**REAL TIME PROCESSING**

* **Sharath Chandra**

**Project Overview:**

The project aims to build a real-time data processing solution leveraging Azure Data Factory for ingesting streaming data and Azure Databricks for processing and analysing the data in near-real-time using Spark streaming.

**About the Project:**

Our project aims to implement a robust solution for real-time data processing, leveraging the capabilities of Azure Data Factory (ADF) for data ingestion and Azure Databricks for near-real-time processing and analysis using Spark streaming.

**Key Components:**

* **Azure Data Factory (ADF):** ADF serves as the data ingestion and orchestration tool, facilitating the seamless collection of streaming data from diverse sources such as IoT devices, social media feeds, or application logs. Through ADF's connectors and pipelines, data is efficiently ingested into the Azure ecosystem with minimal latency.
* **Azure Databricks:** Databricks is utilized for near-real-time data processing and analysis. Leveraging the power of Apache Spark, Databricks enables continuous computation on the incoming data streams, allowing for real-time insights and analytics. Spark's in-memory processing ensures efficient handling of large volumes of data with low latency, enabling timely decision-making.

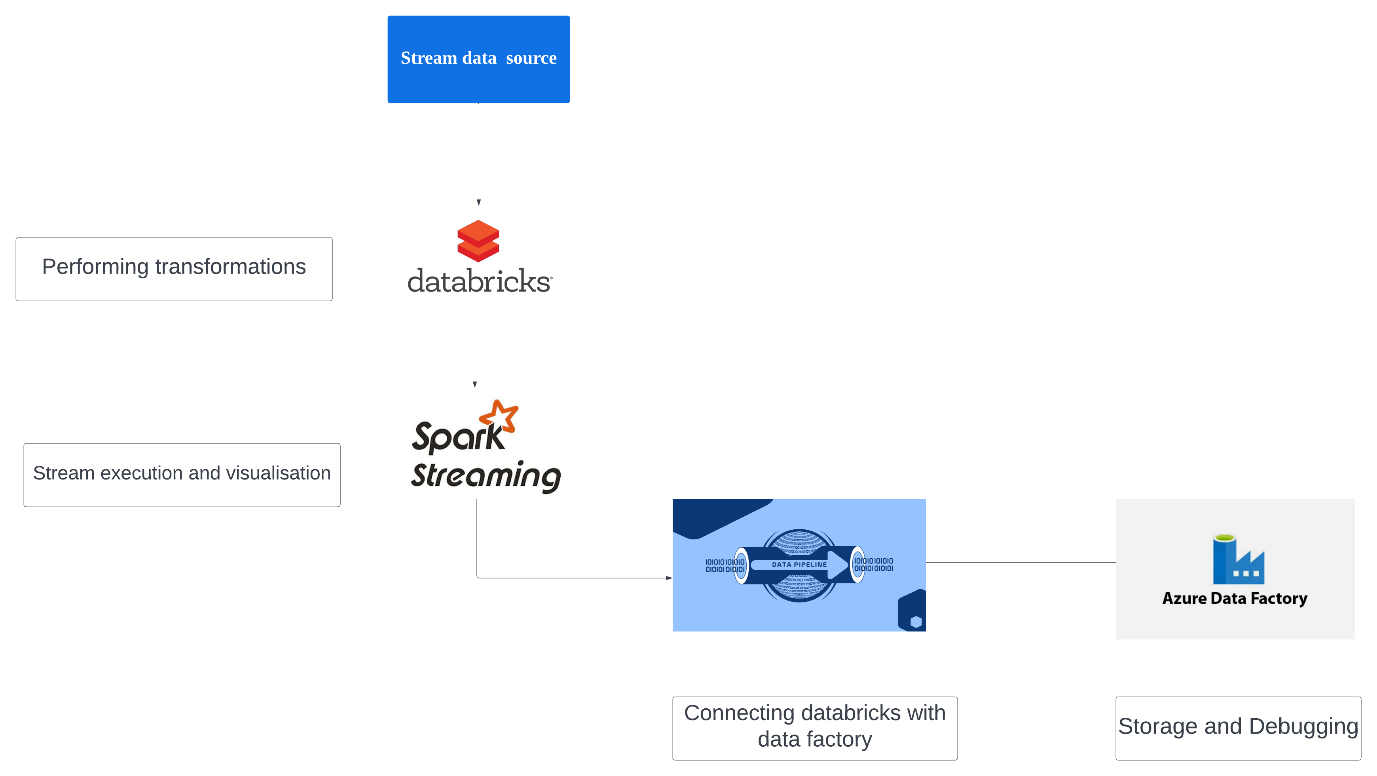
**Uses in Real Time:**

Data Factory ensures that data flows seamlessly from source to destination in real-time, allowing businesses to stay up-to-date with the latest information.

Databricks enables organizations to analyze and derive insights from streaming data as it arrives, empowering them to make informed decisions and take immediate actions based on the most current data available.

Together, Data Factory and Databricks form a powerful duo for real-time data processing, helping businesses stay agile, responsive, and competitive in today's fast-paced digital landscape.

**Architecture Diagram:**



**Source Date Files**

In this project we are creating a delta table using the azure databrick notebook

We are creating the delta table in the local azure storage



**Project Requirements:**

**Functional Requirements:**

* **Streaming Data Ingestion:** Azure Data Factory should support ingestion of streaming data from diverse sources such as IoT devices, web applications, or any other relevant sources.
* **Data Preprocessing:** Implement data preprocessing tasks within Azure Data Factory to cleanse, transform, and enrich the incoming streaming data. This includes tasks like data normalization, filtering, and aggregations.
* **Integration with Azure Databricks:** Establish seamless integration between Azure Data Factory and Azure Databricks to transfer pre-processed streaming data for further analysis**.**
* **Real-time Processing with Databricks:** Utilize Azure Databricks' capabilities for real-time processing using Spark streaming to enable near-real-time analysis.
* **Scalability**: Design the solution to be highly scalable to handle varying data volumes and processing demands.

**Non-Functional Requirements:**

* **Reliability:** The system should be highly reliable, ensuring minimal downtime and data loss during ingestion, preprocessing, and analysis.
* **Latency:** Ensure low latency for data processing and analysis to enable near-real-time insights and decision-making.
* **Security:** Implement robust security measures to protect data during transit and at rest, adhering to industry-standard encryption and access control protocols.
* **Maintainability**: The system should be easy to maintain and update, with clear documentation and well-defined processes for troubleshooting and monitoring.
* **Scalability**: Design the solution to scale horizontally and vertically to accommodate increasing data volumes and processing requirements without compromising performance.

**How it works:**

**Azure Databricks and Spark Stream Visualization:**

Step 1: Set up an Azure Databricks workspace in the Azure portal.

Step 2: Create a new Databricks cluster with the desired configuration.

Step 3: Open the Databricks workspace and create a new notebook.

Step 4: Write and execute Spark streaming code in the notebook to process streaming data.

Step 5: Visualize the streaming data using Databricks visualization spark.readStream

**Opening Azure Data Factory (ADF) and Providing Access Key:**

Step 6: Navigate to the Azure portal and search for Azure Data Factory.

Step 7: Open the Azure Data Factory instance associated with your project.

Step 8: Access the linked services section and configure a linked service for Azure Databricks.

Step 9: Provide the necessary access key or authentication details to establish a connection with Azure Databricks.

Step 10: Use the linked service in ADF pipelines to trigger Databricks notebooks or jobs for processing streaming data.

**Azure Resources Used for this Project:**

**Azure Data Factory:**

* Azure Data Factory is a cloud-based data integration service that allows you to create, schedule, and manage data pipelines for ingesting, transforming, and moving data across various sources and destinations.
* It provides capabilities for orchestrating complex data workflows, monitoring pipeline executions, and integrating with other Azure services like Azure Databricks and Azure Blob Storage.

**Azure Blob Storage:**

* Azure Blob Storage is a scalable object storage service that allows you to store and manage unstructured data in the cloud.
* It provides high availability, durability, and security for storing various types of data, including documents, images, videos, and data files used in data processing pipelines.

**Azure Databricks:**

* Azure Databricks is an Apache Spark-based analytics platform optimized for the Microsoft Azure cloud.
* It provides a collaborative workspace for data engineers, data scientists, and analysts to build and run Spark-based data processing and machine learning workflows at scale.

**Delta Lake or Delta Table:**

* Delta Lake is an open-source storage layer that brings ACID transactions, data versioning, and schema enforcement to Apache Spark and big data workloads.
* It enables reliable and efficient data processing pipelines by providing features like scalable metadata handling, time travel, and schema evolution.

