Parallel Processing Data Pipeline

Household Power Consumption

By Vaishali Bokadiya
Vaishalibokadiya19@gmail.com

Project Overview:

The objective of this project is to develop a robust and efficient data pipeline in Azure for processing and analyzing household power consumption data. The pipeline will leverage parallel processing capabilities to enhance performance and scalability, ultimately loading the processed data into Azure Data Lake Storage for further analysis and insights generation.

Project Requirements:

1. **Data Integration and Ingestion:**
    - Implement data ingestion mechanisms to raw household power consumption dataset from https://data.world/databeats/household-power-consumption and load it into Azure Data Lake Gen2 Storage Account.
    - Ensure data ingestion processes are reliable, scalable, and capable of handling large volumes of data.

2. **Data Transformation and Preprocessing:**
    - Develop data transformation logic to preprocess and cleanse raw power consumption data.
    - Implement data quality checks and validation processes to ensure data accuracy and integrity.
    - Perform necessary data transformations such as aggregation, and enrichment to prepare the data for analysis.

3. **Parallel Processing with Azure Databricks:**
    - Utilize Azure Databricks for parallel processing of power consumption data to improve performance and scalability.
    - Design and implement parallel processing workflows using Spark clusters in Azure Databricks.

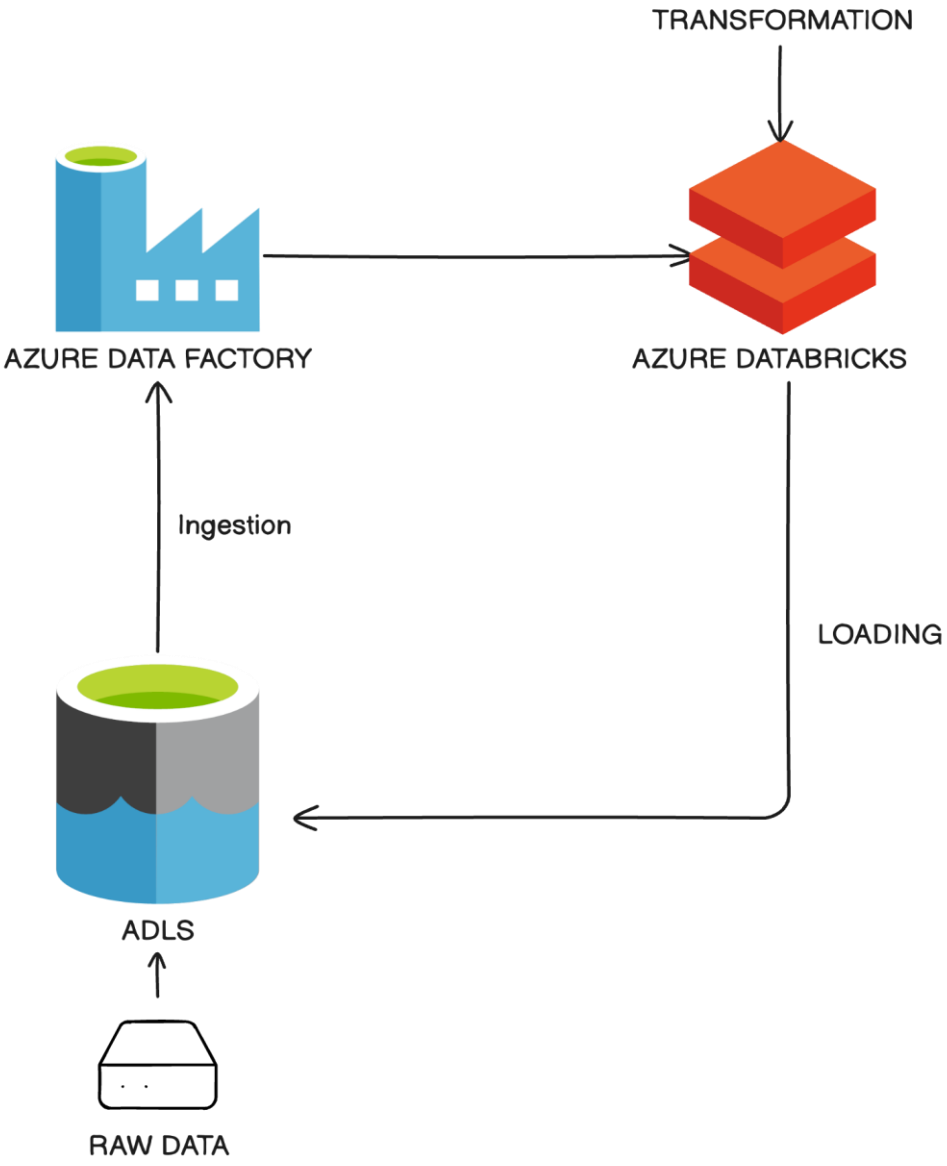
- Optimize cluster configurations and resource allocation for efficient parallel processing.

4. **Integration with Azure Data Lake Gen2:**
    - Configure Azure Data Lake Gen2 Storage Account as the destination for storing processed power consumption data.
    - Establish seamless integration between Azure Databricks and Azure Data Lake Gen2 for data transfer and storage.

5. **Orchestration with Azure Data Factory:**
    - Develop data pipelines within Azure Data Factory to orchestrate data movement and transformation tasks.
    - Define data ingestion activities to extract data from source systems and load it into Azure Databricks for processing.
    - Implement data transformation activities to preprocess and cleanse the data within Databricks, then store the processed data in Azure Data Lake Gen2.

6. **Monitoring and Logging:**
    - Implement monitoring and logging mechanisms to track the performance and health of the data pipeline.
    - Monitor key metrics such as data ingestion rates, processing times, and error rates.
    - Configure alerts and notifications for proactive monitoring and issue resolution.

Architecture Diagram:



Azure Resources used for the Project:

1. **Azure Databricks**

   Azure Databricks is a fast, easy, and collaborative Apache Spark-based analytics platform optimized for Azure. It provides a fully managed, cloud-based environment that integrates seamlessly with other Azure services, allowing data engineers, data scientists, and analysts to collaborate on big data and machine learning projects.

2. **Azure Data Lake Gen 2 Storage Account**

   Azure Data Lake Storage Gen2 is a scalable and secure data lake solution built on top of Azure Blob Storage and the Azure Blob Storage file system (ABFS). It combines the capabilities of Azure Blob Storage with hierarchical namespace and file system semantics, providing a unified data lake storage solution for big data analytics and data warehousing workloads.

3. **Azure Data Factory**

   Azure Data Factory is a cloud-based data integration service that allows you to create, schedule, and manage data pipelines for moving and transforming data across various sources and destinations. It enables you to orchestrate and automate data workflows, facilitating data integration, transformation, and loading tasks.

About Dataset:

Individual household electric power consumption dataset collected via submeters placed in 3 distinct areas of a home

Data Set Information

This household electricity consumption dataset contains 260,640 measurements gathered between January 2007 and June 2007 (6 months). It is a subset of a larger, original archive that contains 2,075,259 measurements gathered between December 2006 and November 2010 (47 months).

Attribute Information

1. **date:** Date in format dd/mm/yyyy
2. **time:** time in format hh:mm:ss
3. **global_active_power:** household global minute-averaged active power (in kilowatt)
4. **global_reactive_power:** household global minute-averaged reactive power (in kilowatt)
5. **voltage:** minute-averaged voltage (in volt)
6. **global_intensity:** household global minute-averaged current intensity (in ampere)
7. **sub_metering_1:** energy sub-metering No. 1 (in watt-hour of active energy). It corresponds to the kitchen, containing mainly a dishwasher, an oven and a microwave (hot plates are not electric but gas powered).
8. **sub_metering_2:** energy sub-metering No. 2 (in watt-hour of active energy). It corresponds to the laundry room, containing a washing-machine, a tumble-drier, a refrigerator and a light.
9. **sub_metering_3:** energy sub-metering No. 3 (in watt-hour of active energy). It corresponds to an electric water-heater and an air-conditioner.

Dataset Snapshot

household_power_consumption.csv

household_power_consumption.csv > data

```
Date,Time,Global_active_power,Global_reactive_power,Voltage,Global_intensity,Sub_metering_1,Sub_metering_2,Sub_metering_3
01-01-2007,00:00:00,2.58,0.136,241.97,10.6,0,0,0
01-01-2007,00:01:00,2.552,0.1,241.75,10.4,0,0,0
01-01-2007,00:02:00,2.55,0.1,241.64,10.4,0,0,0
01-01-2007,00:03:00,2.55,0.1,241.71,10.4,0,0,0
01-01-2007,00:04:00,2.554,0.1,241.98,10.4,0,0,0
01-01-2007,00:05:00,2.55,0.1,241.83,10.4,0,0,0
01-01-2007,00:06:00,2.534,0.096,241.07,10.4,0,0,0
01-01-2007,00:07:00,2.484,0,241.29,10.2,0,0,0
01-01-2007,00:08:00,2.468,0,241.23,10.2,0,0,0
01-01-2007,00:09:00,2.486,0,242.18,10.2,0,0,0
01-01-2007,00:10:00,2.492,0,242.46,10.2,0,0,0
01-01-2007,00:11:00,2.5,0,242.88,10.2,0,0,0
01-01-2007,00:12:00,2.494,0,242.57,10.2,0,0,0
01-01-2007,00:13:00,2.492,0,242.41,10.2,0,0,0
01-01-2007,00:14:00,2.48,0,241.81,10.2,0,0,0
01-01-2007,00:15:00,2.478,0,241.73,10.2,0,0,0
01-01-2007,00:16:00,2.47,0,241.29,10.2,0,0,0
01-01-2007,00:17:00,2.466,0,241.11,10.2,0,0,0
01-01-2007,00:18:00,2.456,0,240.59,10.2,0,0,0
01-01-2007,00:19:00,2.46,0,240.83,10.2,0,0,0
01-01-2007,00:20:00,2.544,0.092,240.9,10.6,0,0,0
01-01-2007,00:21:00,2.55,0.116,241.15,10.4,0,1,0
01-01-2007,00:22:00,2.554,0.118,241.55,10.6,0,1,0
01-01-2007,00:23:00,2.65,0.218,241.67,11,0,2,0
01-01-2007,00:24:00,2.682,0.258,242.45,11,0,1,0
01-01-2007,00:25:00,2.66,0.252,241.6,11,0,1,0
```

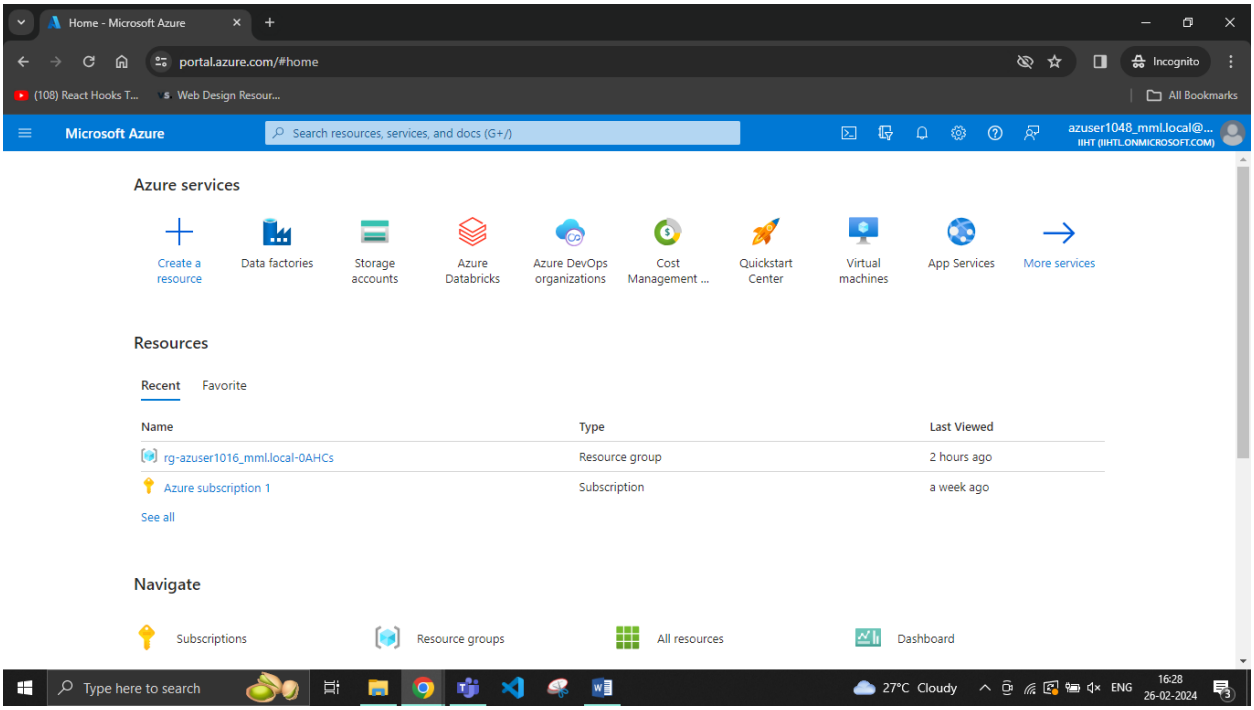
How it works:

1. **Set up Azure Data Lake Gen2:** Ensure that you have an Azure Data Lake Gen2 storage account created and properly configured.
2. **Create an Azure Databricks Workspace:** Set up an Azure Databricks workspace in your Azure portal. You can follow the documentation to create a Databricks workspace if you haven't already done so.
3. **Create an Azure Data Factory:** Create an Azure Data Factory instance in your Azure portal if you haven't already. ADF will orchestrate the data loading process and integrate with Databricks.
4. **Link Azure Databricks to Azure Data Factory:** In Azure Data Factory, create a linked service for Azure Databricks. This involves providing the necessary connection details and authentication credentials to access your Databricks workspace.
5. **Develop Data Processing Logic in Databricks:** In your Databricks workspace, develop the necessary data processing logic using Apache Spark. This can include data transformation, cleansing, enrichment, etc. Ensure that your Spark job is optimized for parallel processing to leverage Databricks' capabilities effectively.
6. **Create Data Factory Pipeline:** In Azure Data Factory, create a pipeline that orchestrates the data loading process. Add activities to trigger the Databricks notebook job, specifying any parameters or dependencies required for execution.
7. **Configure Parallelism:** Within your Data Factory pipeline, configure parallelism settings to optimize data loading performance. This may involve partitioning data, parallelizing processing tasks, and tuning resource allocation in Databricks to maximize throughput.
8. **Schedule and Monitor:** Schedule the Data Factory pipeline to run at specified intervals or trigger it based on event triggers. Monitor the execution of the pipeline and Databricks jobs to ensure they run smoothly and meet performance expectations.

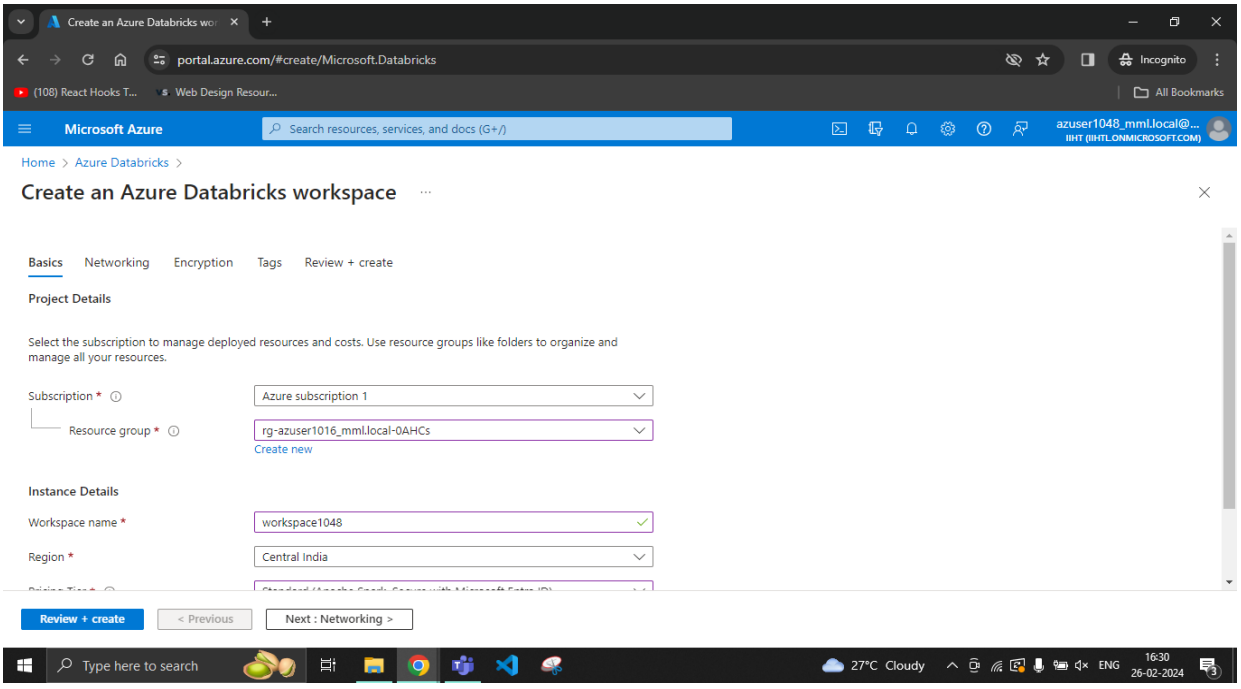
Tasks Performed:

1. Create Azure Databricks workspace, and create a compute and a notebook

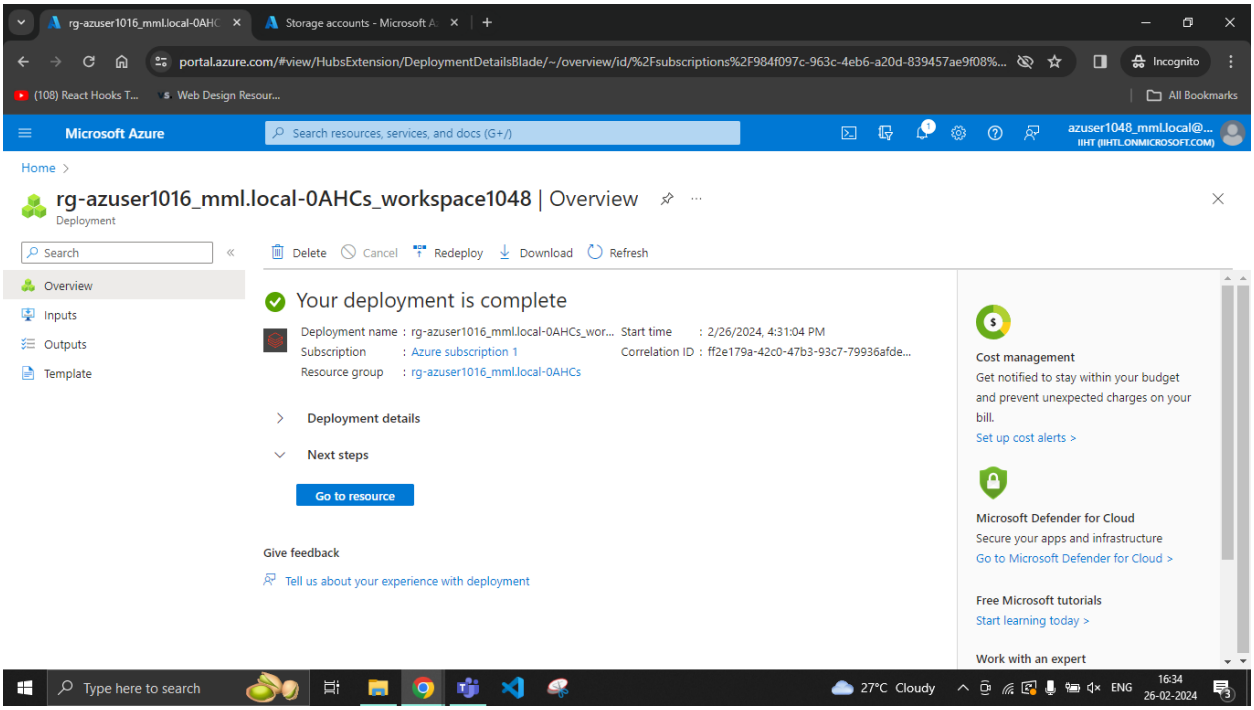
Open Microsoft Azure



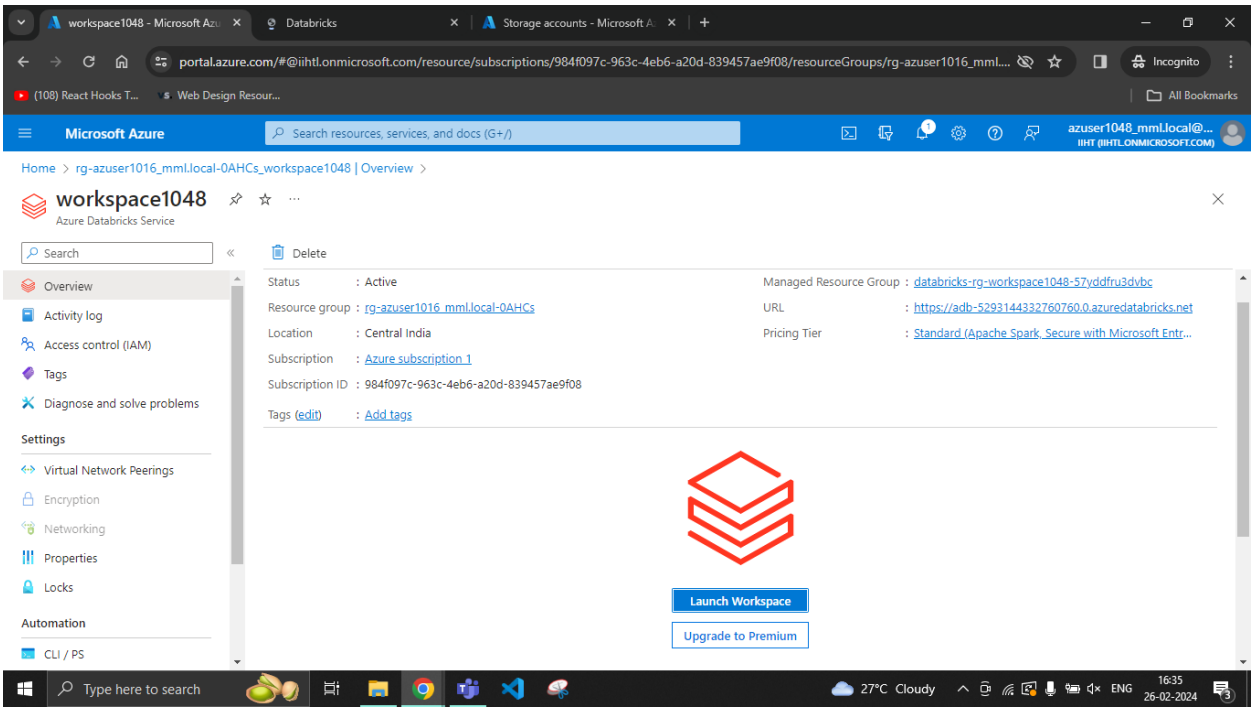
Create Azure Databricks workspace



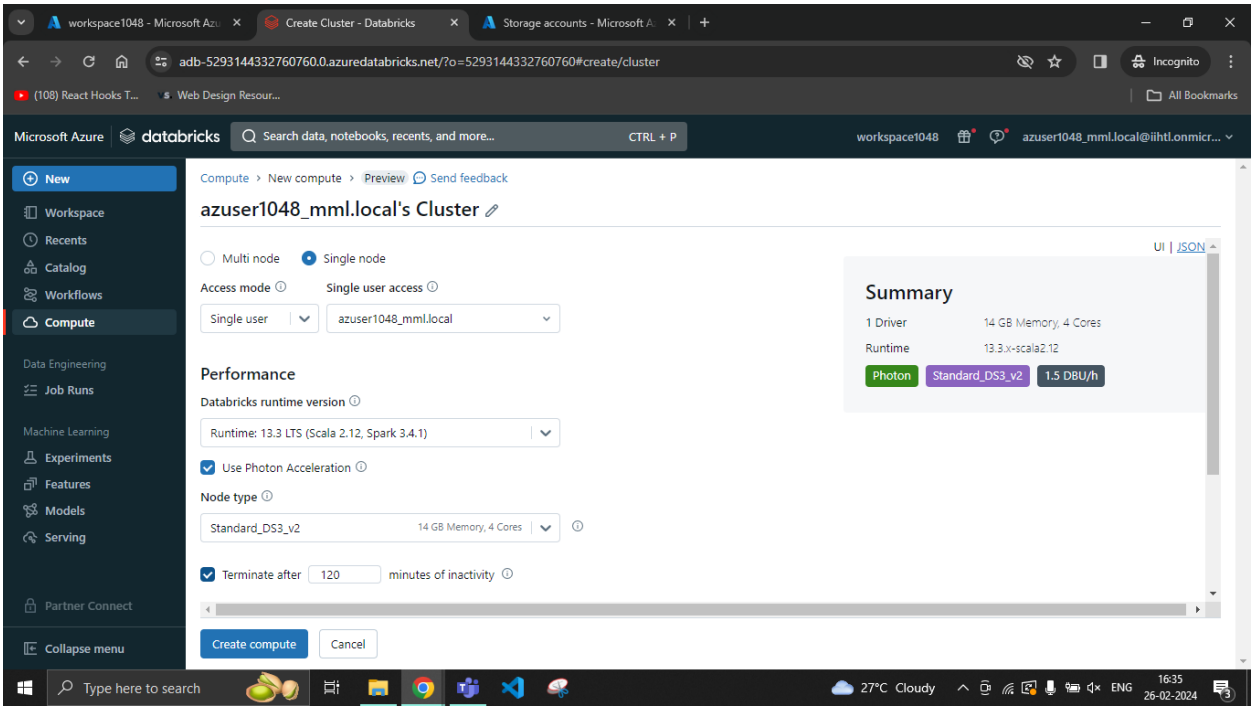
Go to the resource



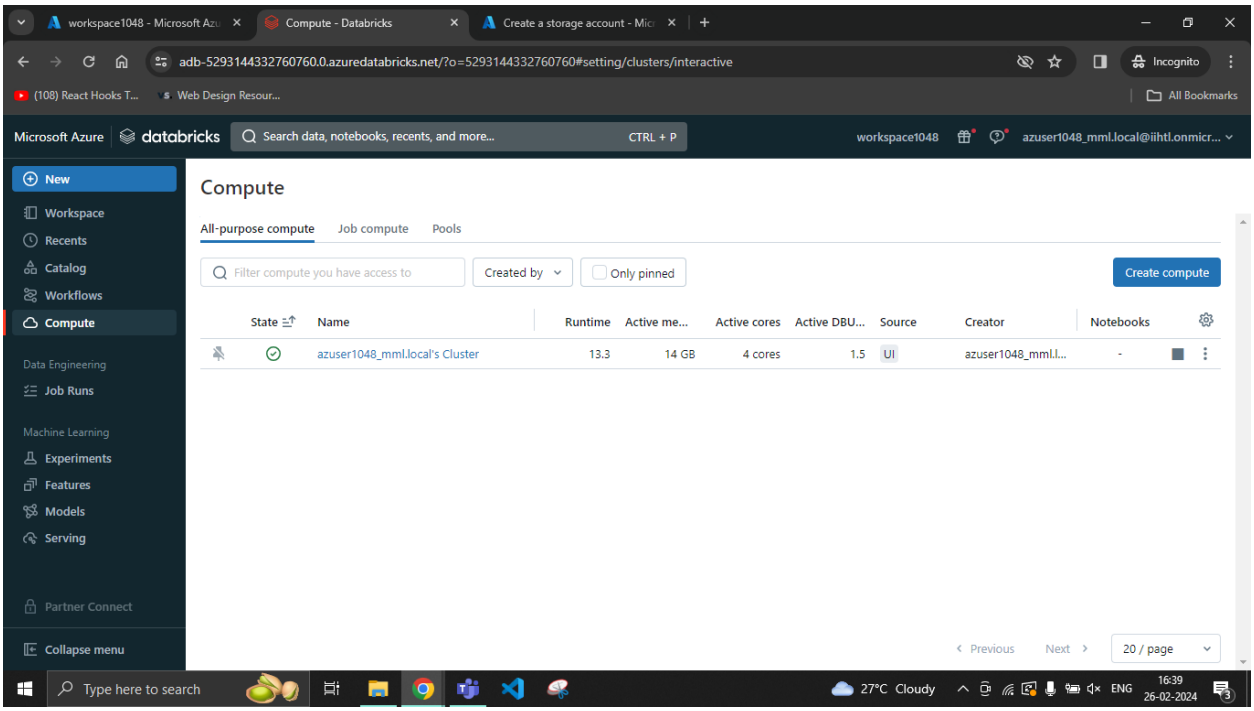
Launch Workspace



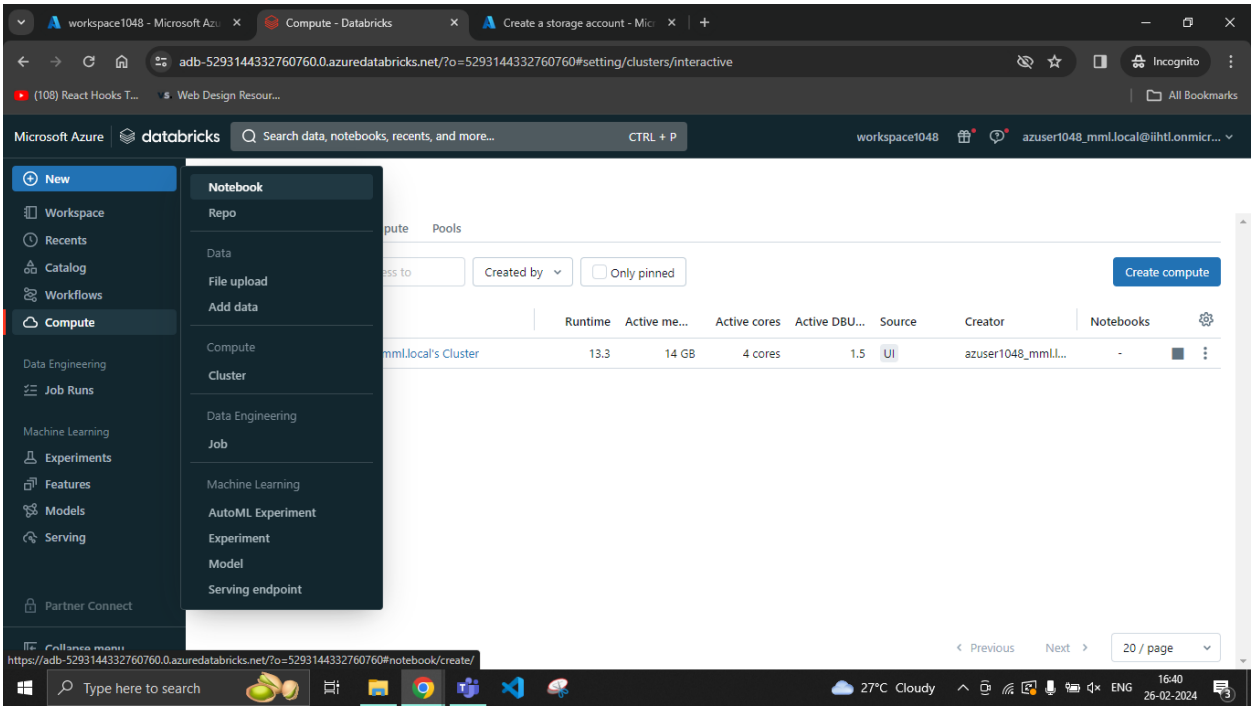
Create a cluster



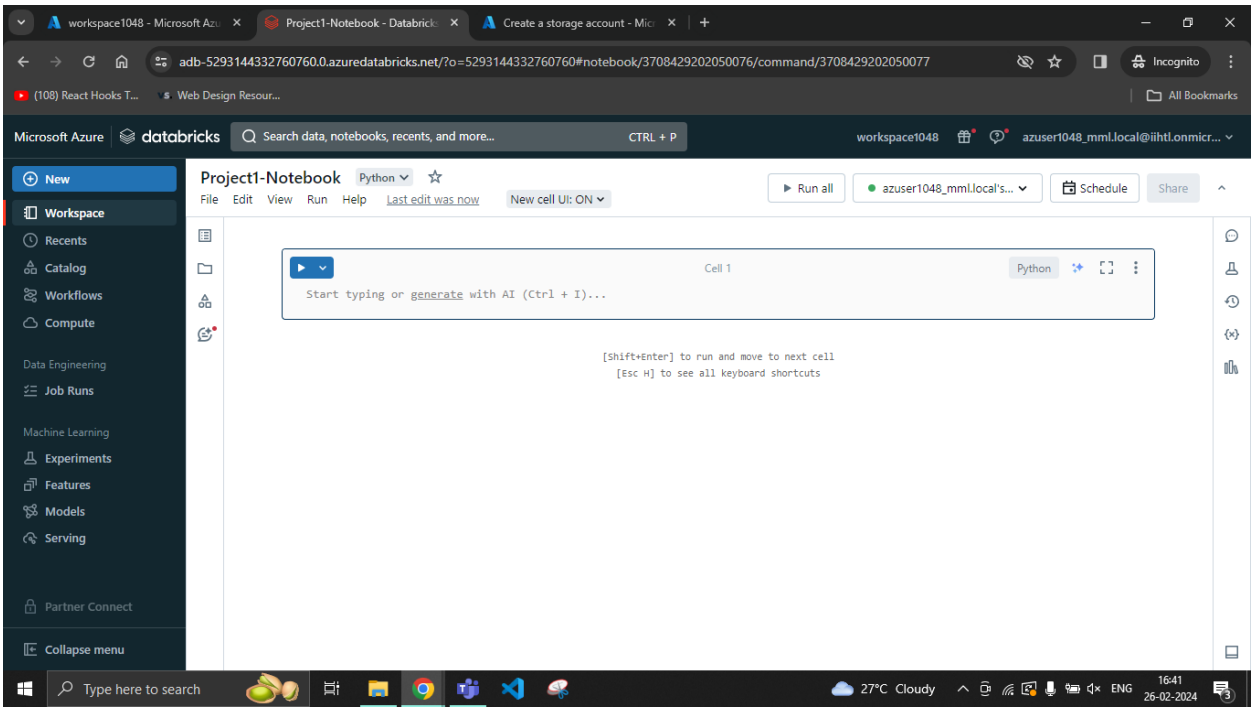
