Capstone Project II

Abdullah Altulahi

Ahad Mufareh

Ameerah Althobiti

SDA Data Engineer Bootcamp

GitHub link:

[Stackoverflow](https://github.com/abdullahfawazaltulahi/Stackoverflow_project)

# **Introduction of Capstone Project Documentation**

Our project aims to process and store data from various sources related to the StackOverflow platform. The data includes posts, post types, and user information. The goal is to create a data pipeline that extracts the data from different sources, performs necessary transformations, and stores it in the appropriate directories in Azure Storage Blob.

In this paper will document all work to be easy to understand the work and clarify each step of the Project. And the process of the project will be :

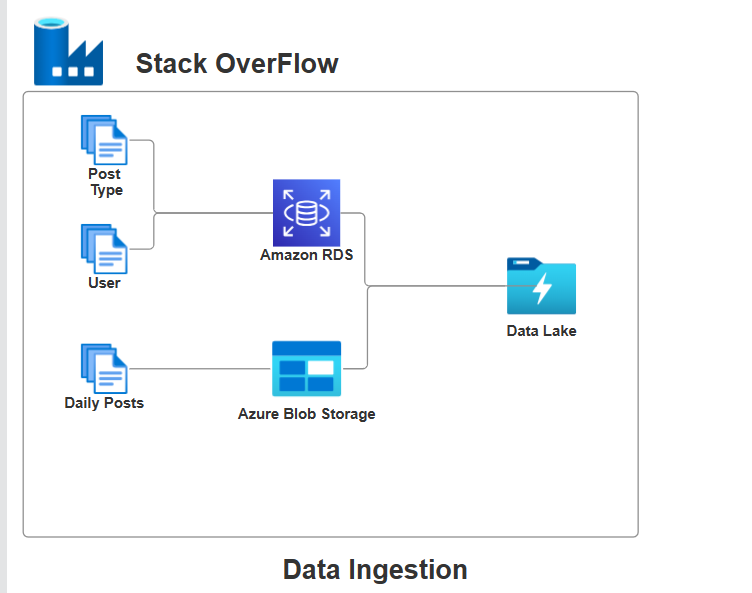
* Data ingestion
* Data Transformation
* Data visualization

## **Part 1: Data ingestion**

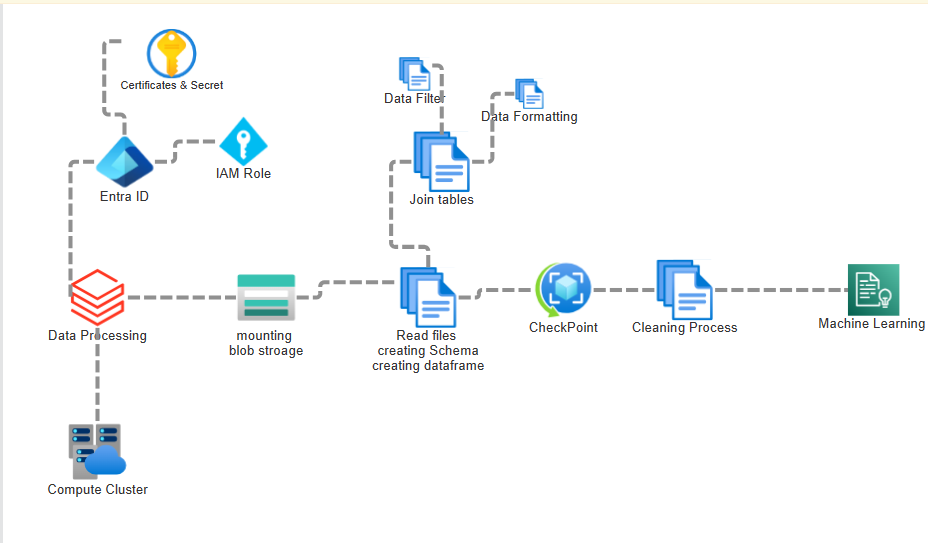
We will use Azure data Factory to build our pipelines that load the Data into Data lake. To complete the first Part of the Project we need to build:

* Links Services
* Dataset
* Containers

*[below diagram explain the source and the data came from]*



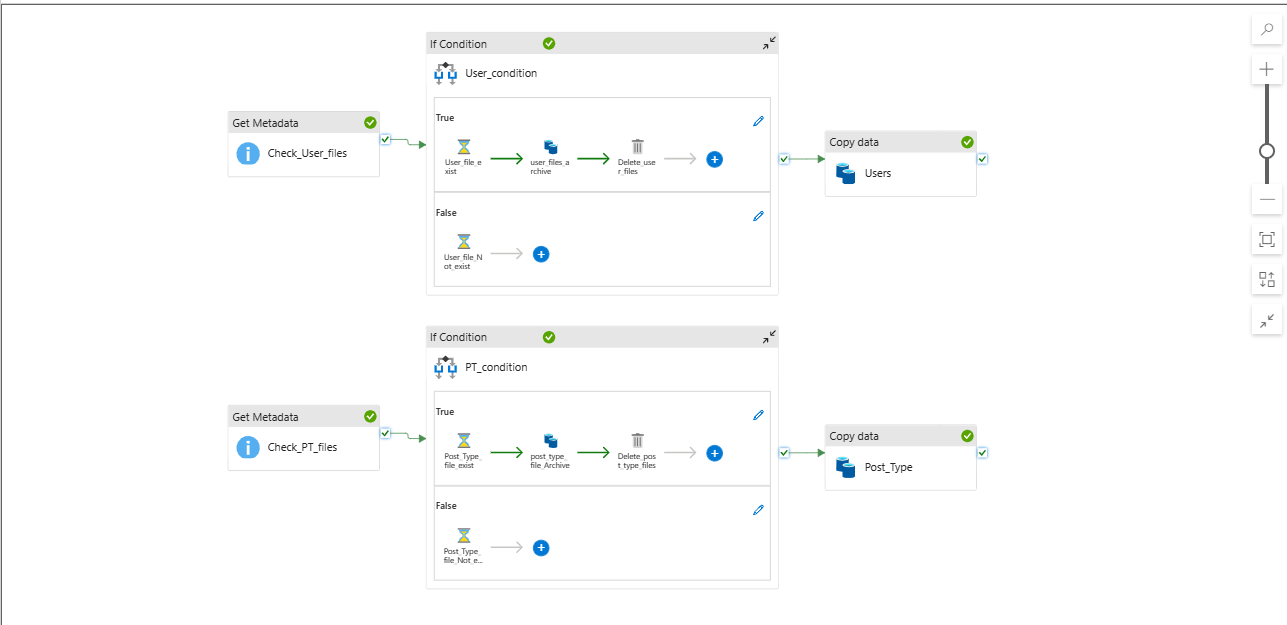
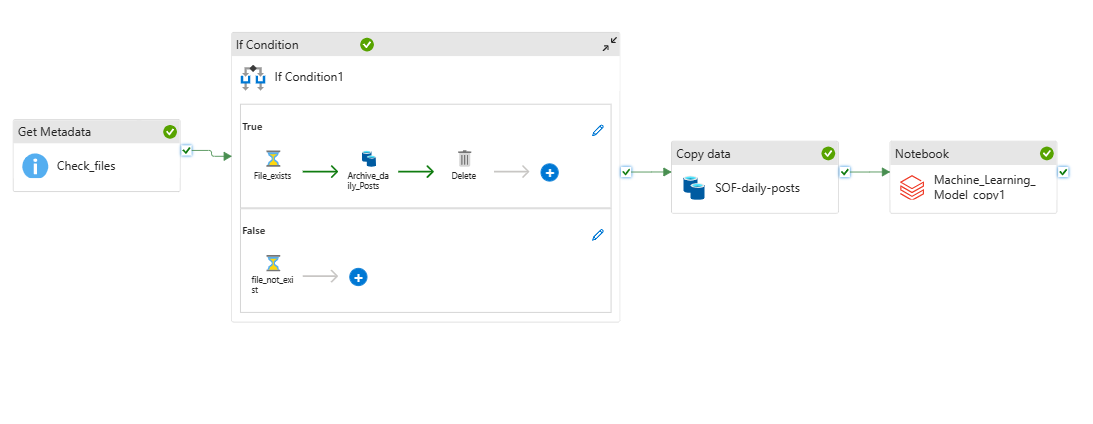
## **Data transformation**