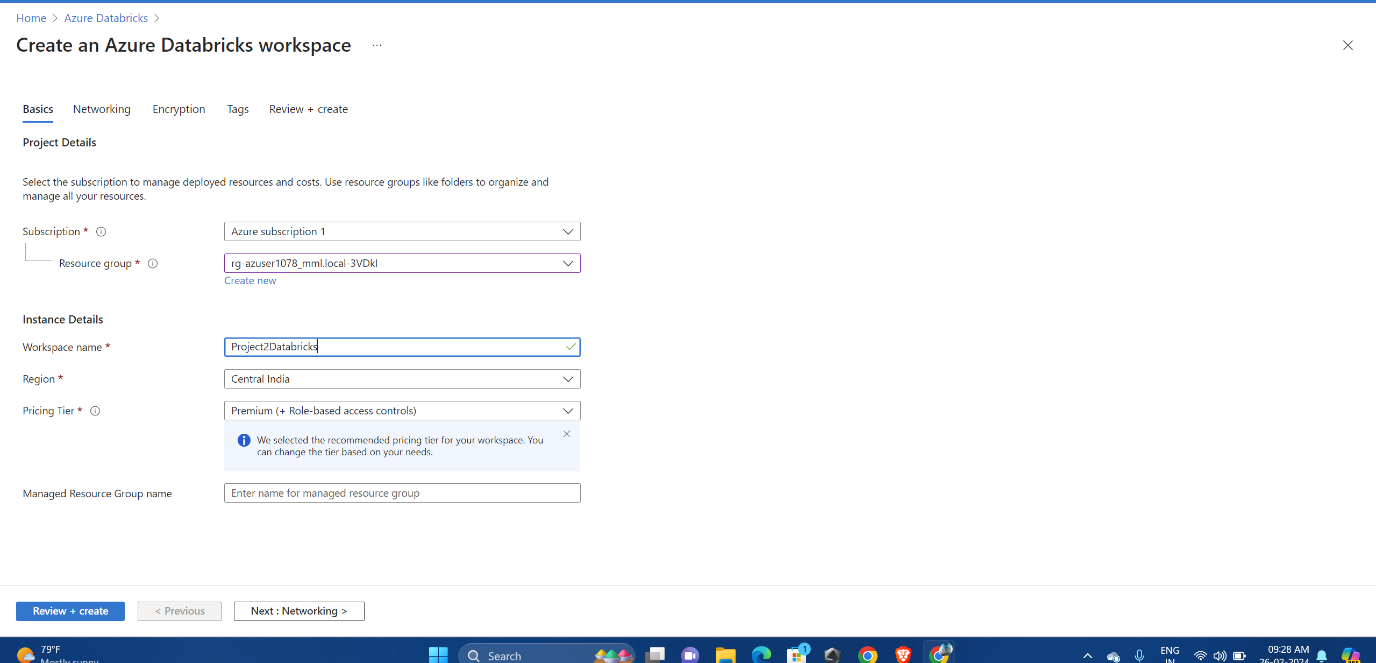
**Spark Streaming:**

* Spark Streaming is an extension of the core Spark API that enables scalable, high-throughput, fault-tolerant stream processing of live data streams.
* It allows you to process and analyze real-time data streams using the same programming model as batch processing, making it easy to build end-to-end data processing pipelines for streaming data.

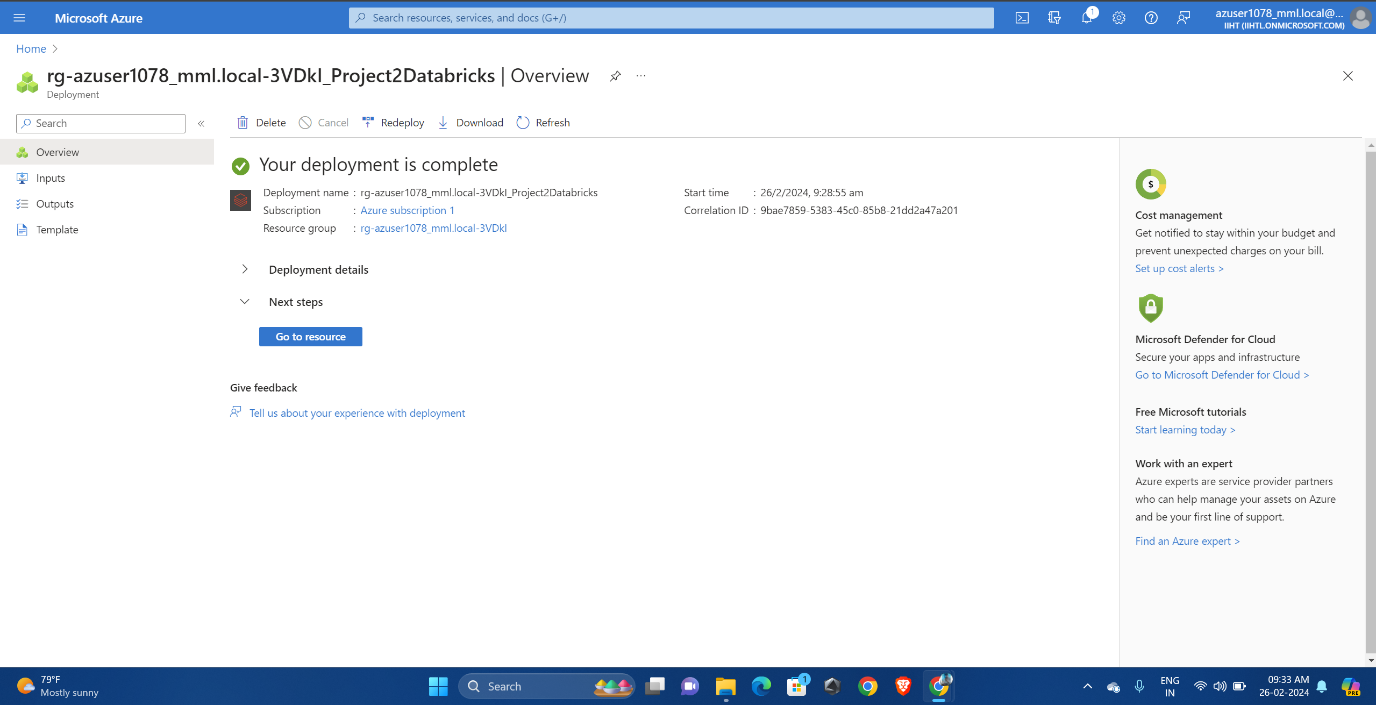
**Implementation of the Project with results:**

1. **Creating Azure Databricks account:**

* Open your azure account and search Azure Databricks.
* Click on Create

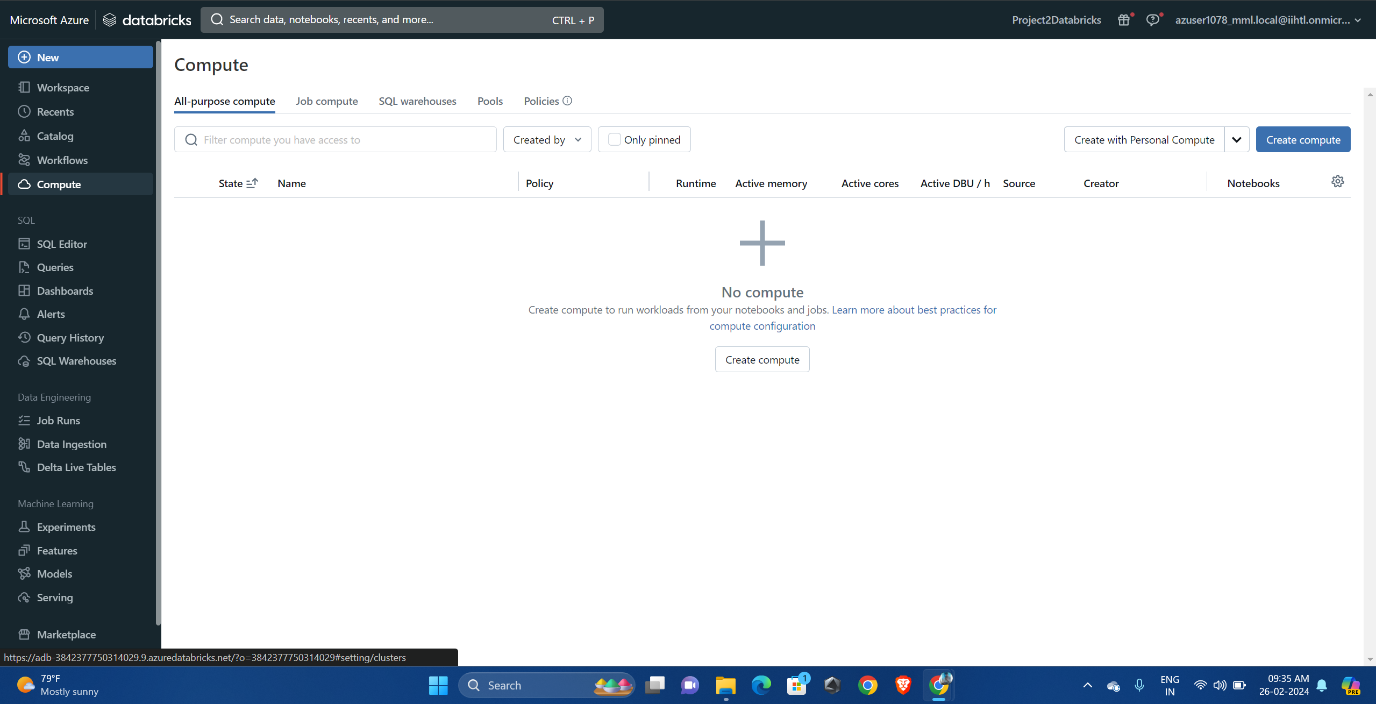


* Databricks Deployment is successful. Now click on ‘Go to resource’

