaded sample CSV:

EmployeeID,Name,Department,Salary

101,John Doe,IT,60000

102,Jane Smith,HR,55000

**Task 2: Azure Data Factory**

* Created an event-based trigger for new files in incoming/
* Copy Activity moves files to intermediate/
* Azure Function Activity validates data
* If Condition:
  + If valid → insert data into SQL
  + If invalid → move file to failed/
* Retry policy enabled for robustness

**Task 3: Azure Function**

* Function name: validateEmployeeData
* Validates required fields: EmployeeID, Name, Department, Salary
* Returns 200 if valid, 400 if any field is missing.

**Code Sample:**

import logging, csv, io

import azure.functions as func

app = func.FunctionApp()

REQUIRED\_FIELDS = ["EmployeeID", "Name", "Department", "Salary"]

def validate\_csv(csv\_text: str) -> list[str]:

reader = csv.DictReader(io.StringIO(csv\_text))

errors = []

for row\_num, row in enumerate(reader, start=2):

for field in REQUIRED\_FIELDS:

if not row.get(field):

errors.append(f"Row {row\_num}: Missing '{field}'")

return errors

@app.function\_name(name="validateEmployeeData")

@app.route(route="validate", auth\_level=func.AuthLevel.FUNCTION, methods=["POST"])

def validate\_employee\_data(req: func.HttpRequest) -> func.HttpResponse:

try:

csv\_text = None

if req.headers.get("Content-Type", "").startswith("multipart/form-data"):

form\_data = req.get\_form()

if b"file" in form\_data:

csv\_bytes = form\_data[b"file"]

csv\_text = csv\_bytes.decode("utf-8")

if csv\_text is None:

csv\_text = req.get\_body().decode("utf-8")

errors = validate\_csv(csv\_text)

if errors:

return func.HttpResponse("Validation Failed:\n" + "\n".join(errors), status\_code=400)

return func.HttpResponse("All employee records are valid.", status\_code=200)

except Exception as exc:

logging.exception("Unexpected server error")

return func.HttpResponse(f"Server Error: {exc}", status\_code=500)

**Task 4: Azure Logic App**

* Triggered via HTTP POST from ADF
* Includes a condition to check validation status
* Actions:
  + If success → Send email: “ Employee data successfully loaded.”
  + If failure → Send email: “ Data validation failed.”
* Endpoint secured using Azure AD RBAC