Cluster has been successfully created



Create a new notebook

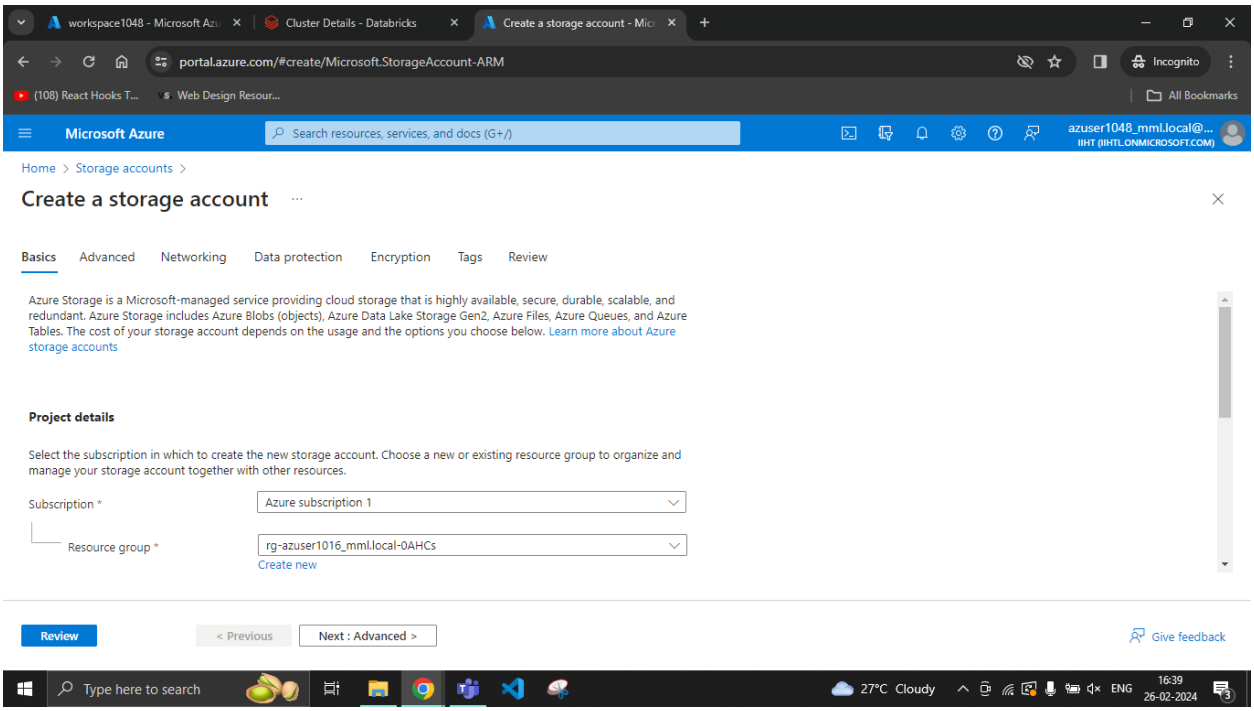


Rename the notebook

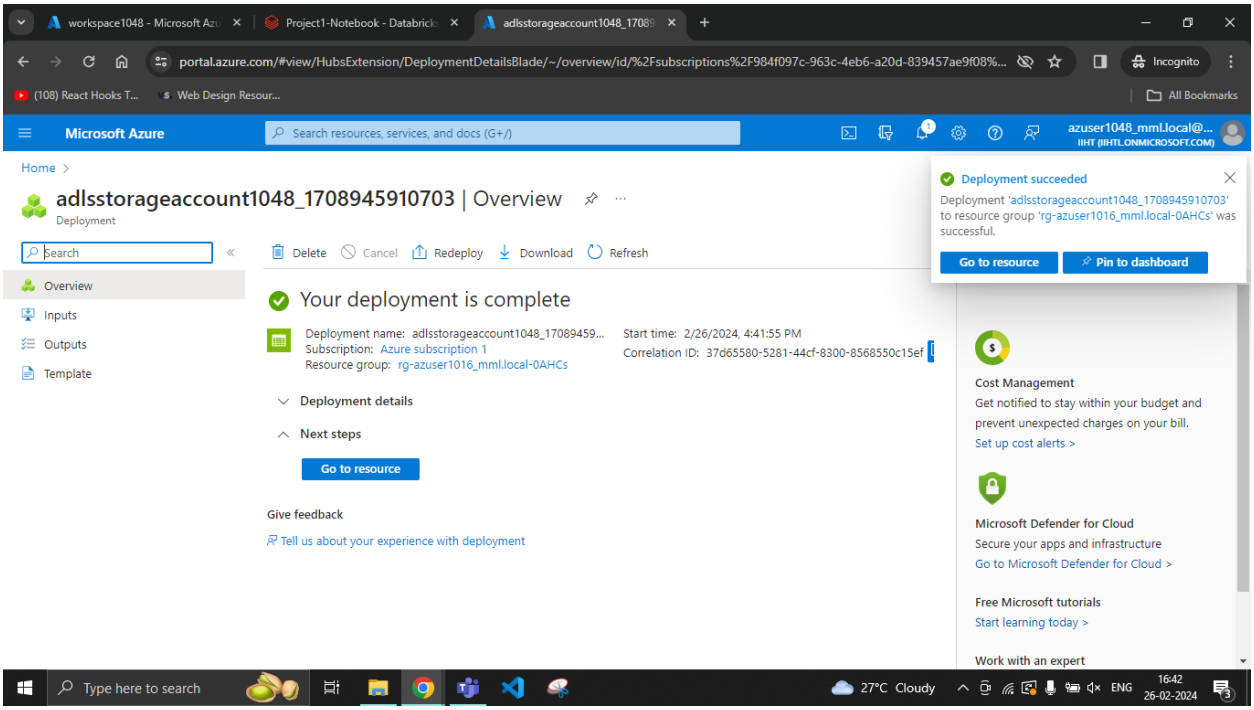


2. Create an Azure Data Lake Gen 2 Storage Account, create a container inside it and upload the raw data.

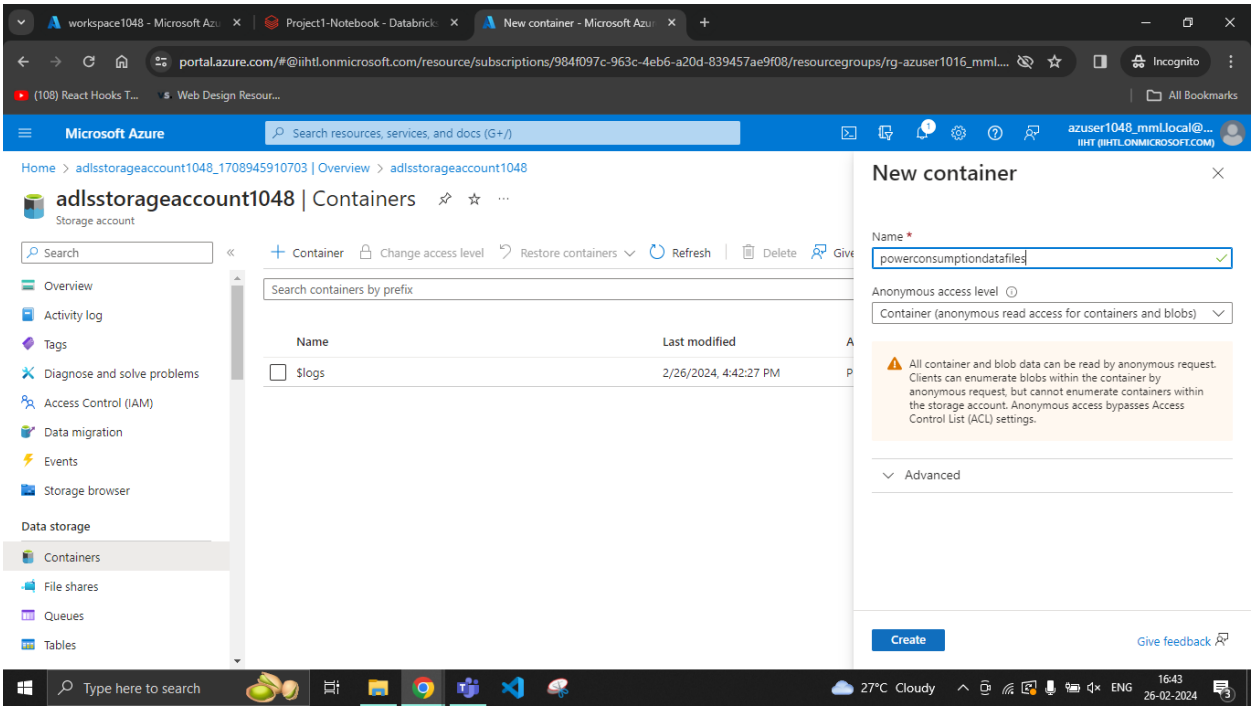
Create Azure Data Lake Storage Account



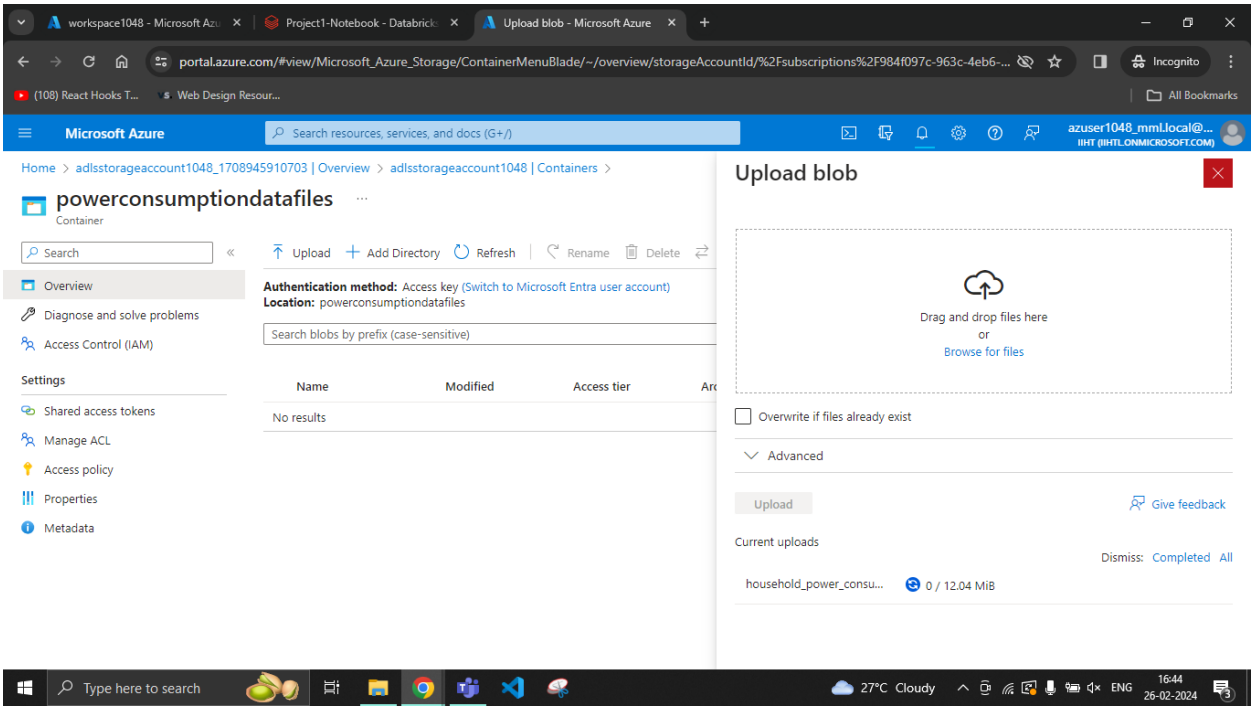
ADLS account has been successfully created. Go to the resource.