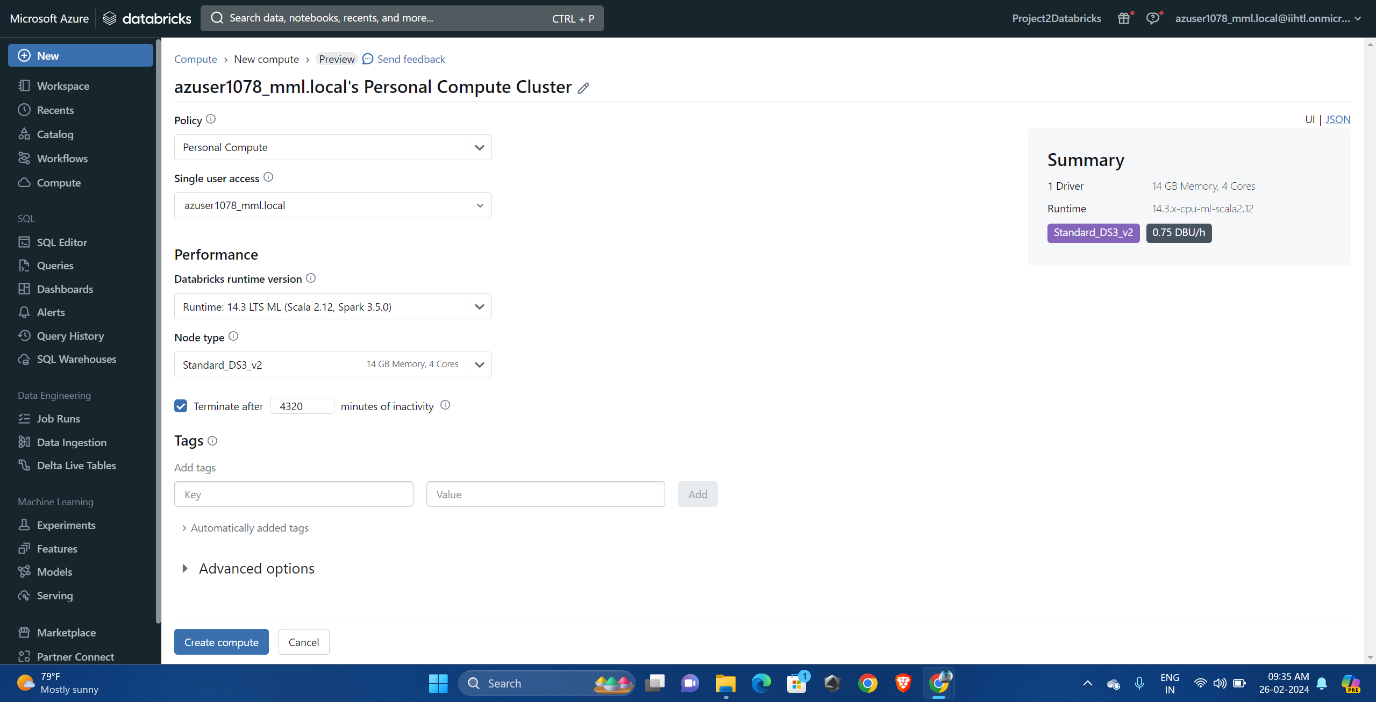


**Creating a Cluster :**

* After opening the workspace go to the compute session
* Click on the ‘Create with Personal Compute’

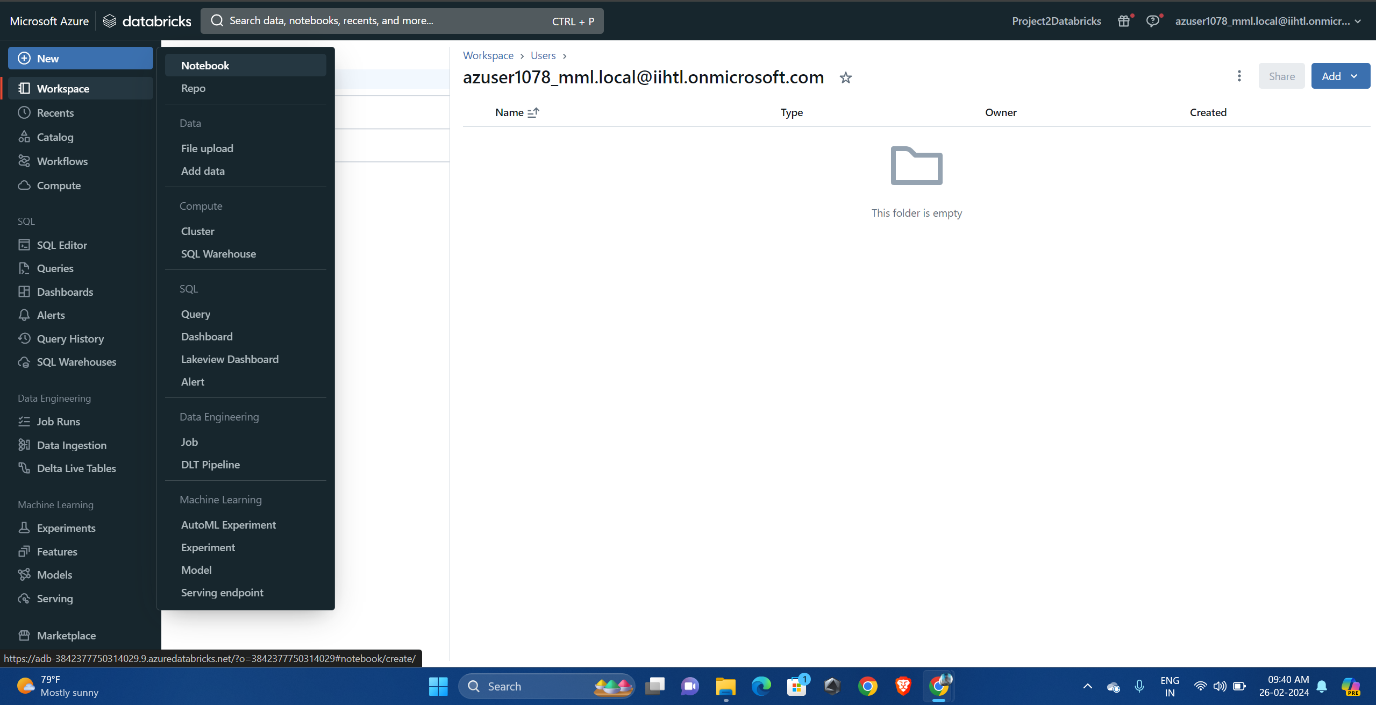


* Provide the cluster name, version, termination period



**Azure Notebook:**

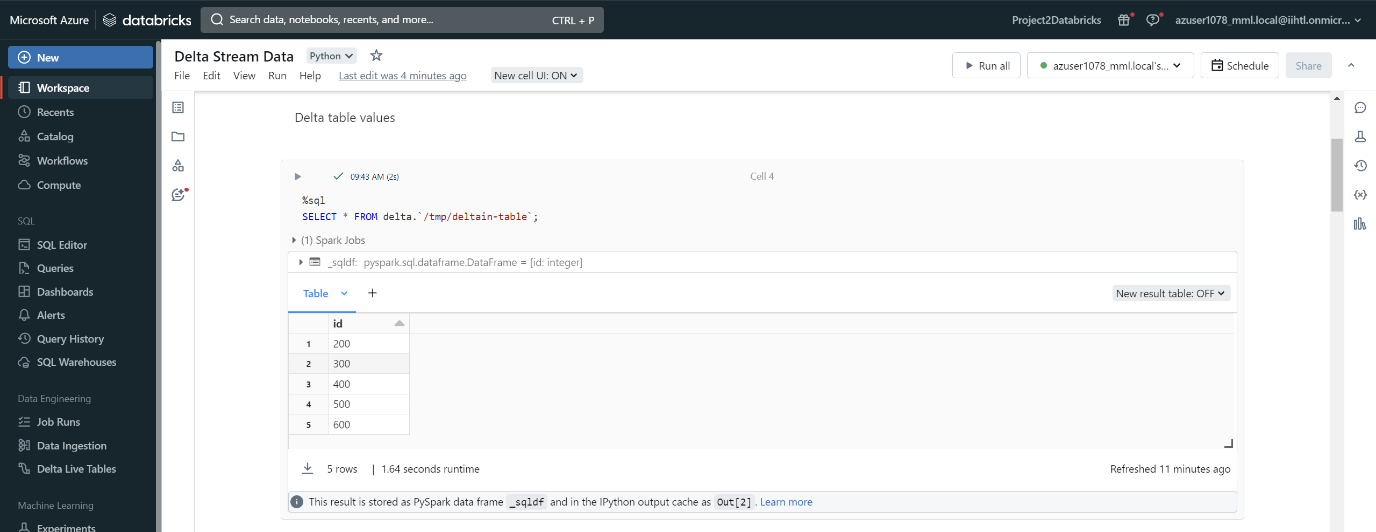
* Click on new and select notebook then a notebook will be created