**Task 5: Monitoring and Logging**

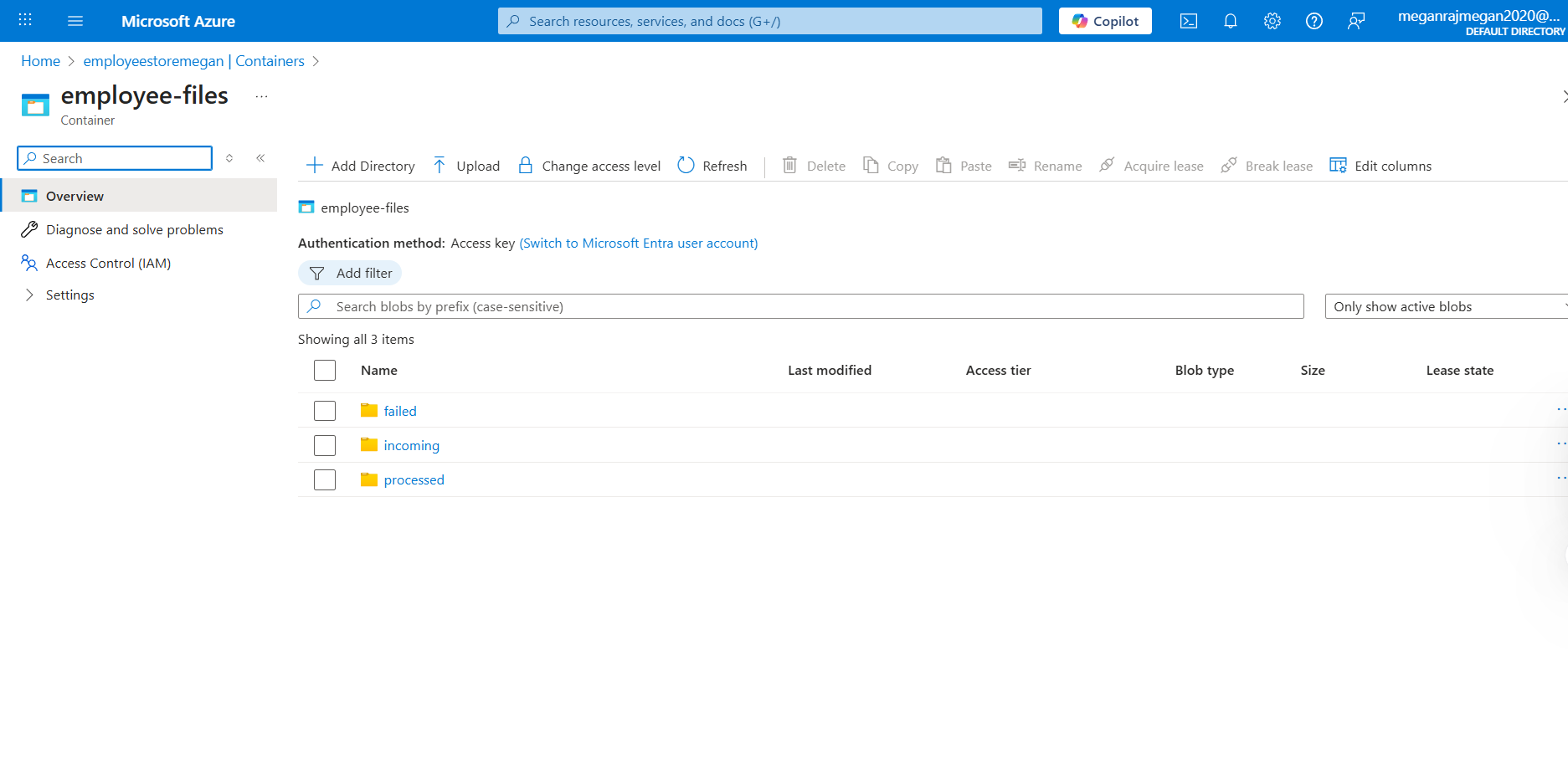
* Azure Monitor was used to track pipeline runs and activity logs
* Application Insights was **not** used in this implementation.
* Logs were monitored using the built-in **ADF Monitor tab** and **Function App log stream**.

