**Aim** - Develop ETL (Extract-Transform-Load) pipeline to demonstrate Azure Data Warehousing capabilities

**Resources used** - Azure Resource group, Azure Storage Account, Azure Data Factory, Azure Databricks,Github.

**Procedure**-

Procedure is divided into 3 parts

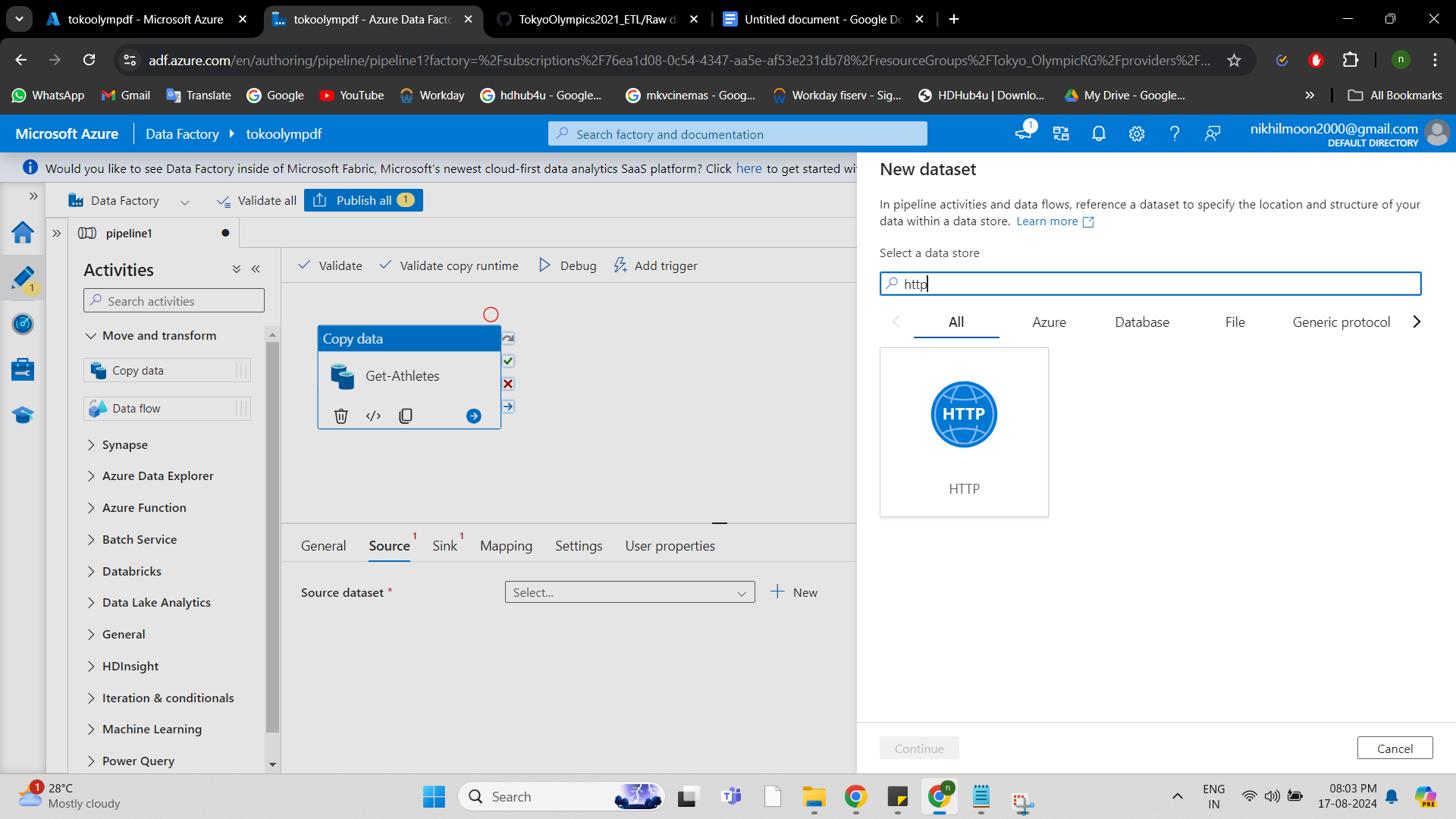
**1). Get & load data**

**2). Transform data**

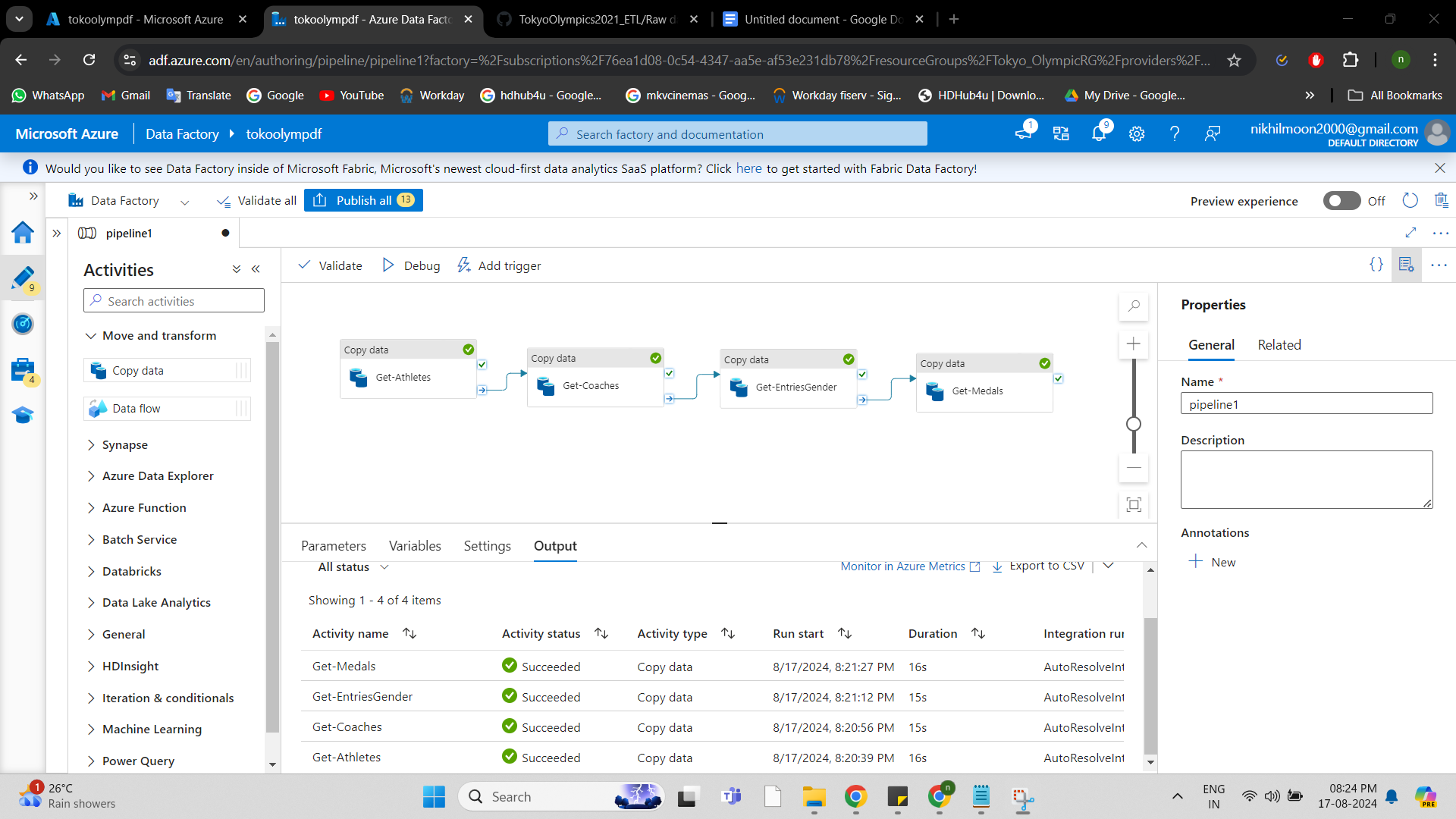
**3). Load data**

**1). Get & load data**

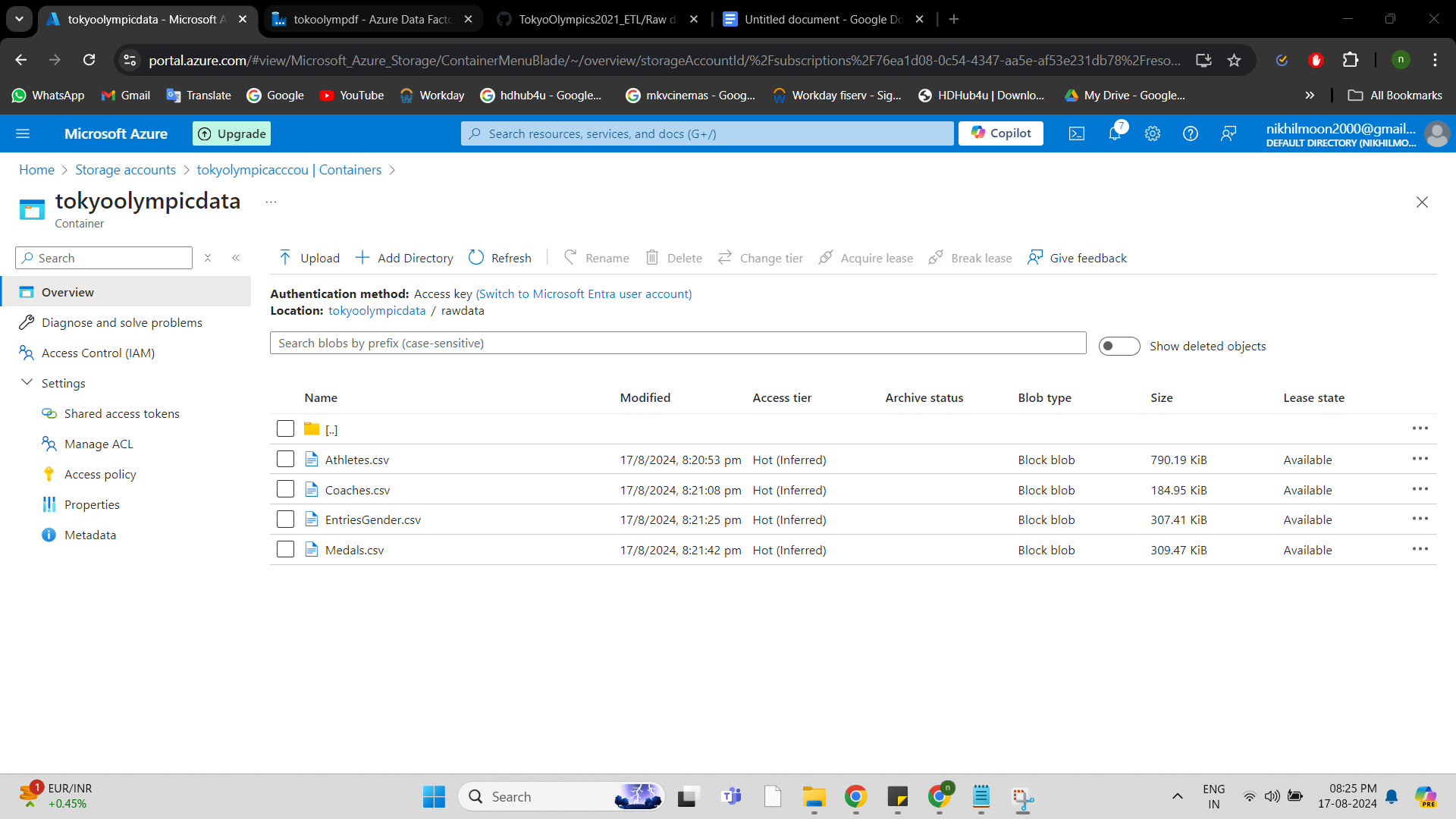
To get the data from git folder and put the data into storage container folder

1. Initialise a Git repository and organise all raw data into a single folder to serve as the project's raw data repository.
2. Create a Resource Group, preferably in the South India region, to facilitate future compute resource creation in Databricks; this will act as the central organizational unit for all Azure resources.