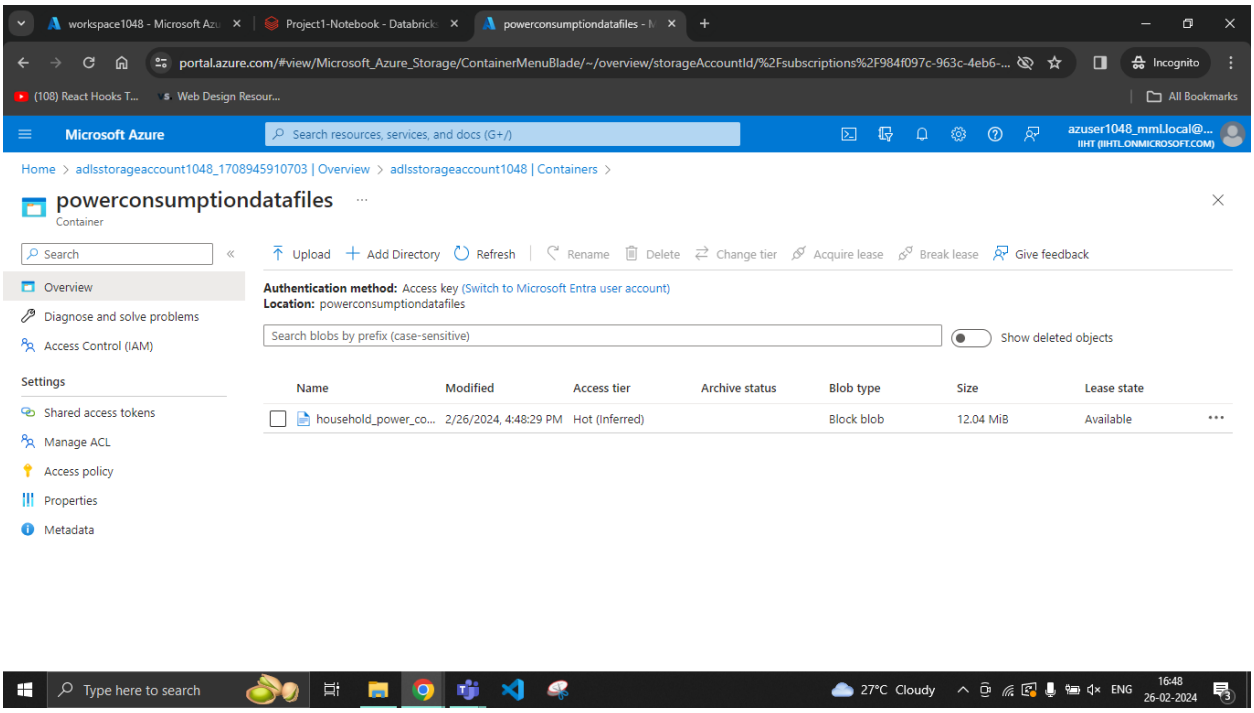
Open ADLS and create a container



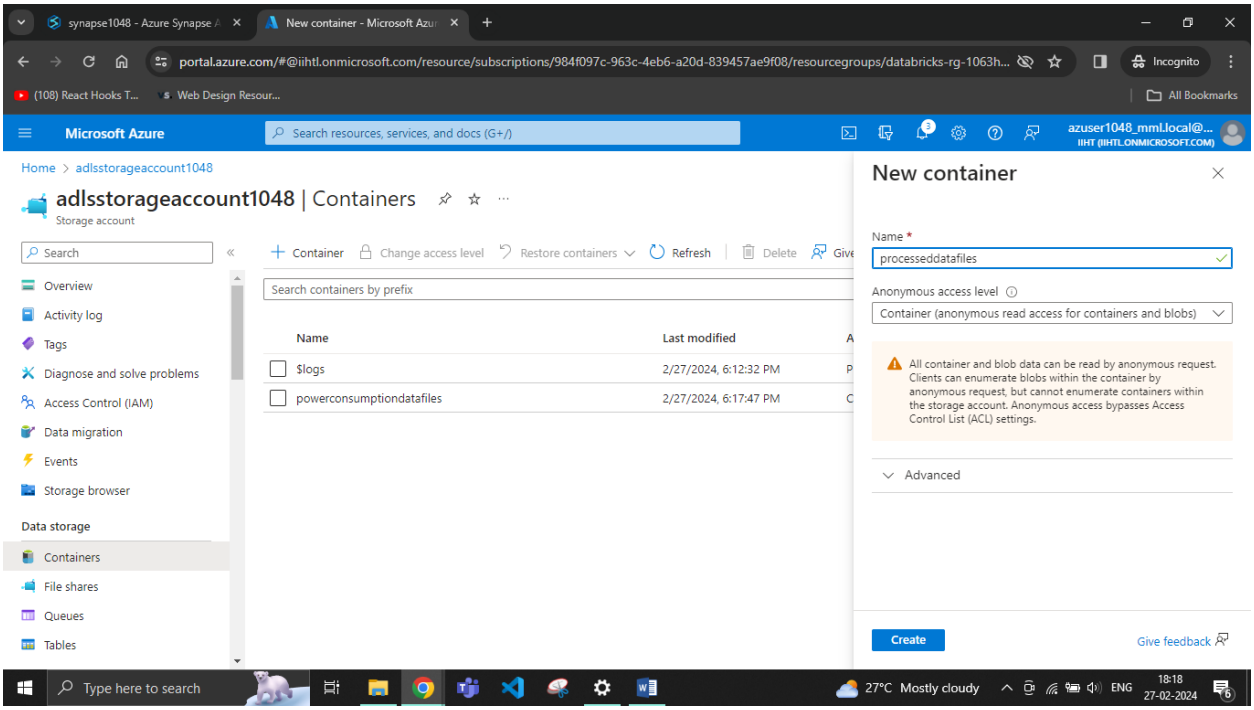
Upload data file in the container



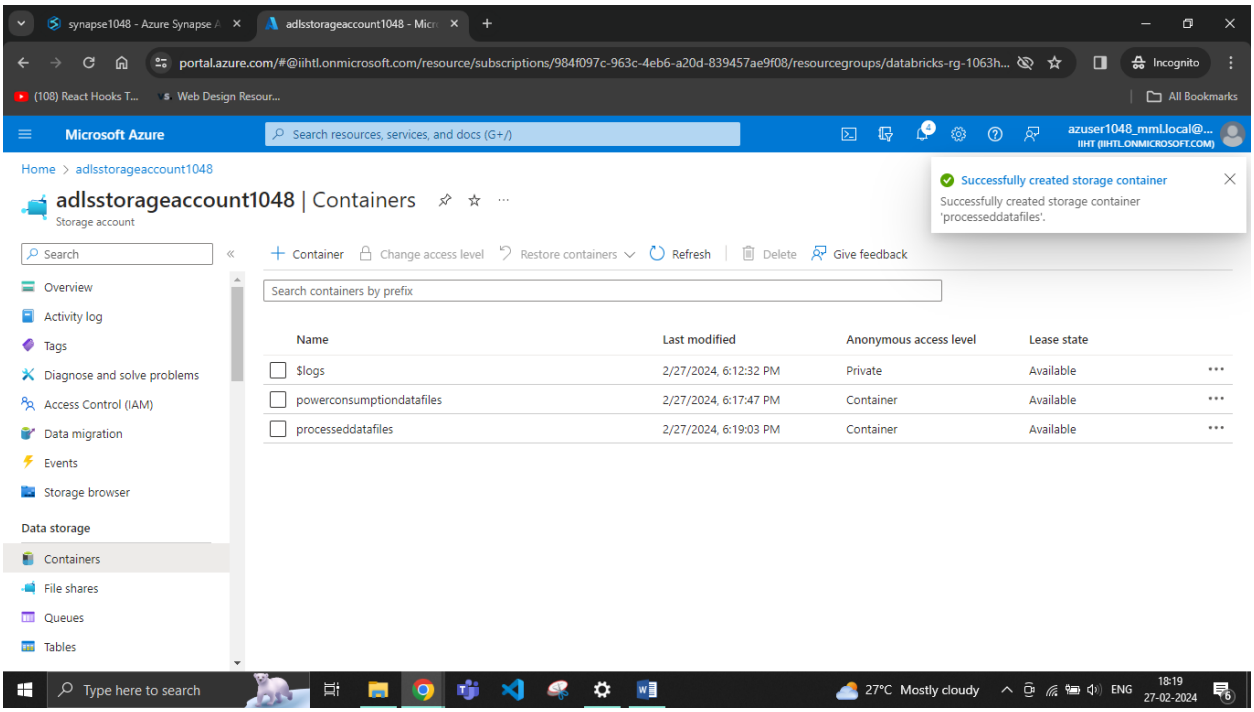
Raw data file has been uploaded successfully



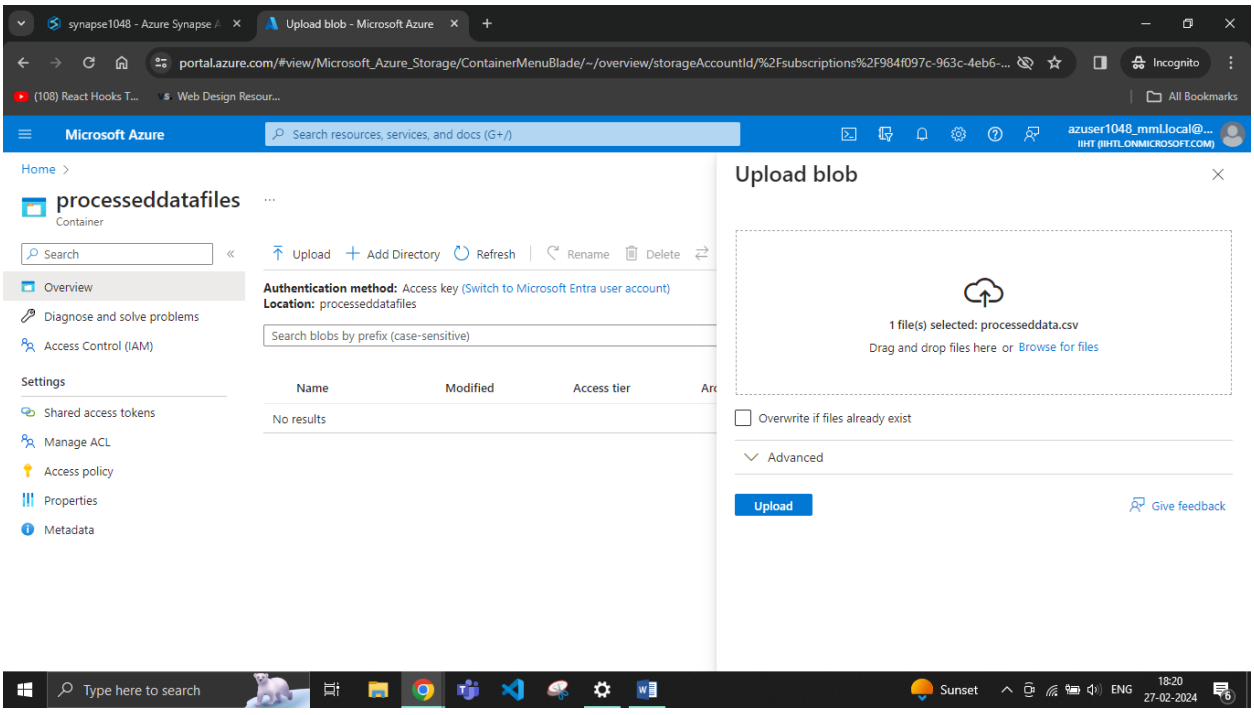
Create another container in which the processed data will be loaded.



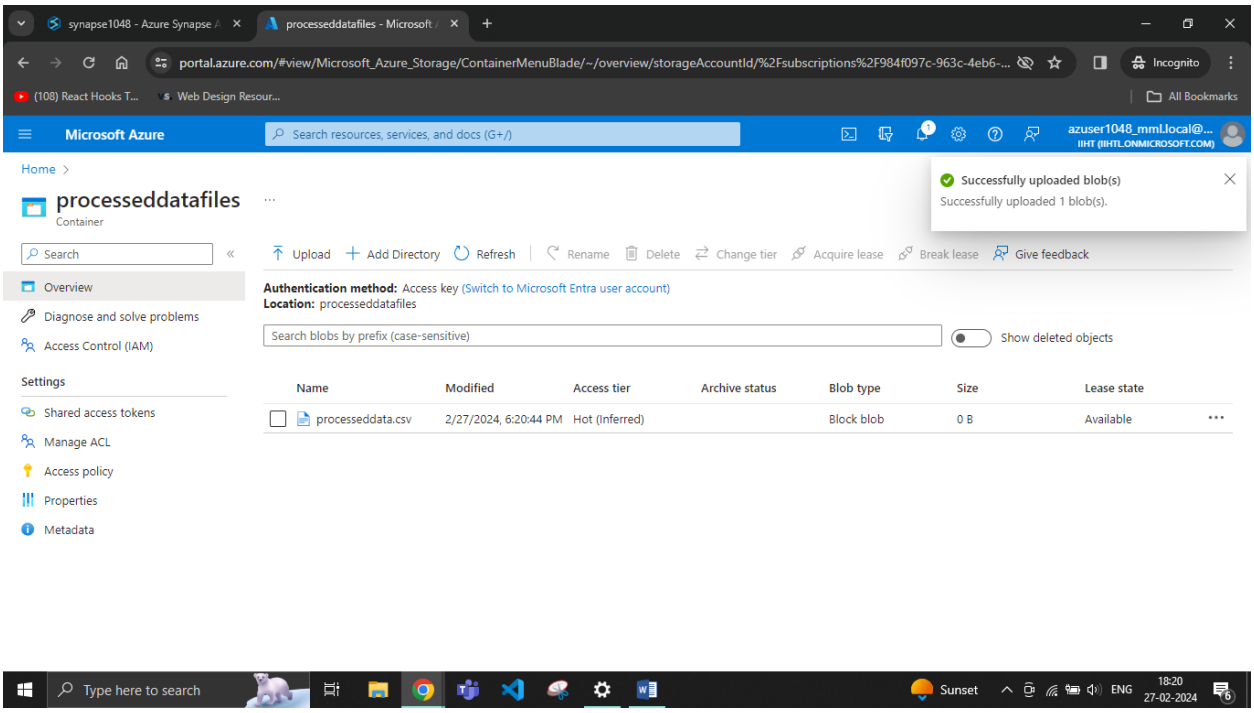
Container successfully created



Upload an empty processed data.csv file in the container.