**6. Conclusion**

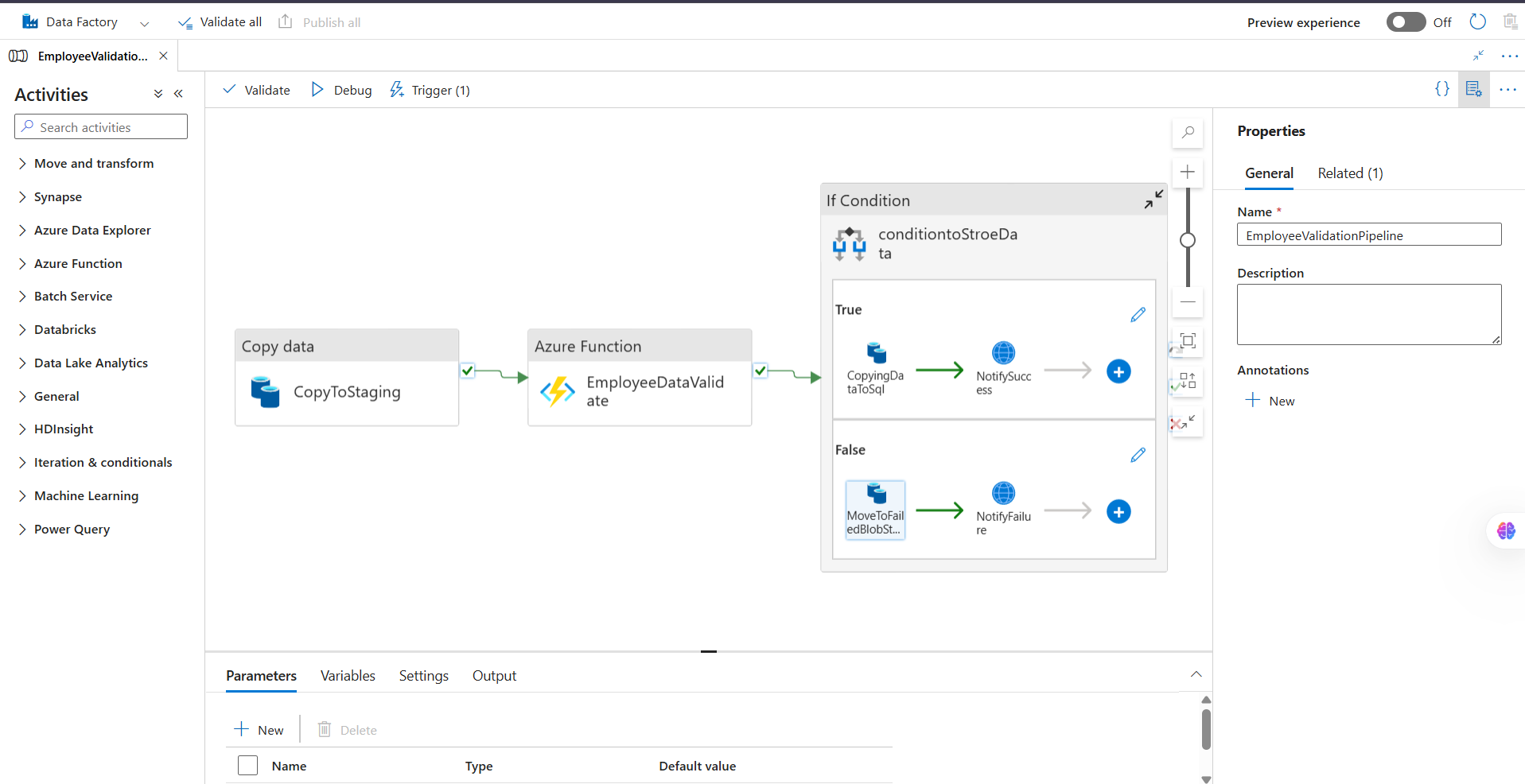
This project demonstrates a complete Azure-based automation pipeline to validate and process employee CSV files. It ensures that only clean data is inserted into the database, automates alerts, and provides observability through monitoring tools. The solution is scalable, maintainable, and aligns with cloud-native best practices.