**In this part of Data transformation in Azure databricks we work in the first :**

* Creating application register and certificates & secret and give this apps a role for Blob storage that authenticate and use mountpoint to Blob Storage. When we give role contributor for blob storage we can mount to it And read the files that we need it and applies certain process that contain reading the files and join necessary tables together and format it and save this point for any fault tolerance may happen. Then we clean the text which we was use ‘body = text’ which contain - Remove URLs such as http://stackoverflow.com - Remove special characters - Substituting multiple spaces with single space - Lowercase all text - Trim the leading/trailing whitespaces. Then we apply the Machine learning which Logistic regression and before training we have to tokenize and apply stopword remover then we use count vectorization . and TF-IDF vectorization then we encode the label which is tags then we train the model and evaluated

## **Project scenario :**



1. Every day at 2AM, the StackOverflowETL\_daily pipeline is triggered. It starts by checking if there are any new posts to process. The Get Metadata activity, "Check\_files," checks if there are any files in the "Posts\_destination\_af" dataset.

2. If there are new posts, the pipeline proceeds to process them. The Copy Data activity, "SOF-daily-posts," reads the Post\_source dataset, which is a collection of parquet files located in the public Azure Storage Blob. It copies the data to the "post\_destination" dataset in your Azure Storage Blob container.

3. After copying the data, the pipeline checks if the posts have already been archived. The Get Metadata activity, "Check\_files," checks if the posts exist in the "post\_destination\_af" dataset.

4. If the posts exist, the pipeline archives them. The Copy Data activity, "Archive\_daily\_Posts," reads the "post\_destination\_af" dataset and uses a dynamic "\*.parquet" file path to select all parquet files. It copies these files to the "AFP" dataset, which represents the archive directory for parquet files. The original posts are then deleted using the Delete activity.

5. If the posts do not exist, the pipeline waits for 3 seconds before continuing.

6. Additionally, every week at 2AM, the StackOverflowETL\_weekly pipeline is triggered. It starts by checking if there are any new user files to process. The Get Metadata activity, "Check\_User\_files," checks if there are any files in the "User\_destination" dataset.

7. If there are new user files, the pipeline proceeds to archive them. The Copy Data activity, "user\_files\_archive," reads the "User\_destination" dataset and copies the files to the "ArchvieFiles\_des" dataset, which represents the archive directory for CSV files. After archiving, the original user files are deleted using the Delete activity.

8. If there are no new user files, the pipeline waits for 3 seconds before continuing.

9. The pipeline then proceeds to process the user information. The Copy Data activity, "Users," reads the "Users\_source" dataset, which is a PostgreSQL database. It manually enters the table information and copies the data to the "users\_destination" dataset, representing the landing directory for user information.

10. Next, the pipeline checks if there are any new post type files to process. The Get Metadata activity, "Check\_PT\_files," checks if there are any files in the "PostType\_destination" dataset.

11. If there are new post type files, the pipeline proceeds to archive them. The Copy Data activity, "post\_type\_file\_Archive," reads the "PostType\_destination" dataset and copies the files to the "ArchvieFiles\_des" dataset. After archiving, the original post type files are deleted using the Delete activity.

12. If there are no new post type files, the pipeline waits for 3 seconds before continuing.

13. Finally, the pipeline processes the post types. The Copy Data activity, "Post\_Type," reads the "PostType\_source" dataset, which is a PostgreSQL database. It manually enters the table information and copies the data to the "PostType\_destination" dataset, representing the

## **Project summary**

1. Azure Storage Blob Setup:

- Storage Account: StackOverflowProjectRG

- Container: stackoverflow

- Directories:

- archive

- BI

- Landing

- Logs

- model

- stringindexer

- synapse

- Subdirectories in Landing:

- stackOverFlow-Posts

- stackOverFlow-PostType

- stackOverFlow-Users

2. Data Factory Setup:

- Data Factory Name: SOFDataPipelineFactory

- Linked Services:

- wcd\_public\_blob:

- Linked to a public Azure Storage Blob to access the Post\_source dataset.