3. After establishing the Resource Group, proceed to set up a Storage Account, ensuring the "Enable hierarchical namespace" option is selected to allow for a structured folder system within the blob container; utilize the ARM template from the Git repository for expedited deployment.
4. Within the blob container, create two subfolders to maintain organized data storage.
5. Deploy an Azure Data Factory resource, making use of the ARM template available in the repository for a streamlined setup process.
6. Following the Azure Data Factory deployment, access the workspace, and commence the construction of a new Azure Data Pipeline.
7. Initiate the data copy process by dragging and dropping the 'Copy Data' activity into the pipeline.Assign a descriptive name to the activity to reflect its purpose.
8. For the source configuration
   1. - Establish a new connection, naming it appropriately for easy identification.
   2. - Select 'HTTP' as the data store type.
   3. - Choose 'CSV' as the file format for the data to be copied.
   4. - Set the authentication type to 'Anonymous' and provide the URL for the files located in the
   5. Finalize the source configuration by clicking 'Create'.
9. For the sink configuration:
   1. - Finalize the source configuration by clicking 'Create'.
   2. - For the sink configuration:
   3. - Create a new connection within the designated resource group and storage account.
   4. - Provide a specific file name, including the necessary file extension.
   5. - Complete the sink setup by clicking 'Create'.