**7. Screenshots**

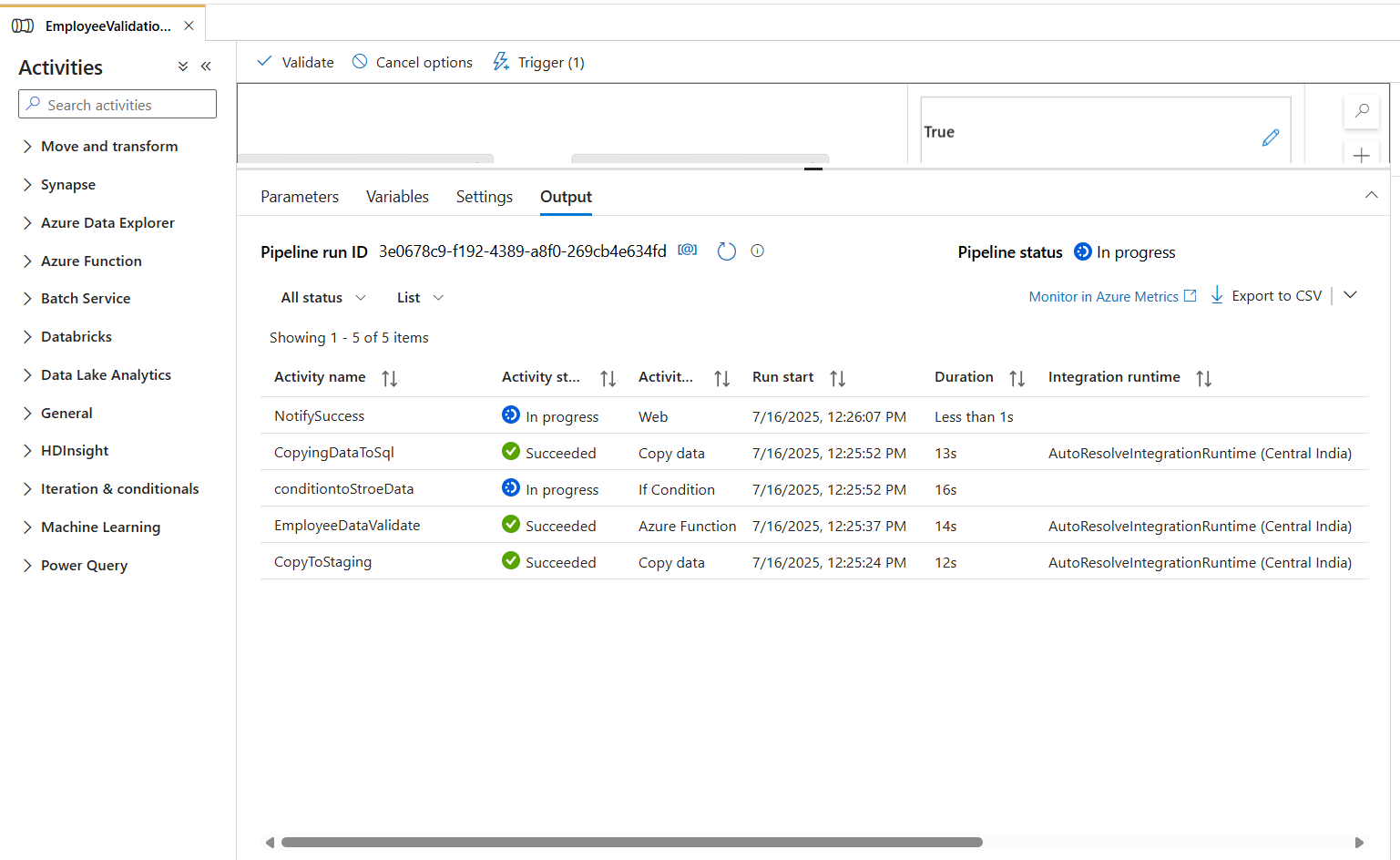
* Blob Storage folder structure



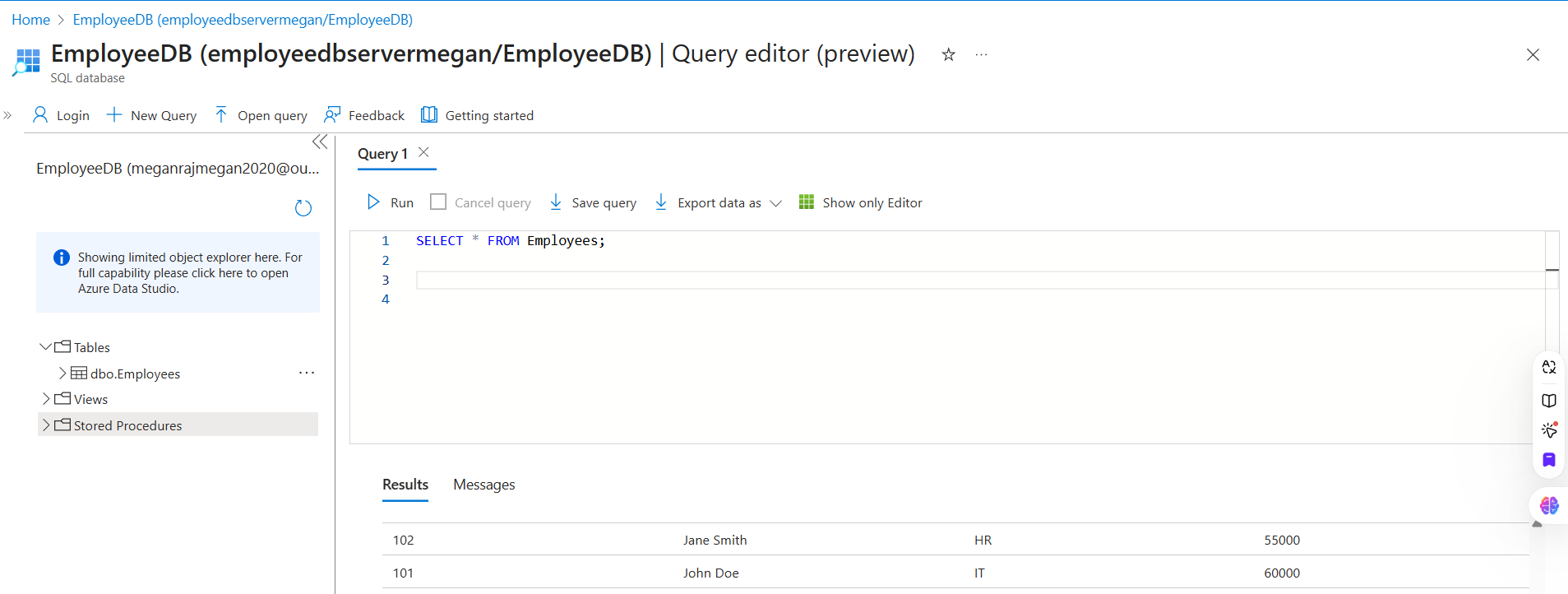