1. Data Source:

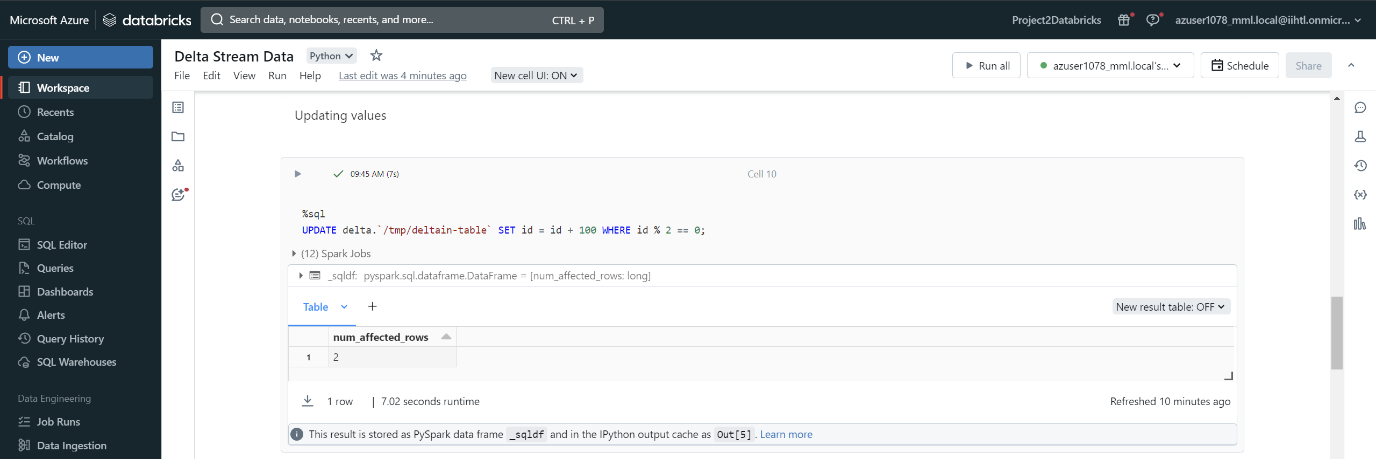
* We will be creating a Delta table and inserting the values:



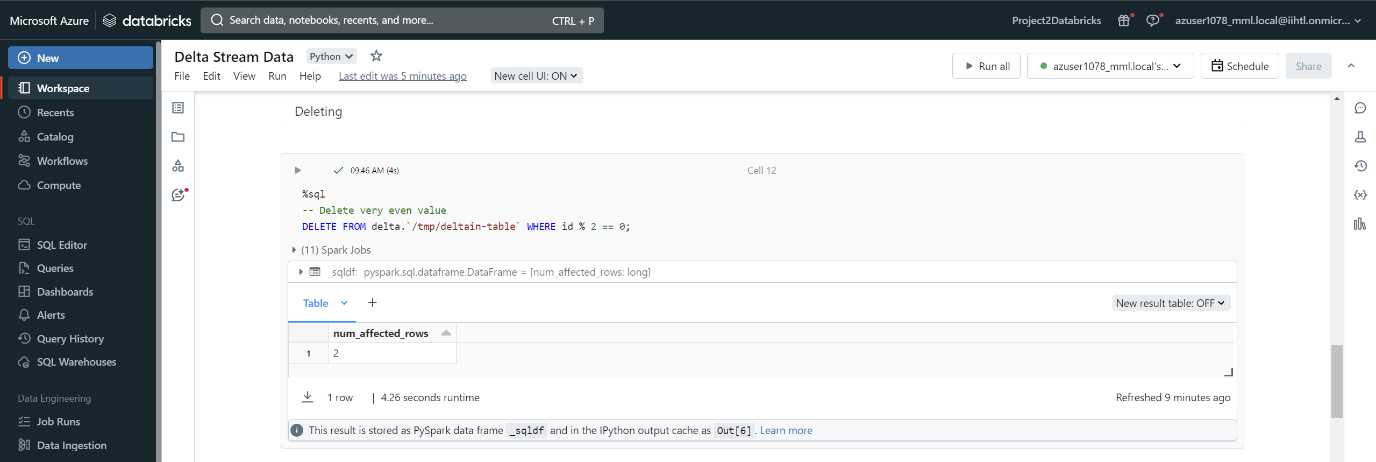


**Performing Transformations:**

* We will be performing the transformations like updating and Deleting operations
* Update:

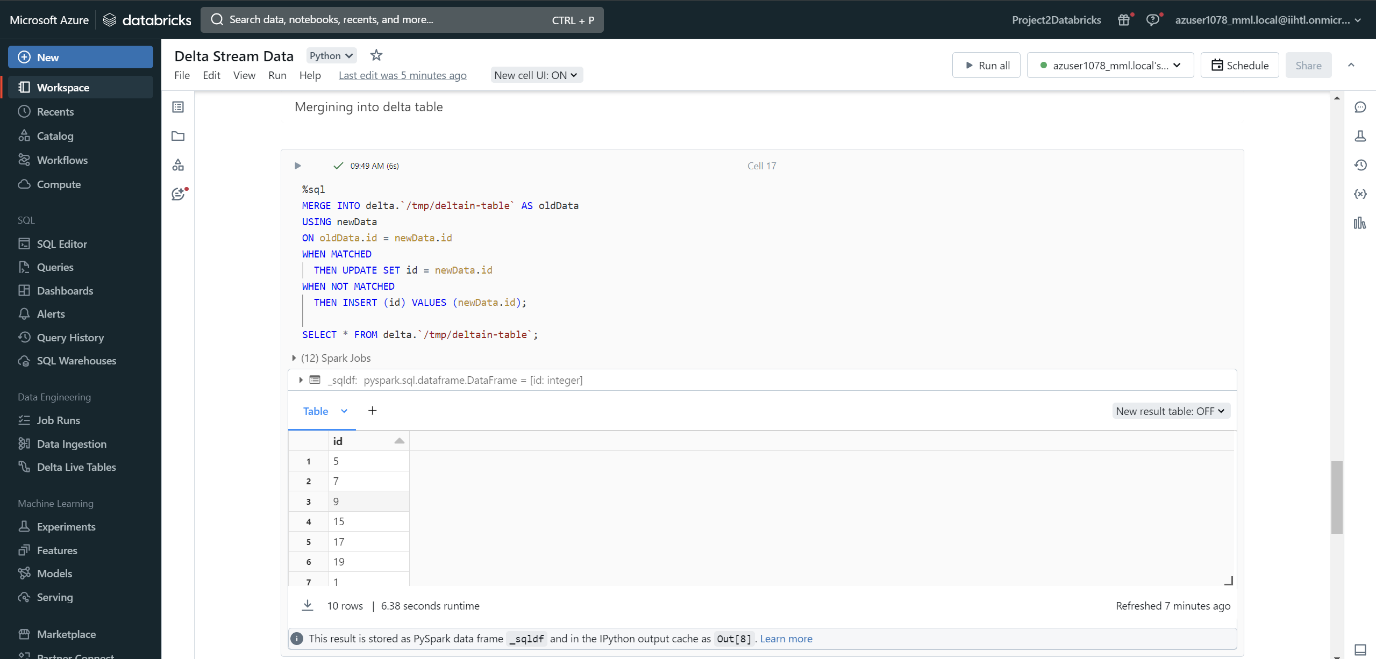


* Deleting:



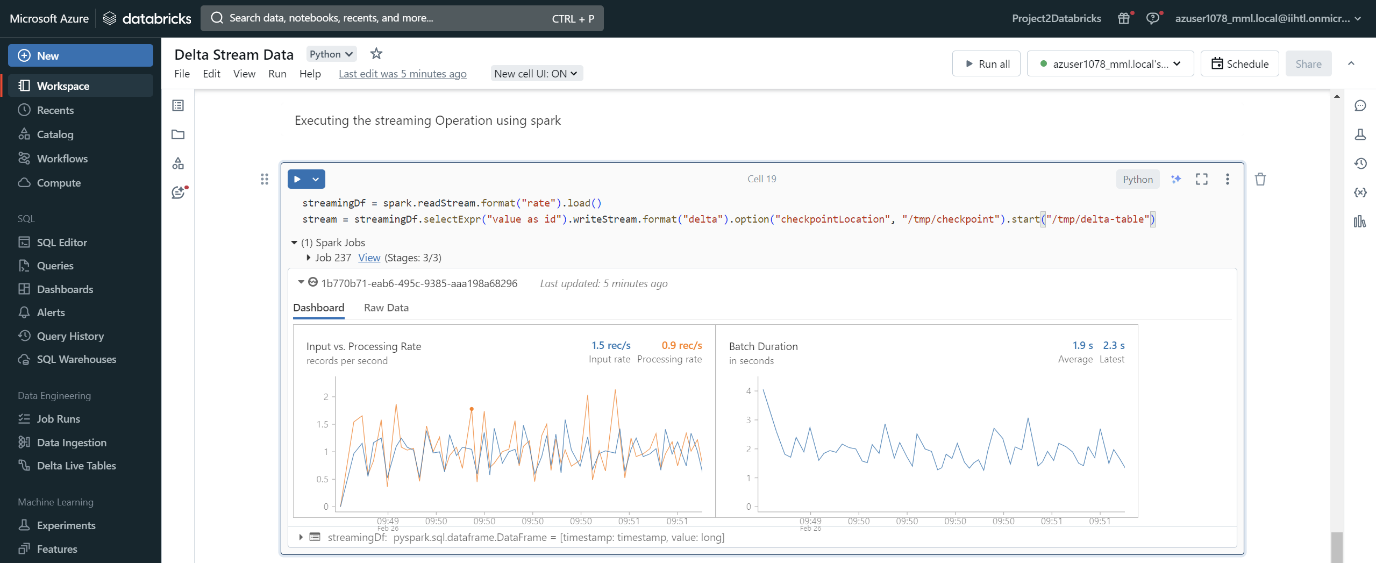
**Creating a Temporary view:**



* Merging data with the view: 

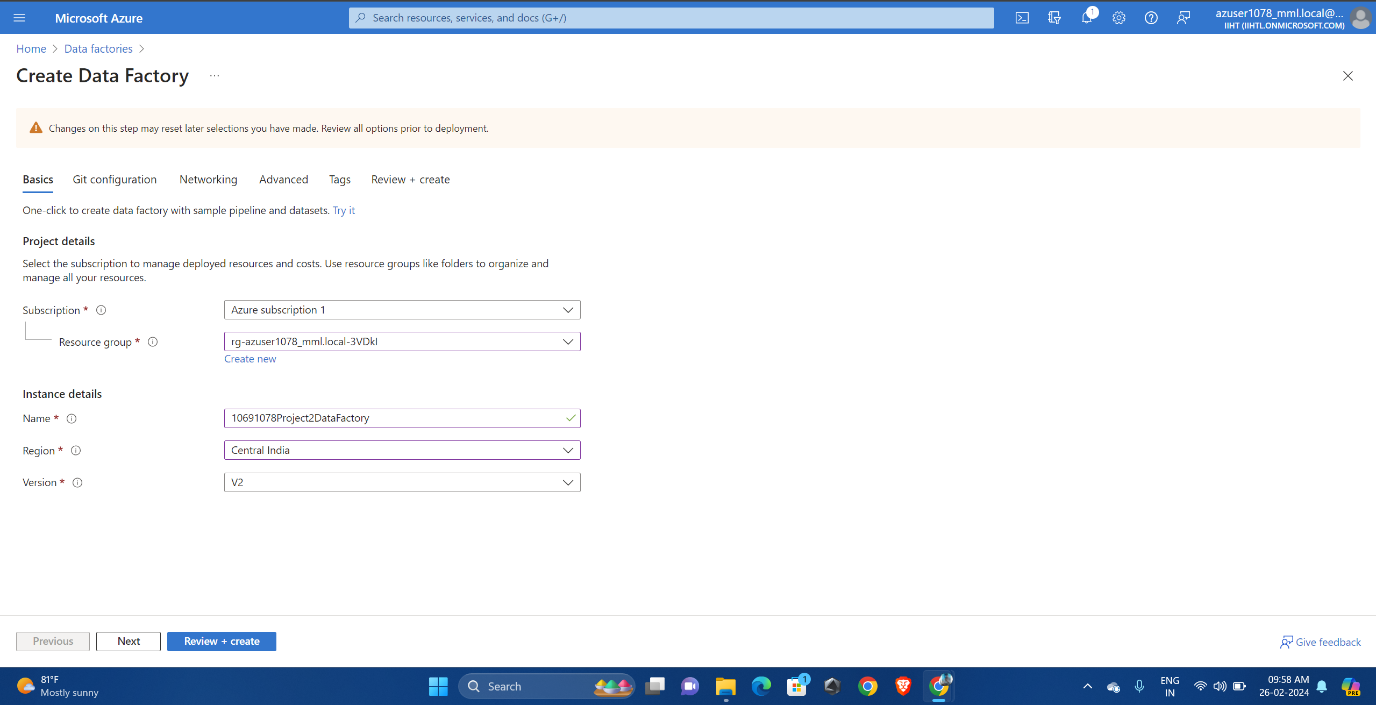
**Spark Streaming:**

* Reading the Data by using the Spark.ReadStream()

