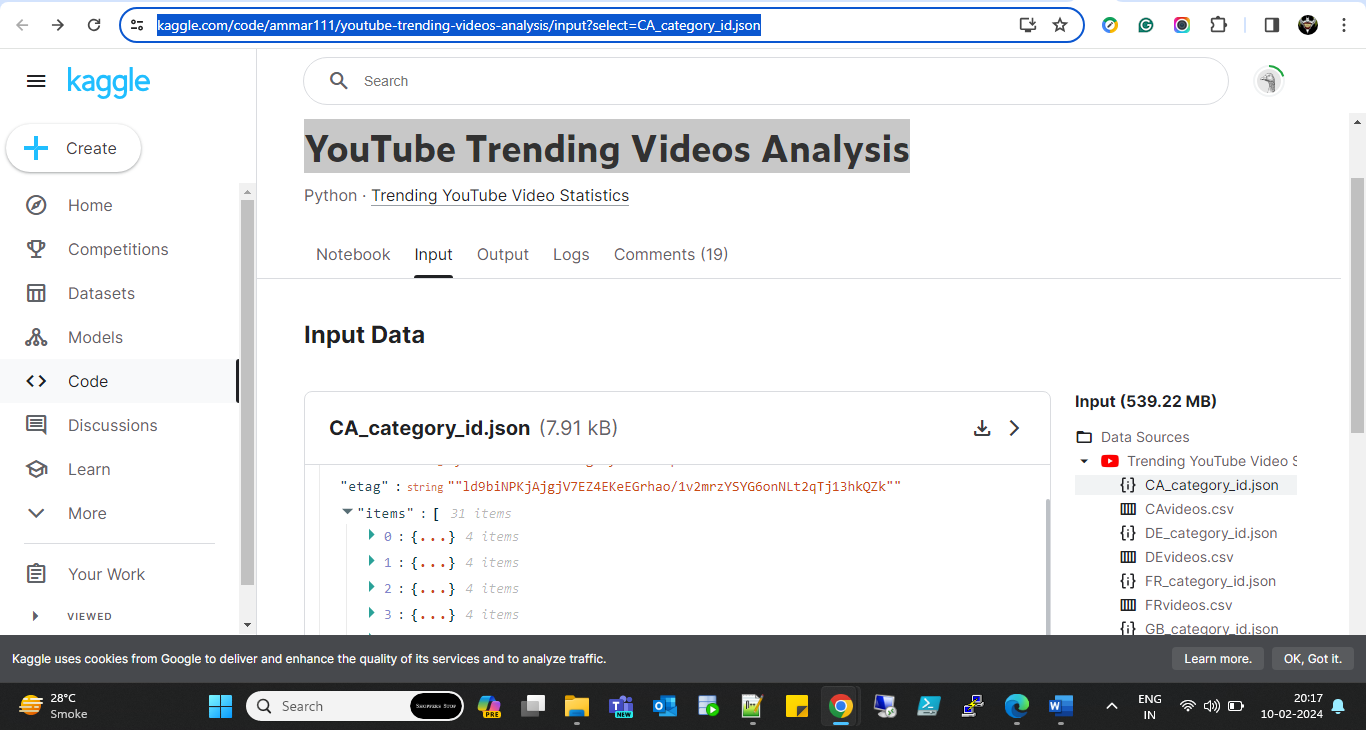
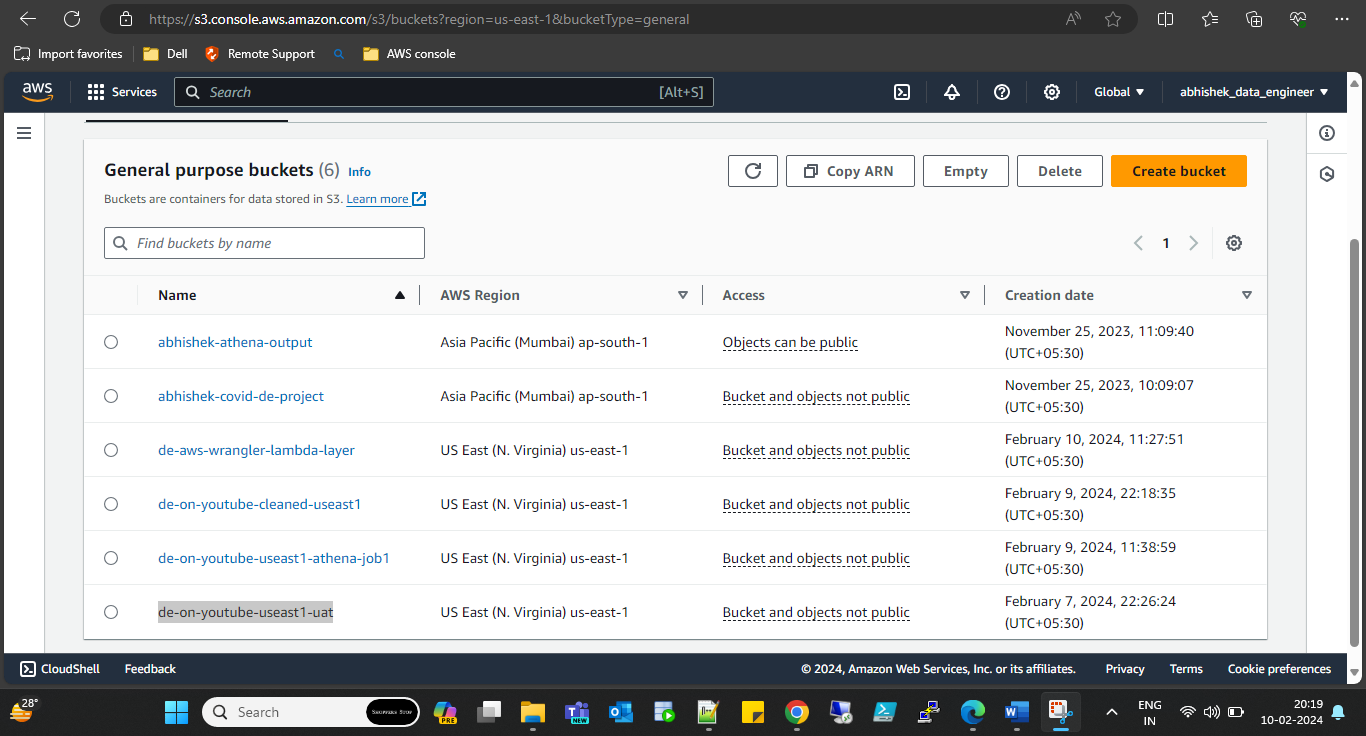