- RDS\_db:

- Linked to a PostgreSQL database for accessing the Users\_source and PostType\_source datasets.

- my\_blob:

- Linked to the stackoverflow Azure Storage Blob container for various datasets.

3. Datasets:

- Post\_source:

- Linked Service: wcd\_public\_blob

- File path: Root folder/de-project-st/posts\_today/<parquet files>

- Users\_source:

- Linked Service: RDS\_db

- Path: raw\_st/users

- PostType\_source:

- Linked Service: RDS\_db

- Path: raw\_st/posttypes

- posttype\_destination:

- Linked Service: my\_blob

- Path: stackoverflow/Landing/stackOverFlow-PostType/

- users\_destination:

- Linked Service: my\_blob

- Path: stackoverflow/Landing/stackOverFlow-Users/

- ArchvieFiles\_des:

- Linked Service: my\_blob

- Path: stackoverflow/Archive/csv/

- Posts\_destination:

- Linked Service: my\_blob

- Path: stackoverflow/Landing/stackOverFlow-Posts/

- Posts\_destination\_af:

- Linked Service: my\_blob

- Path: stackoverflow/Landing/stackOverFlow-Posts/

- ArchvieFiles\_des\_parquet:

- Linked Service: my\_blob

- Path: stackoverflow/Archive/parquet/

- AFP:

- Linked Service: my\_blob

- Path: stackoverflow/Archive/parquet/

4. Pipelines:

- StackOverflowETL\_daily:

- Activities:

- Get Metadata: Check\_files (Linked Service: my\_blob, Dataset: Posts\_destination\_af, Field list: Exists)

- If Condition: If Check\_files.Exists is true

- True Branch:

- Wait Activity: File\_exists (1 second wait time)

- Copy Data Activity: Archive\_daily\_Posts (Source: post\_destination\_af with dynamic "\*.parquet" file path, Sink: AFP)

- Delete Activity: Delete (Source: post\_destination with enabled logging)

- False Branch:

- Wait Activity: file\_not\_exists (3 seconds wait time)

- Trigger: Daily-trigger (scheduled every day at 2AM)

- StackOverflowETL\_weekly:

- Activities:

- Get Metadata: Check\_User\_files (Linked Service: my\_blob, Dataset: User\_destination, Field list: Exists)

- If Condition: If Check\_User\_files.Exists is true

- True Branch:

- Wait Activity: User\_file\_exist (1 second wait time)

- Copy Data Activity: user\_files\_archive (Source: User\_destination, Sink: ArchvieFiles\_des)

- Delete Activity: Delete\_user\_files (Source: User\_destination with enabled logging)

- False Branch:

- Wait Activity: User\_file\_Not\_exist (3 seconds wait time)

- Copy Data Activity: Users (Source: Users\_source entered manually as table, Sink: users\_destination)

- Get Metadata: Check\_PT\_files (Linked Service: my\_blob, Dataset: PostType\_destination, Field list: Exists)

- If Condition: If Check\_PT\_files.Exists is true

- True Branch:

- Wait Activity: Post\_Type\_file\_exist (1 second wait time)

- Copy Data Activity: post\_type\_file\_Archive (Source: PostType\_destination, Sink: ArchvieFiles\_des)

- Delete Activity: Delete\_post\_type\_files (Source: PostType\_destination with enabled logging)

- False Branch:

- Wait Activity: Post\_Type\_file\_Not\_exist (3 seconds wait time)

- Copy Data Activity: Post\_Type (Source: PostType\_source entered manually as table, Sink: PostType\_destination)

- Trigger: weekly-trigger (scheduled once every week at 2AM)