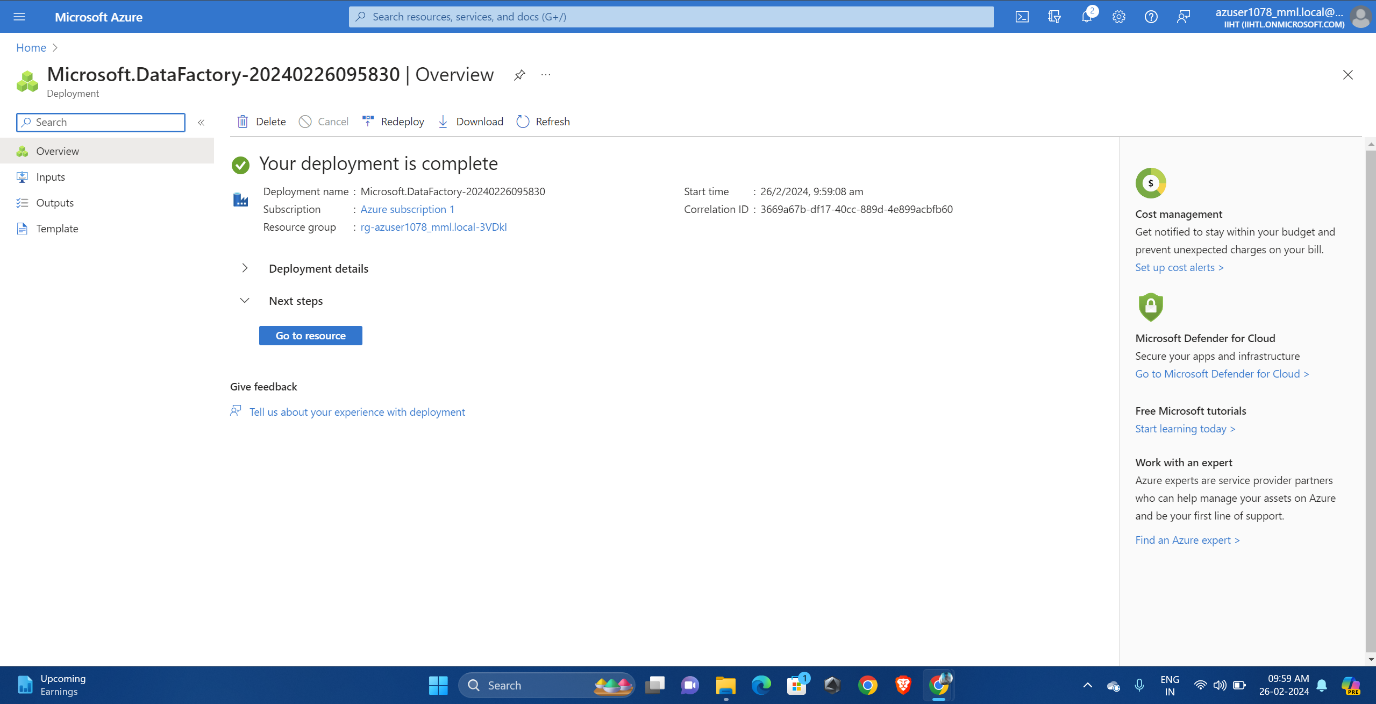


1. Data Factory Creations:

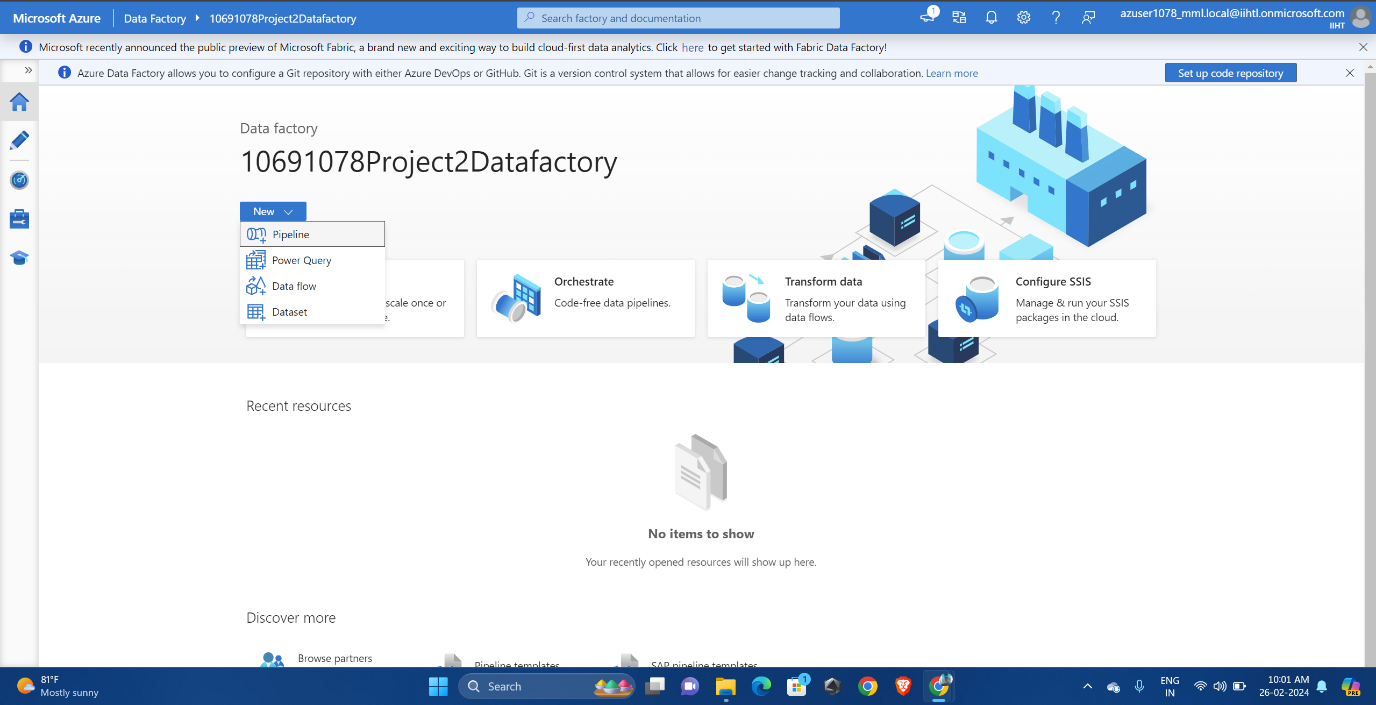
* Go to the azure home page and in the search bar type data factory
* In the data factory session click on create
* Provide the proper resource group, name and region



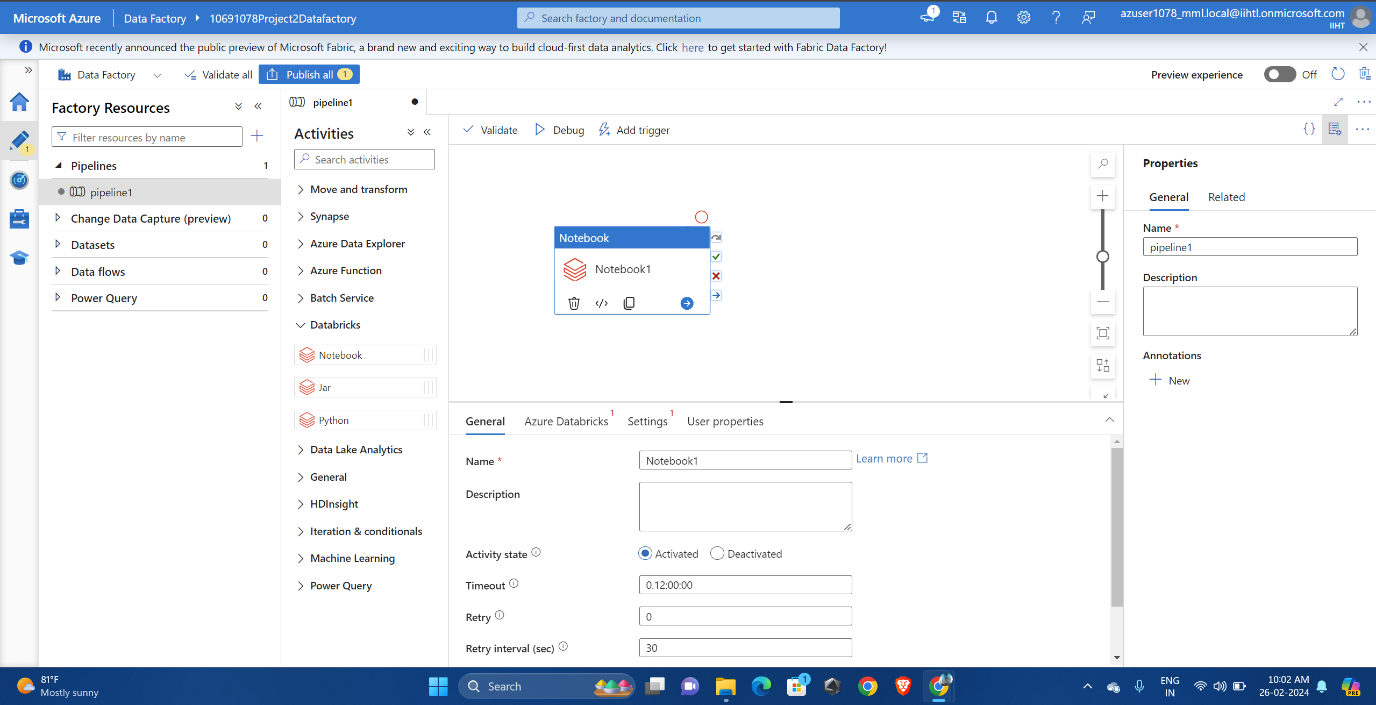
* After clicking on review+ create , then click on create.
* Deployment is complete