Repeat the above steps to transfer all necessary data files from the Git repository to the specified container folder in the storage account. At the end, execute the pipeline



Data should get stored in container folder (Raw Data)

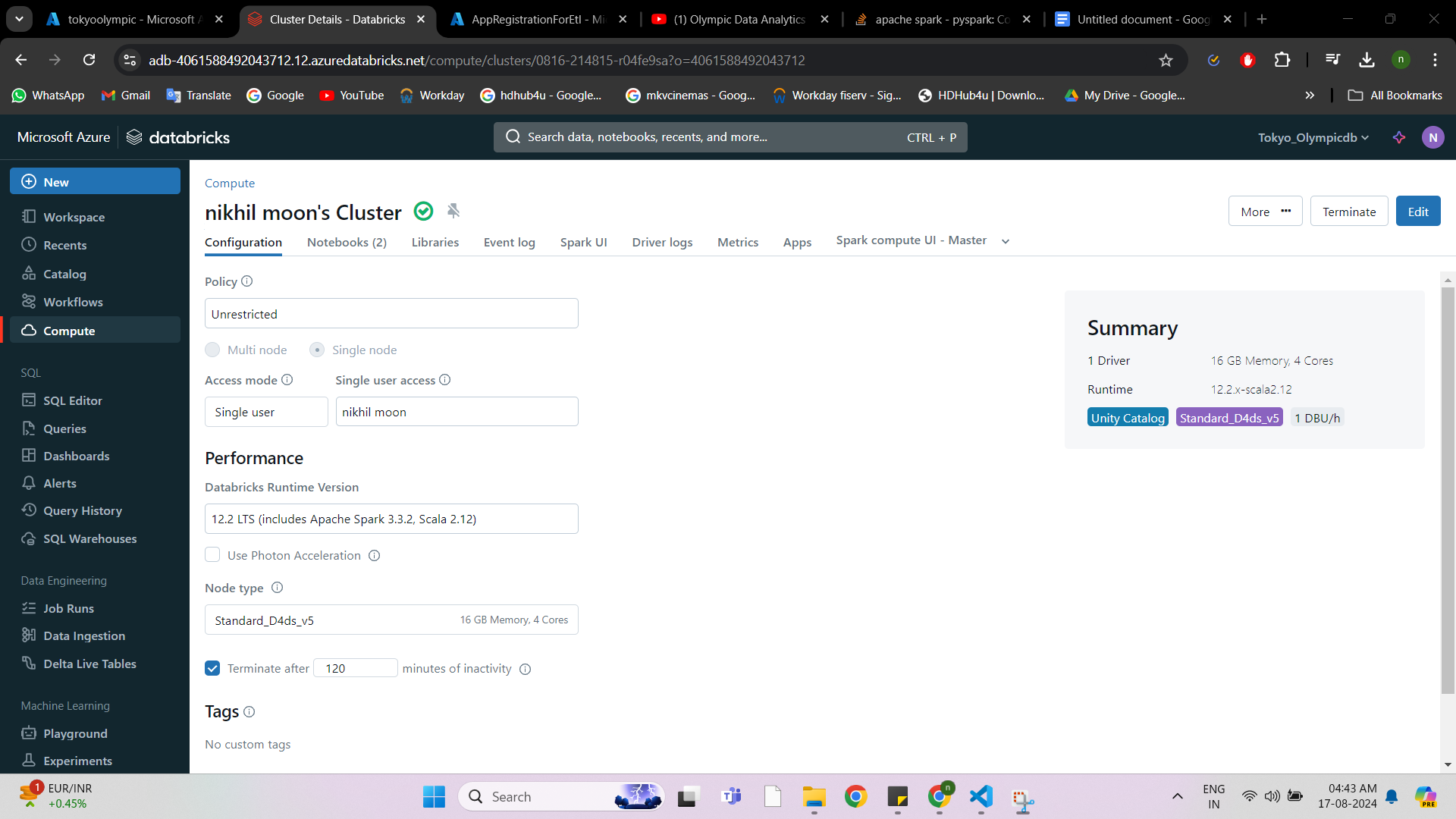


**2). Transform data**

1. Create an Azure Databricks resource through the Azure portal, which may take some time.

2. Launch the Databricks workspace once the resource is provisioned.

3. Create a compute cluster within the Databricks workspace, ensuring it meets the necessary specifications to run Apache Spark applications.