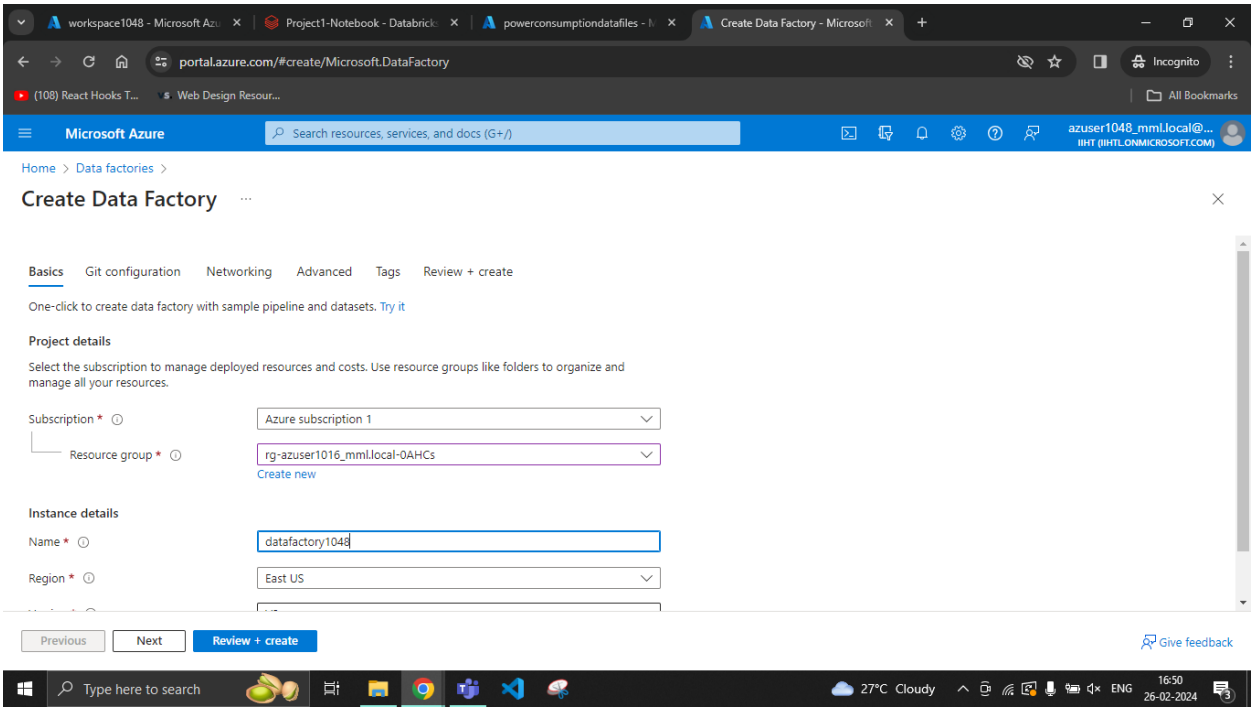


File uploaded successfully

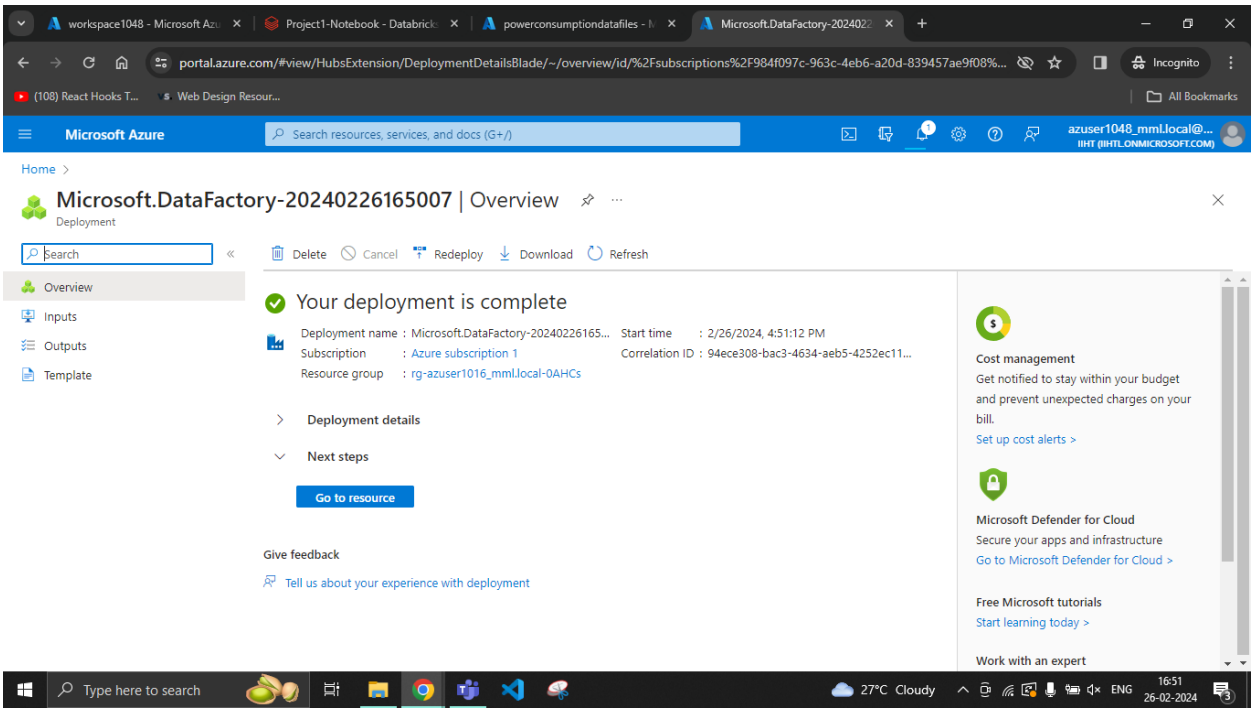


3. Create an Azure Data Factory and create linked services for databricks and ADLS inside it.

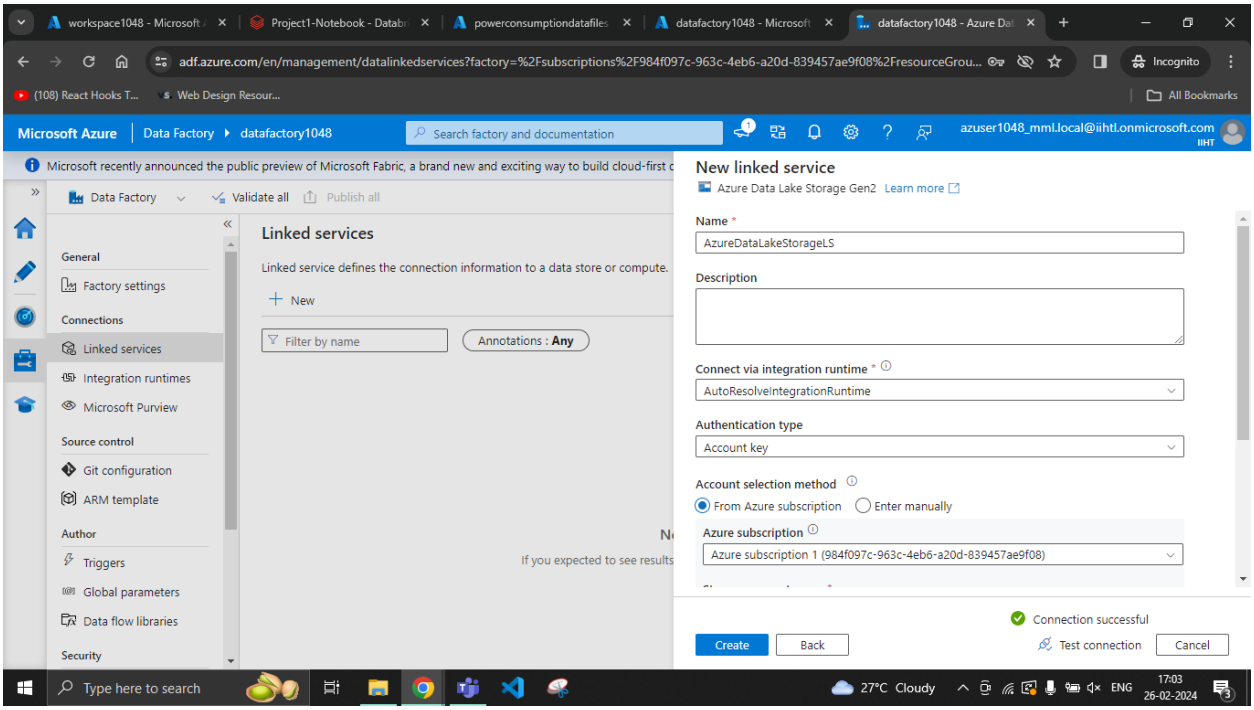
Create data factory



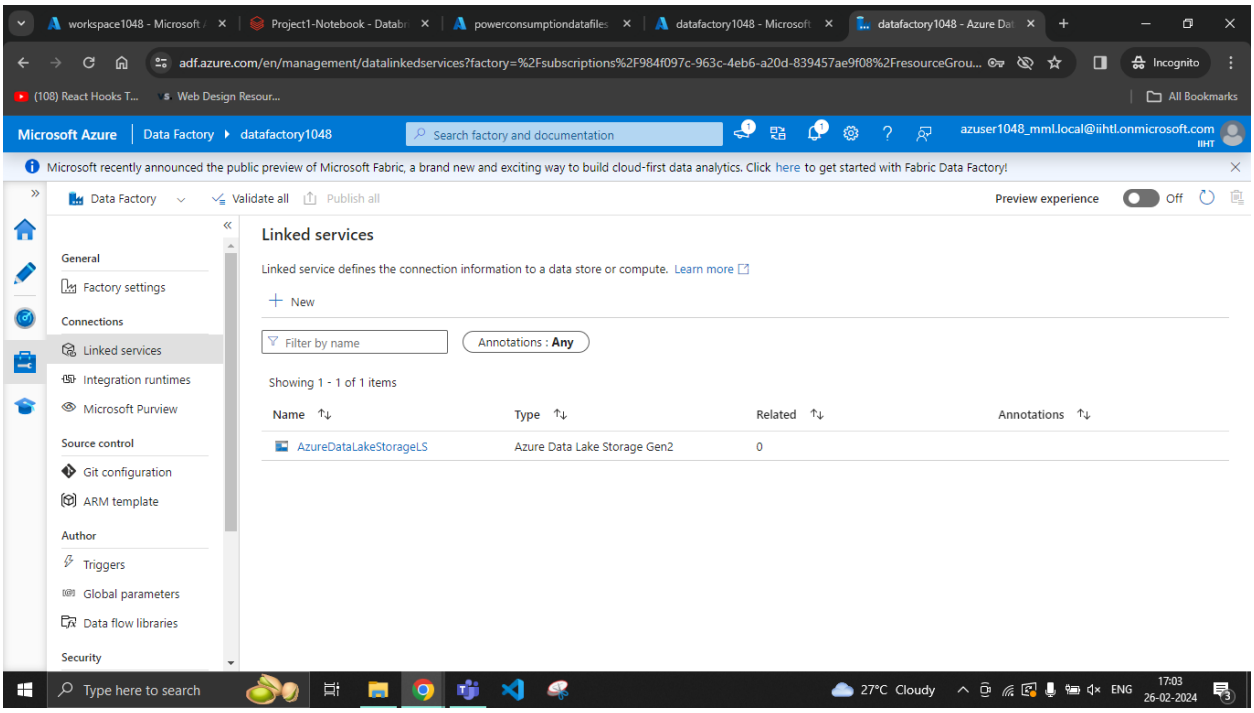
Azure Data Factory has been successfully created. Go to the resource.