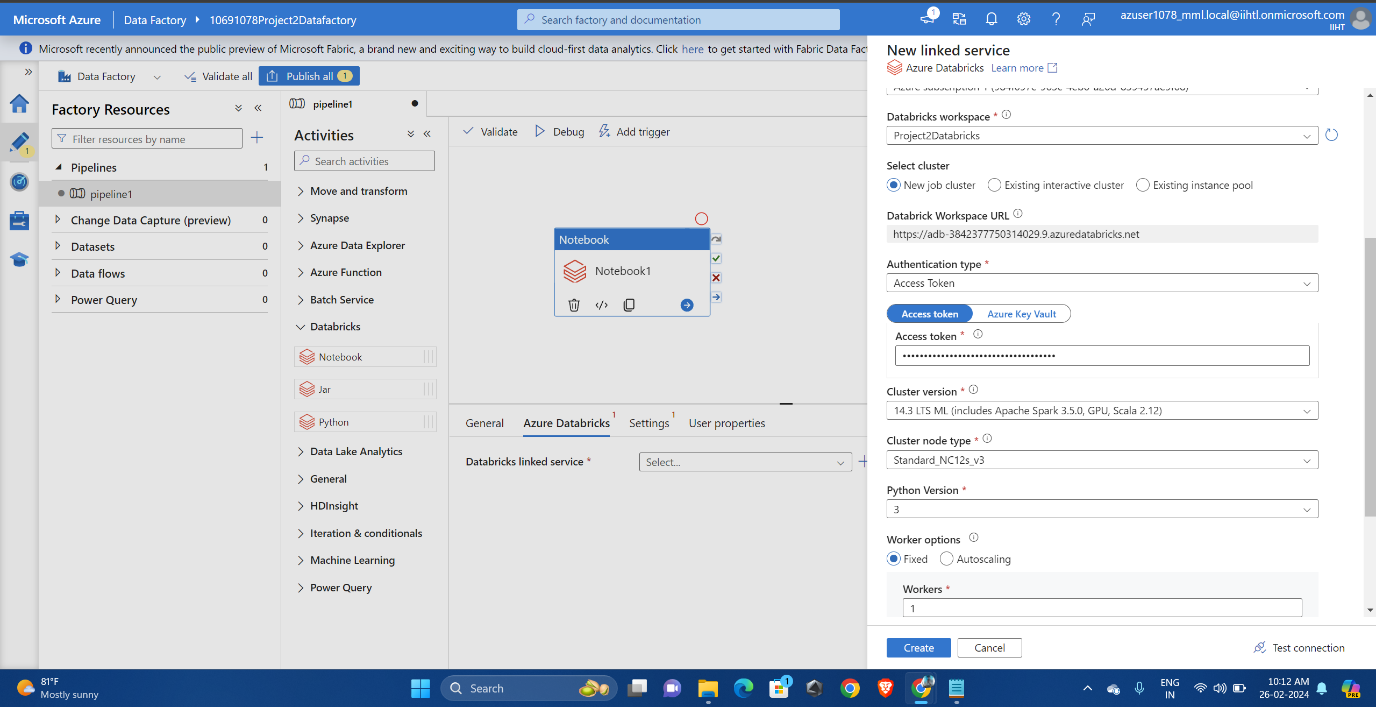


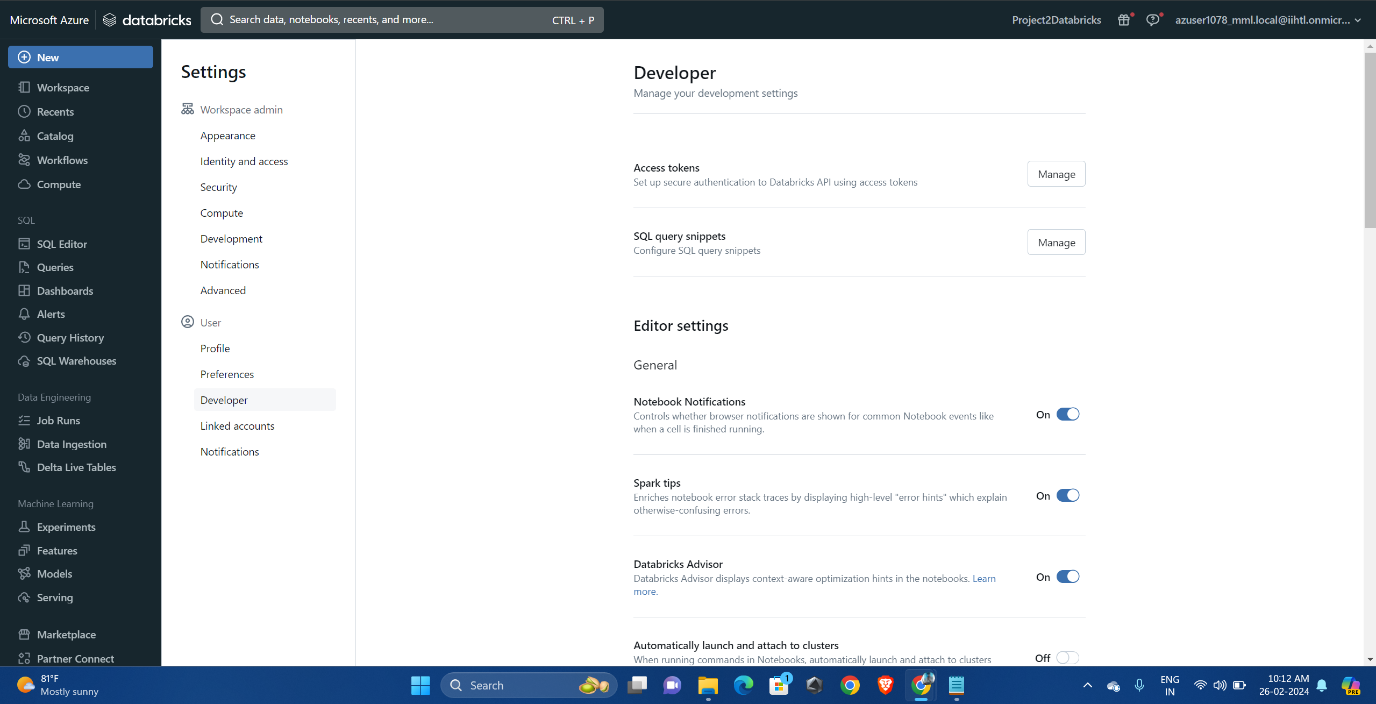
* Go to resource and open data factory work space.
* In the new session click on pipeline.

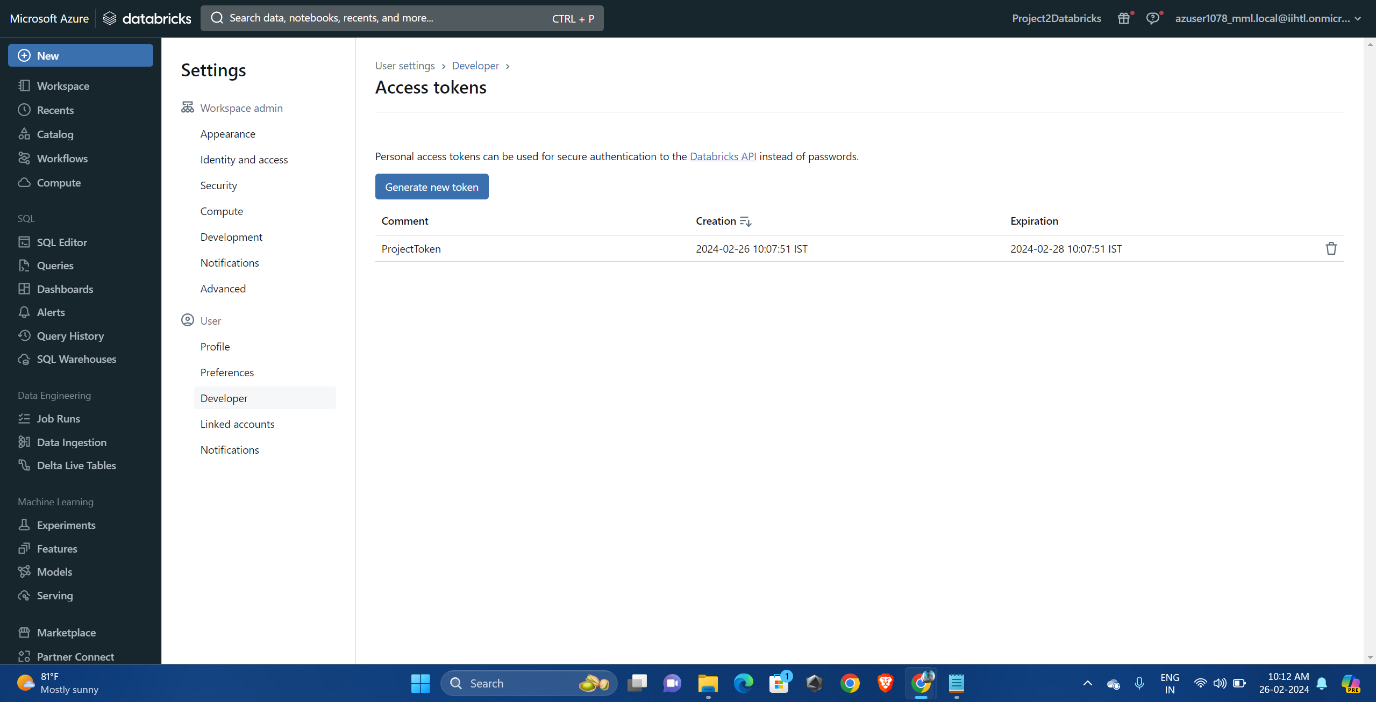


* Go to the pipeline and click on databrick and drag the notebook to workspace.