4. Create a new application registration to establish a connection between Azure Storage and Databricks.

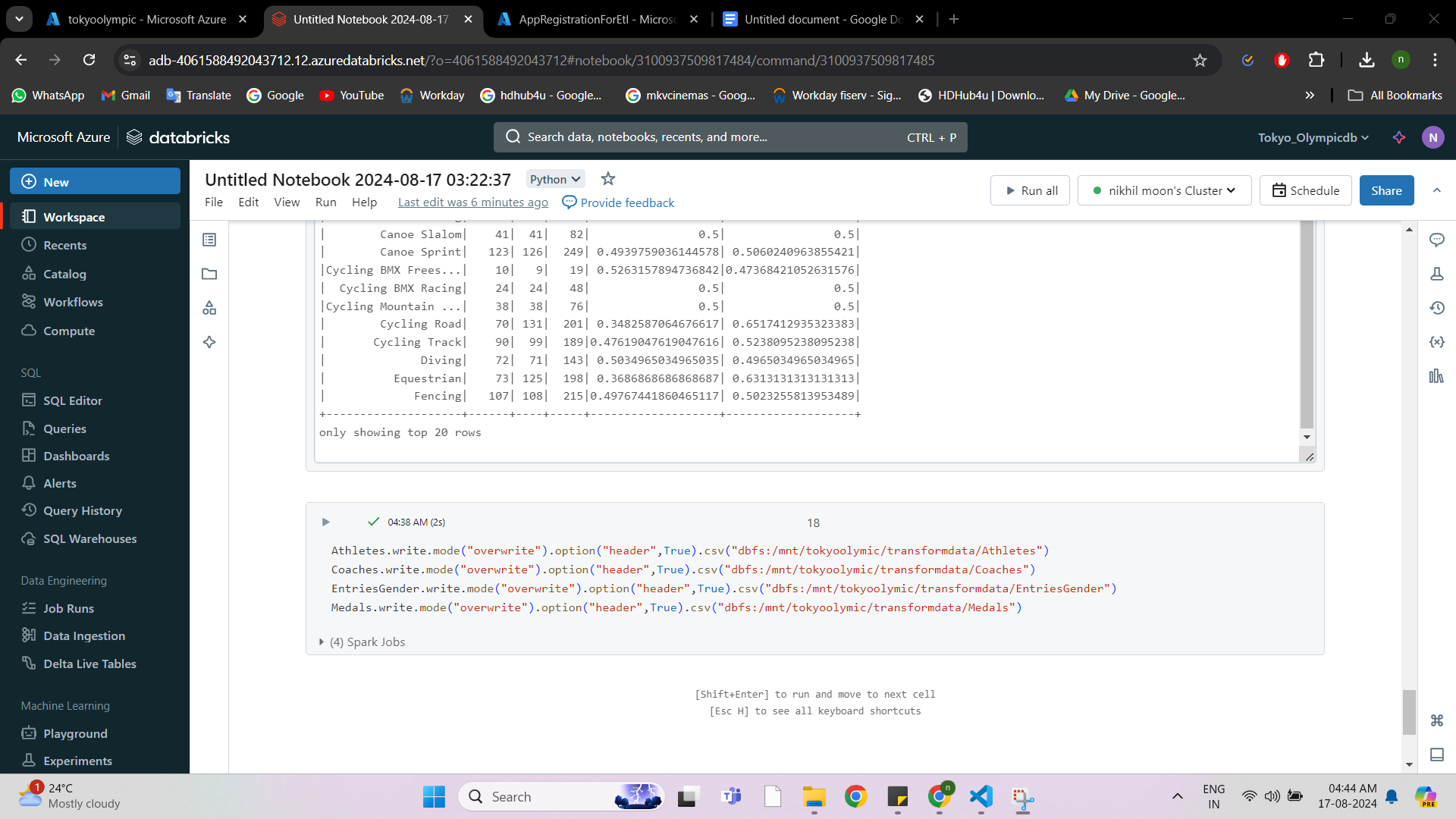
5. Assign the 'Storage Contributor' role to the registered application, allowing it to read and write data in the Azure Storage account.

6. In the Databricks workspace, create a new notebook and establish a connection to mount the Azure Storage by providing the client ID, tenant ID, client secret, container name, and storage account name.

7. Verify the mounted storage by accessing files within the storage account from the Databricks workspace.

8. Perform data transformations as required, such as schema modifications or aggregations like calculating the average number of medals won by female and male athletes for demonstration purposes.

You can refer python files on repo for the codes.



**3). Load data**

Once you have transfrom the data with databricks w’ll load the transformed data into container folder. Utilize Python code to manage and orchestrate the data workflow.

Perform cascading tasks using the transformed data as input.

End results will be in the container folder