* ADF pipeline design



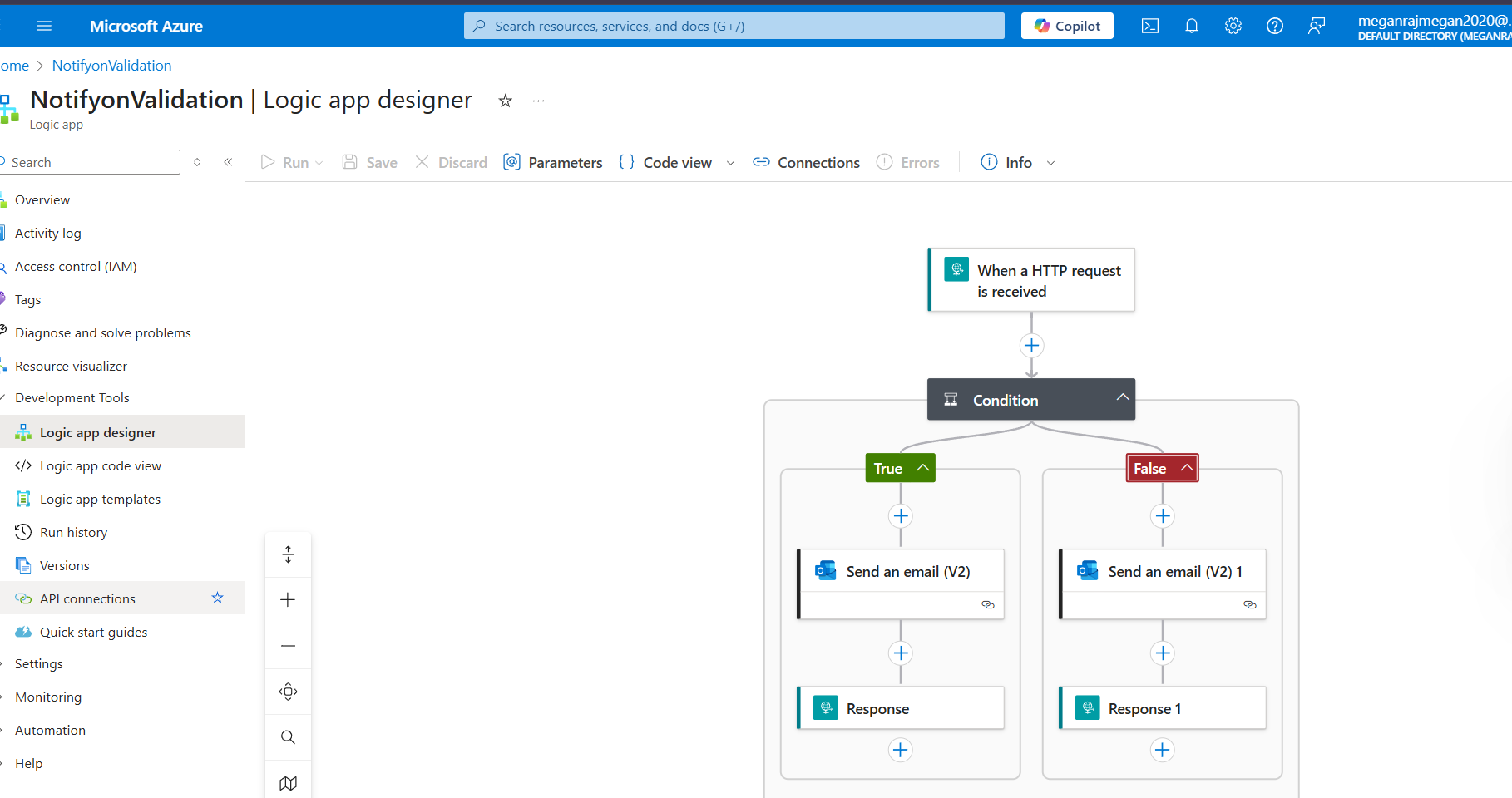
* Azure Function log output



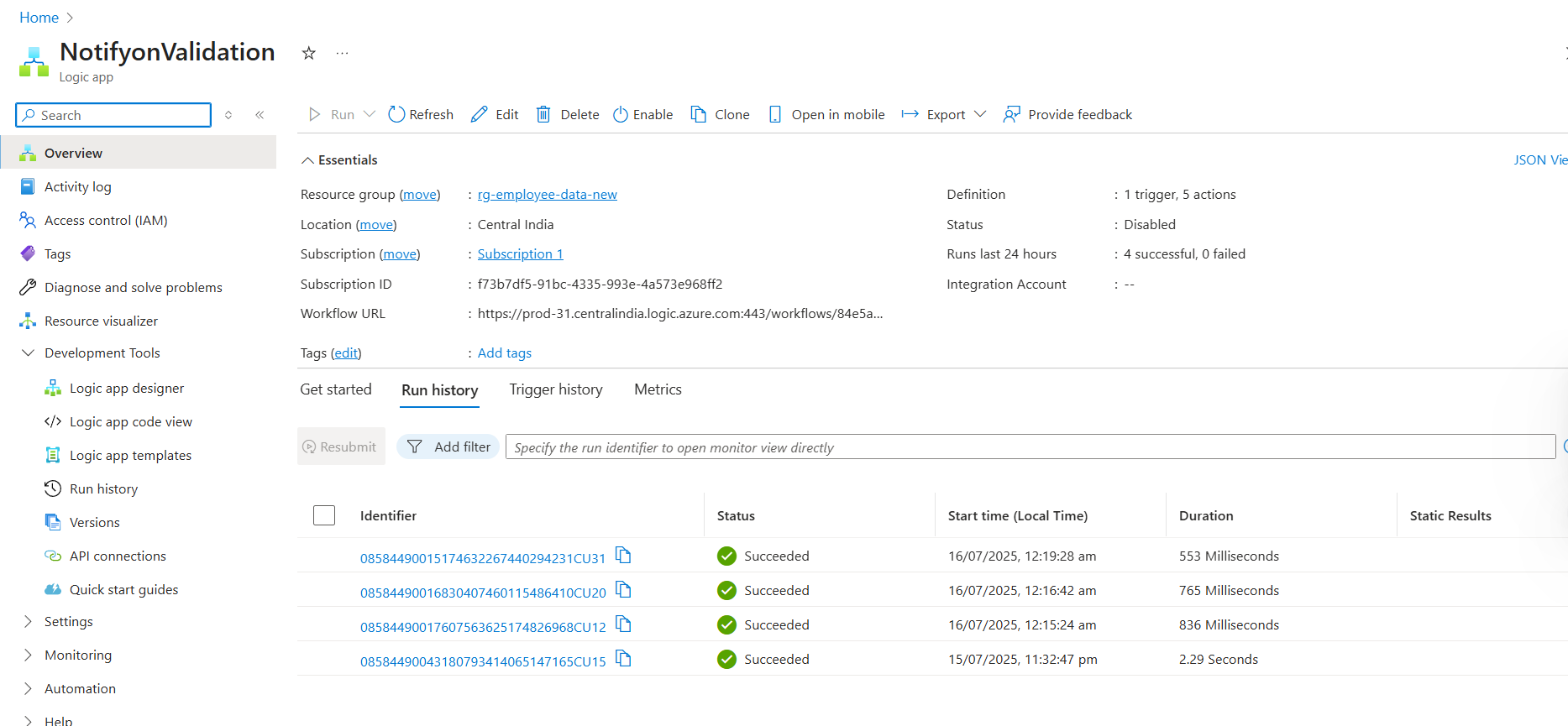