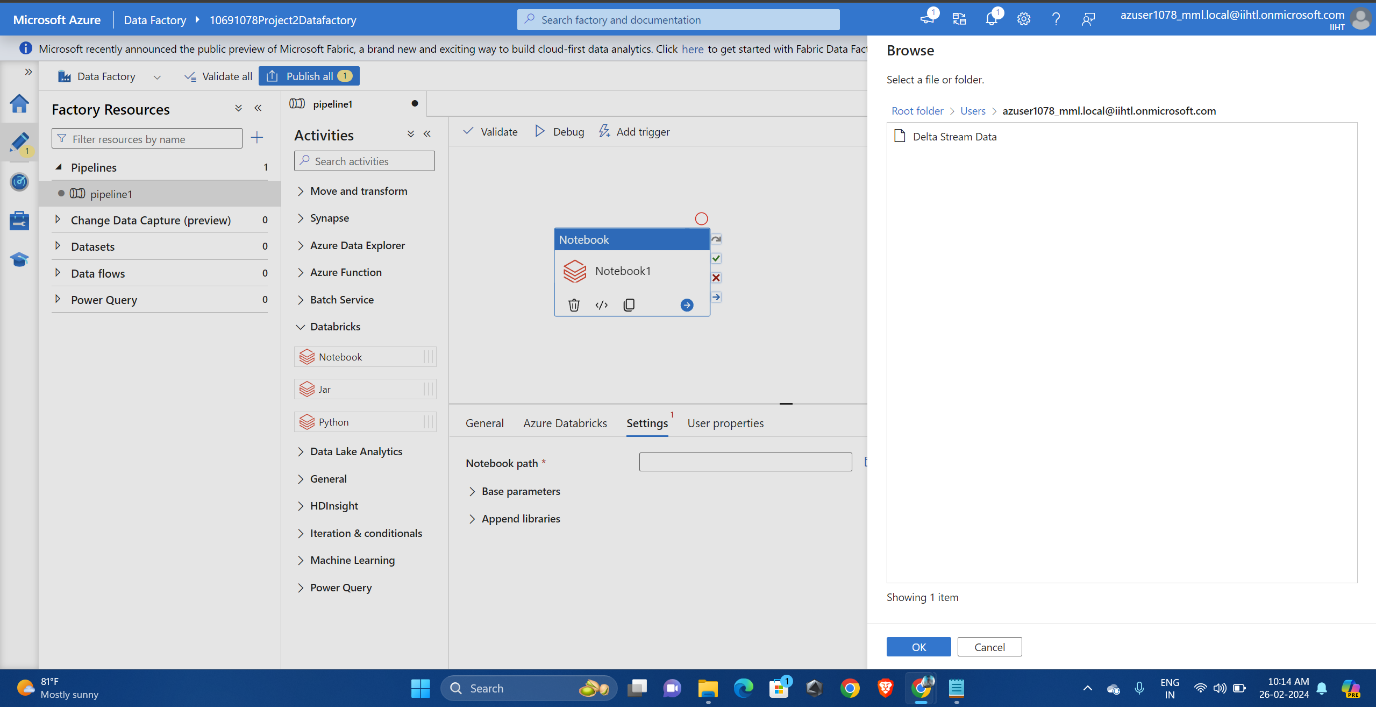


* Provide the proper access keys from the databricks. 
* For finding the access key go to data bricks . Click on user settings and select access token

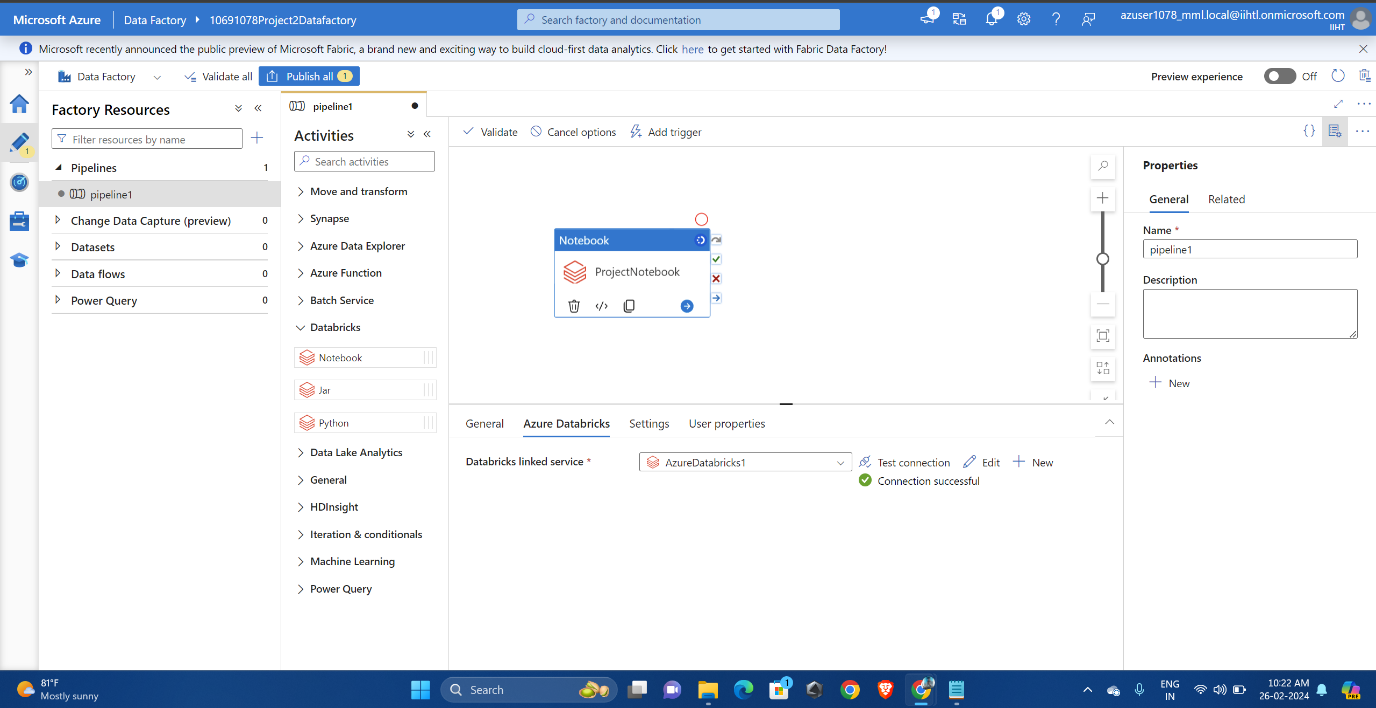




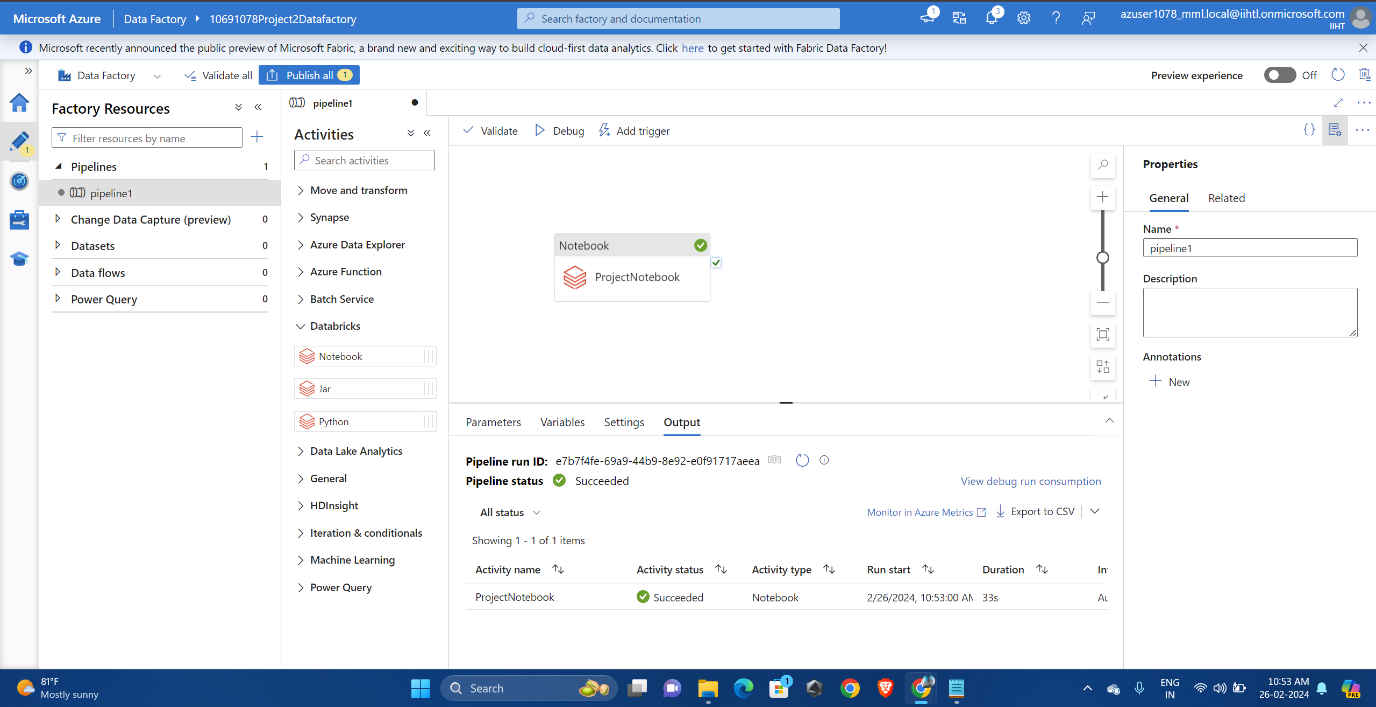
* It will generate a token , copy token value and paste it in the data factory.
* Now go to settings , click on the notebook path and provide the notebook path with we have created earlier.



* Validating the connection



* Debugging the activity



**Conclusion:**

In conclusion, the project successfully achieved stream data visualization by leveraging Azure Databricks for real-time processing and Azure Data Factory for seamless orchestration. Stream data was effectively ingested, processed, and visualized, showcasing the robust capabilities of the integrated platform. The visualization of streaming data provided valuable insights, enabling timely decision-making and analysis. Furthermore, the successful ingestion of streaming data into Azure Data Factory demonstrated the reliability and efficiency of the data pipeline, ensuring a smooth flow from ingestion to visualization. Overall, the project demonstrated the effectiveness of utilizing Azure Databricks and Azure Data Factory for stream data processing and visualization, highlighting their role in enabling scalable and flexible data solutions in modern data architectures.