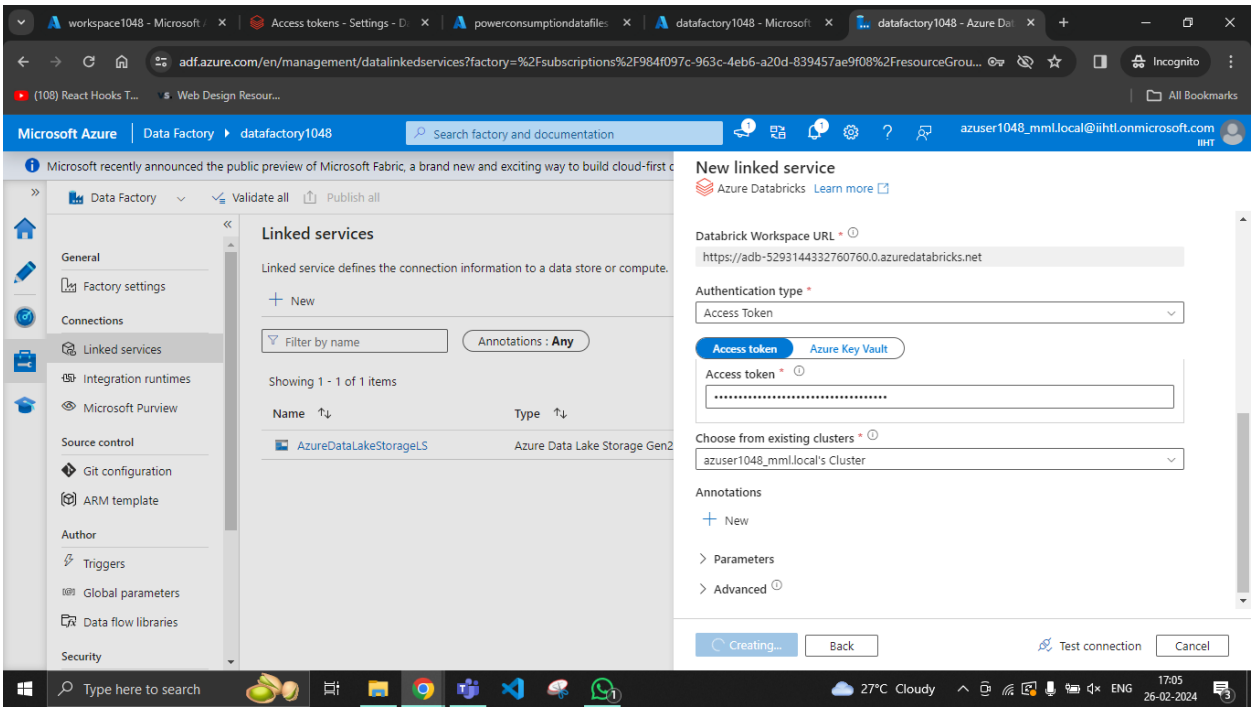
Go to Azure Data Factory and create linked service for ADLS account



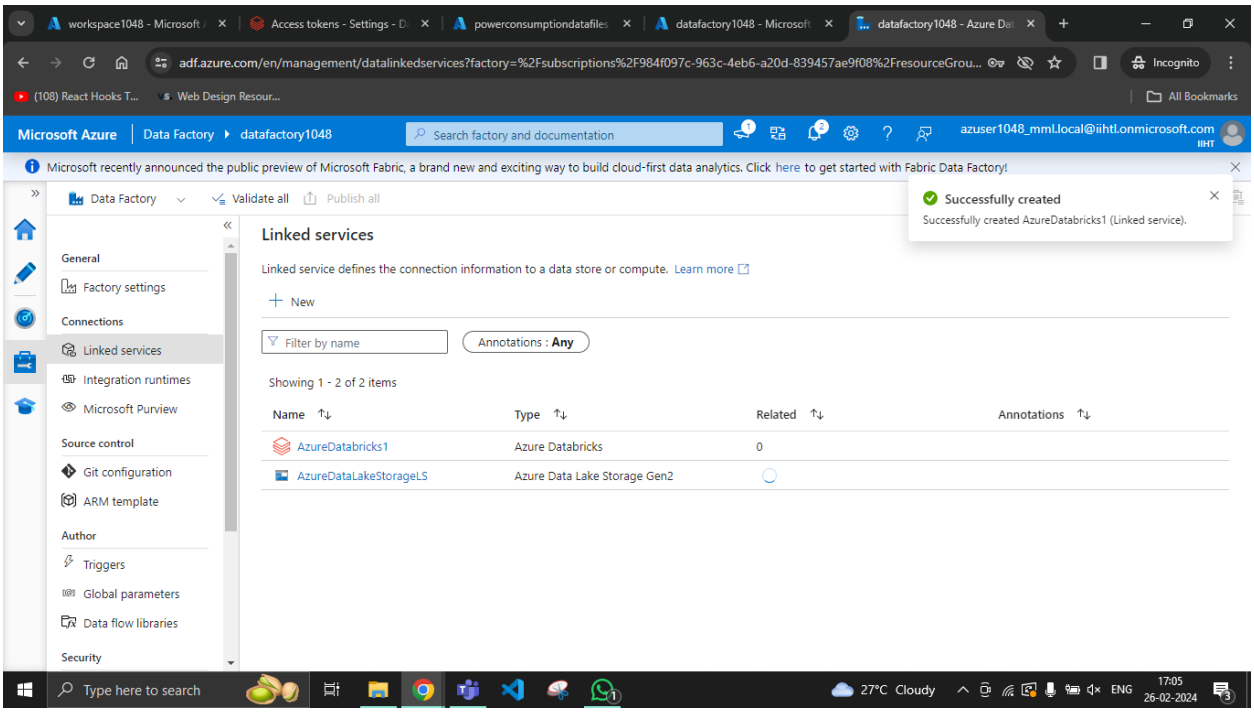
Linked Service for your ADLS Account has been successfully created.



Now create Linked Service for Azure Databricks

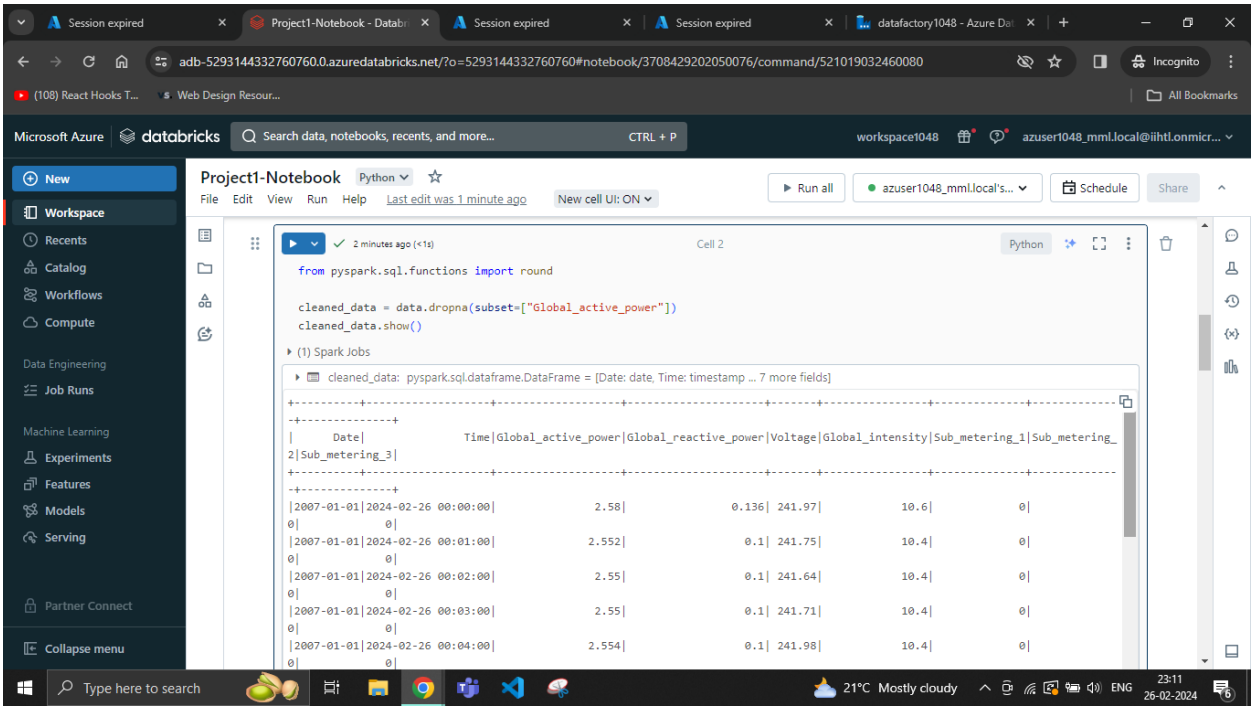


Linked Service for your Databricks has been successfully created.



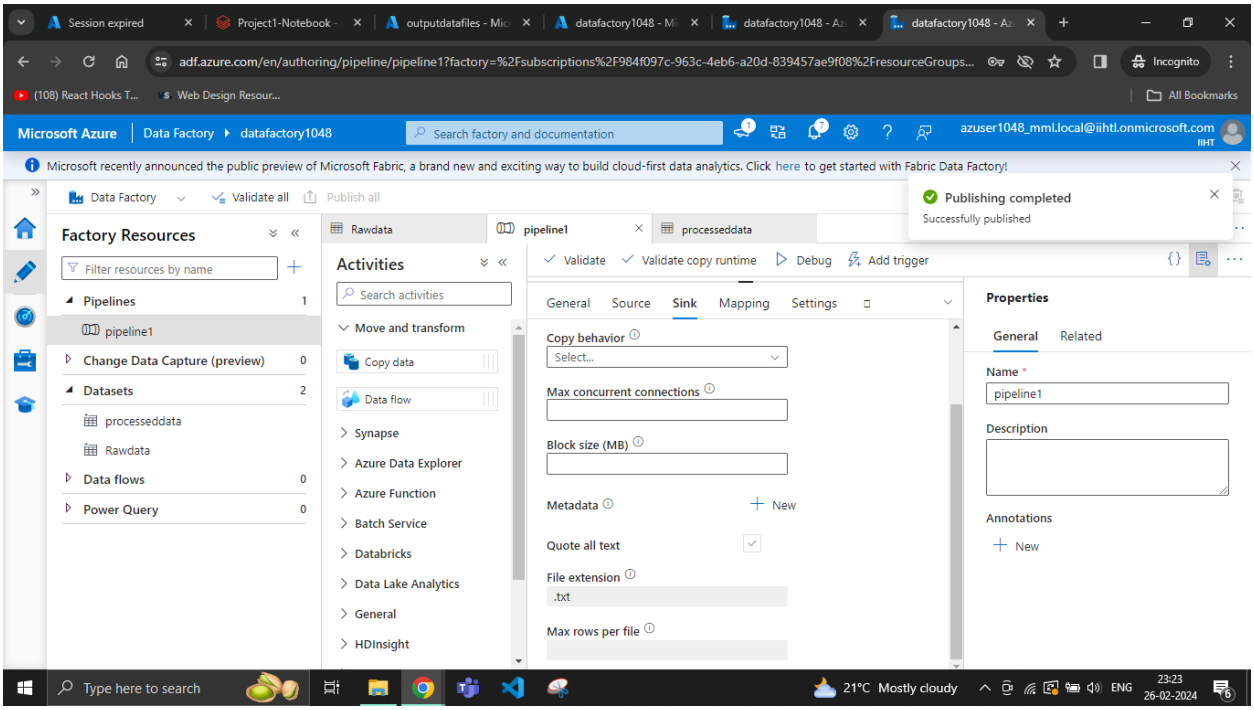
4. Configure the Databricks notebook with the required logic for data cleaning and transformation