* SQL database records



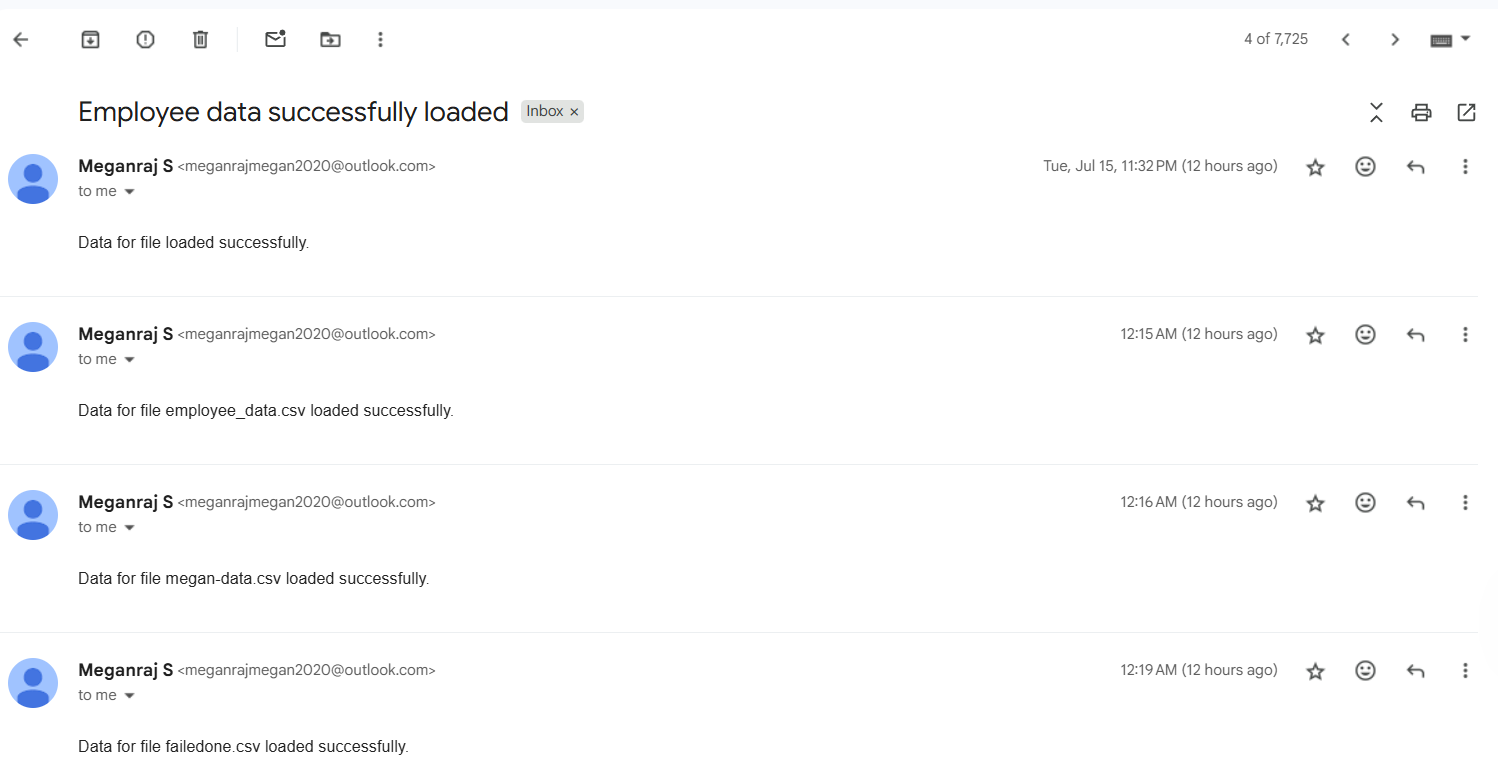
* Logic App Designer



* Logic App run history



* Notified email samples



**8. GitHub Repository**

Link: [Employee-csv-validator-azure](https://github.com/Meganraj/employee-csv-validator-azure.git)