In the Databricks notebook write the code for data cleaning and transformation

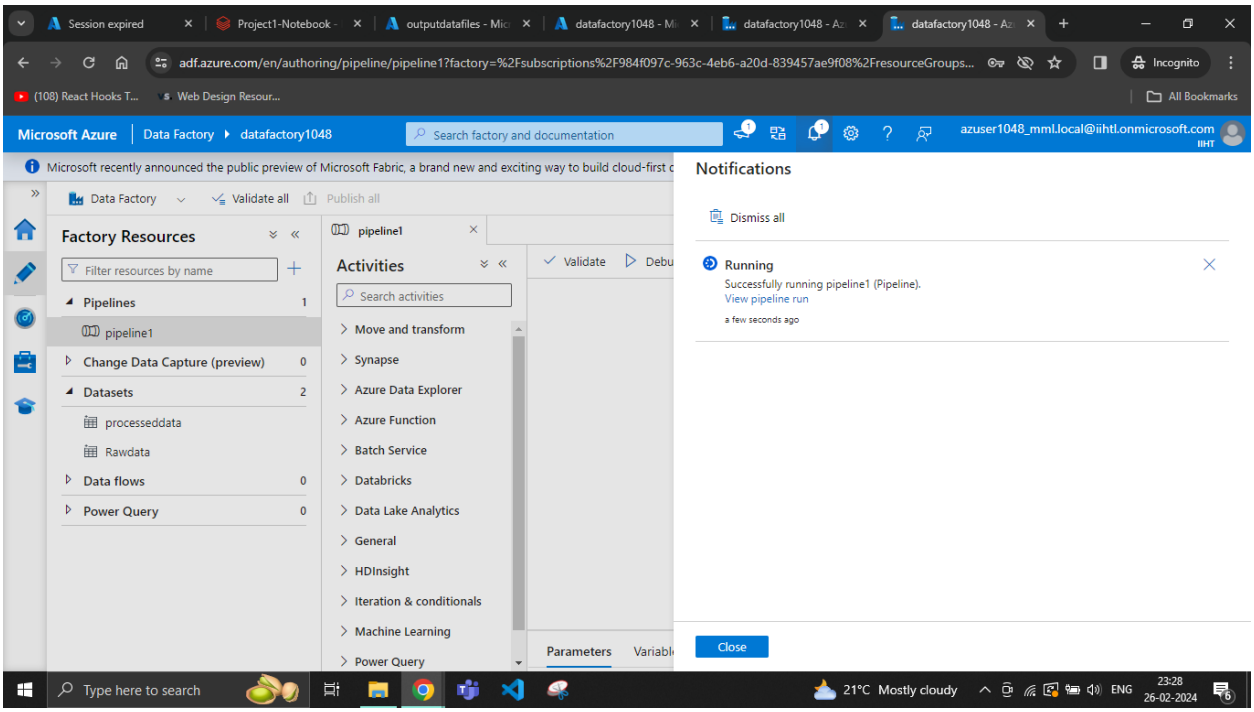


5. Create data pipeline with databricks as activity and trigger the pipeline.

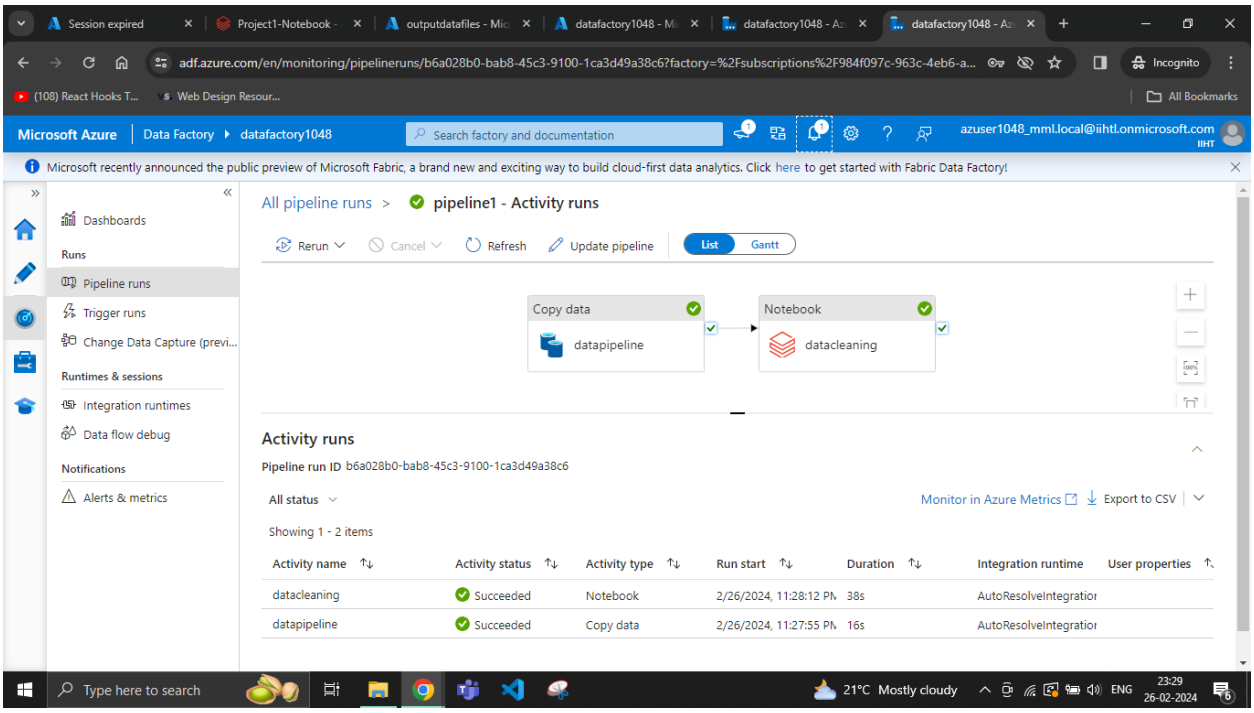
Add copy activity in the pipeline and add databricks notebook as the activity in it



Trigger pipeline run



Pipeline ran successfully



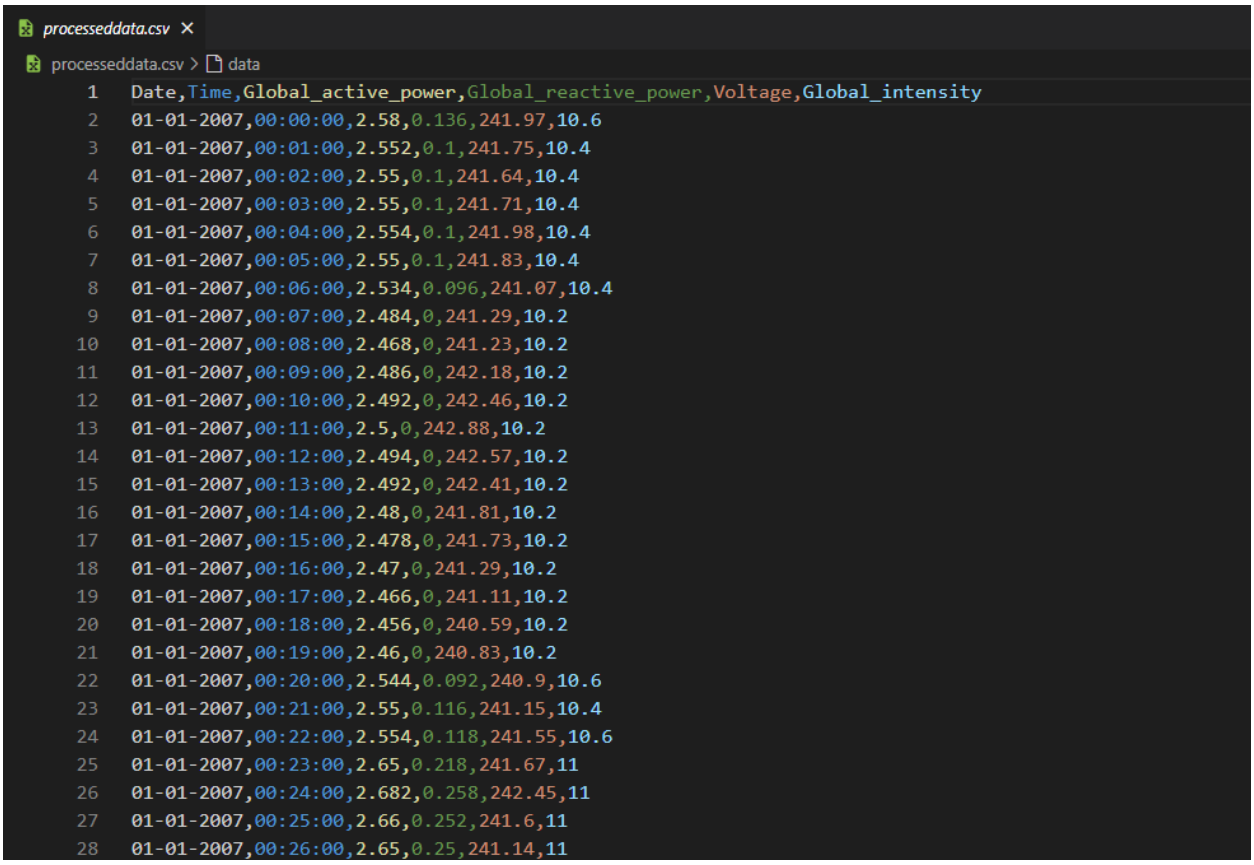
Snapshot of Raw Data from Source:

household_power_consumption.csv

household_power_consumption.csv > data

```
Date,Time,Global_active_power,Global_reactive_power,Voltage,Global_intensity,Sub_metering_1,
Sub_metering_2,Sub_metering_3
01-01-2007,00:00:00,2.58,0.136,241.97,10.6,0,0,0
01-01-2007,00:01:00,2.552,0.1,241.75,10.4,0,0,0
01-01-2007,00:02:00,2.55,0.1,241.64,10.4,0,0,0
01-01-2007,00:03:00,2.55,0.1,241.71,10.4,0,0,0
01-01-2007,00:04:00,2.554,0.1,241.98,10.4,0,0,0
01-01-2007,00:05:00,2.55,0.1,241.83,10.4,0,0,0
01-01-2007,00:06:00,2.534,0.096,241.07,10.4,0,0,0
01-01-2007,00:07:00,2.484,0,241.29,10.2,0,0,0
01-01-2007,00:08:00,2.468,0,241.23,10.2,0,0,0
01-01-2007,00:09:00,2.486,0,242.18,10.2,0,0,0
01-01-2007,00:10:00,2.492,0,242.46,10.2,0,0,0
01-01-2007,00:11:00,2.5,0,242.88,10.2,0,0,0
01-01-2007,00:12:00,2.494,0,242.57,10.2,0,0,0
01-01-2007,00:13:00,2.492,0,242.41,10.2,0,0,0
01-01-2007,00:14:00,2.48,0,241.81,10.2,0,0,0
01-01-2007,00:15:00,2.478,0,241.73,10.2,0,0,0
01-01-2007,00:16:00,2.47,0,241.29,10.2,0,0,0
01-01-2007,00:17:00,2.466,0,241.11,10.2,0,0,0
01-01-2007,00:18:00,2.456,0,240.59,10.2,0,0,0
01-01-2007,00:19:00,2.46,0,240.83,10.2,0,0,0
01-01-2007,00:20:00,2.544,0.092,240.9,10.6,0,0,0
01-01-2007,00:21:00,2.55,0.116,241.15,10.4,0,1,0
01-01-2007,00:22:00,2.554,0.118,241.55,10.6,0,1,0
01-01-2007,00:23:00,2.65,0.218,241.67,11,0,2,0
01-01-2007,00:24:00,2.682,0.258,242.45,11,0,1,0
01-01-2007,00:25:00,2.66,0.252,241.6,11,0,1,0
```

Snapshot of Processed Data from Sink:



processeddata.csv

processeddata.csv > data

```
Date,Time,Global_active_power,Global_reactive_power,Voltage,Global_intensity
01-01-2007,00:00:00,2.58,0.136,241.97,10.6
01-01-2007,00:01:00,2.552,0.1,241.75,10.4
01-01-2007,00:02:00,2.55,0.1,241.64,10.4
01-01-2007,00:03:00,2.55,0.1,241.71,10.4
01-01-2007,00:04:00,2.554,0.1,241.98,10.4
01-01-2007,00:05:00,2.55,0.1,241.83,10.4
01-01-2007,00:06:00,2.534,0.096,241.07,10.4
01-01-2007,00:07:00,2.484,0,241.29,10.2
01-01-2007,00:08:00,2.468,0,241.23,10.2
01-01-2007,00:09:00,2.486,0,242.18,10.2
01-01-2007,00:10:00,2.492,0,242.46,10.2
01-01-2007,00:11:00,2.5,0,242.88,10.2
01-01-2007,00:12:00,2.494,0,242.57,10.2
01-01-2007,00:13:00,2.492,0,242.41,10.2
01-01-2007,00:14:00,2.48,0,241.81,10.2
01-01-2007,00:15:00,2.478,0,241.73,10.2
01-01-2007,00:16:00,2.47,0,241.29,10.2
01-01-2007,00:17:00,2.466,0,241.11,10.2
01-01-2007,00:18:00,2.456,0,240.59,10.2
01-01-2007,00:19:00,2.46,0,240.83,10.2
01-01-2007,00:20:00,2.544,0.092,240.9,10.6
01-01-2007,00:21:00,2.55,0.116,241.15,10.4
01-01-2007,00:22:00,2.554,0.118,241.55,10.6
01-01-2007,00:23:00,2.65,0.218,241.67,11
01-01-2007,00:24:00,2.682,0.258,242.45,11
01-01-2007,00:25:00,2.66,0.252,241.6,11
01-01-2007,00:26:00,2.65,0.25,241.14,11
```

Conclusion:

In conclusion, this project successfully demonstrated parallel processing using Azure Data Factory's Data Pipeline for processing, analysis and movement of household power consumption data. This project has successfully developed a robust and efficient data pipeline in Azure for processing and analyzing household power consumption data. The data was ingested from the source by Azure Data Factory, cleaned and transformed by the Azure Databricks Notebook and the loaded to ADLS with the copy activity in the data pipeline.