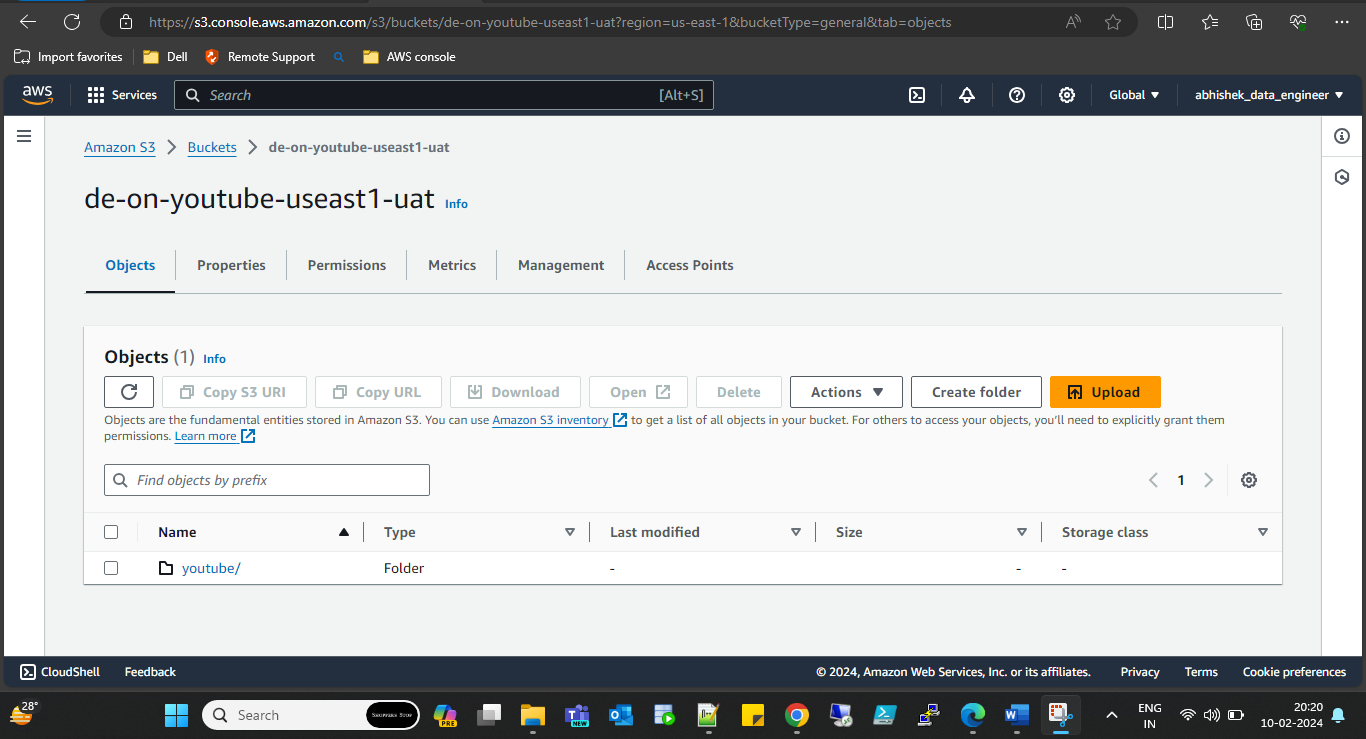
**Trading YouTube Videos Analysis**

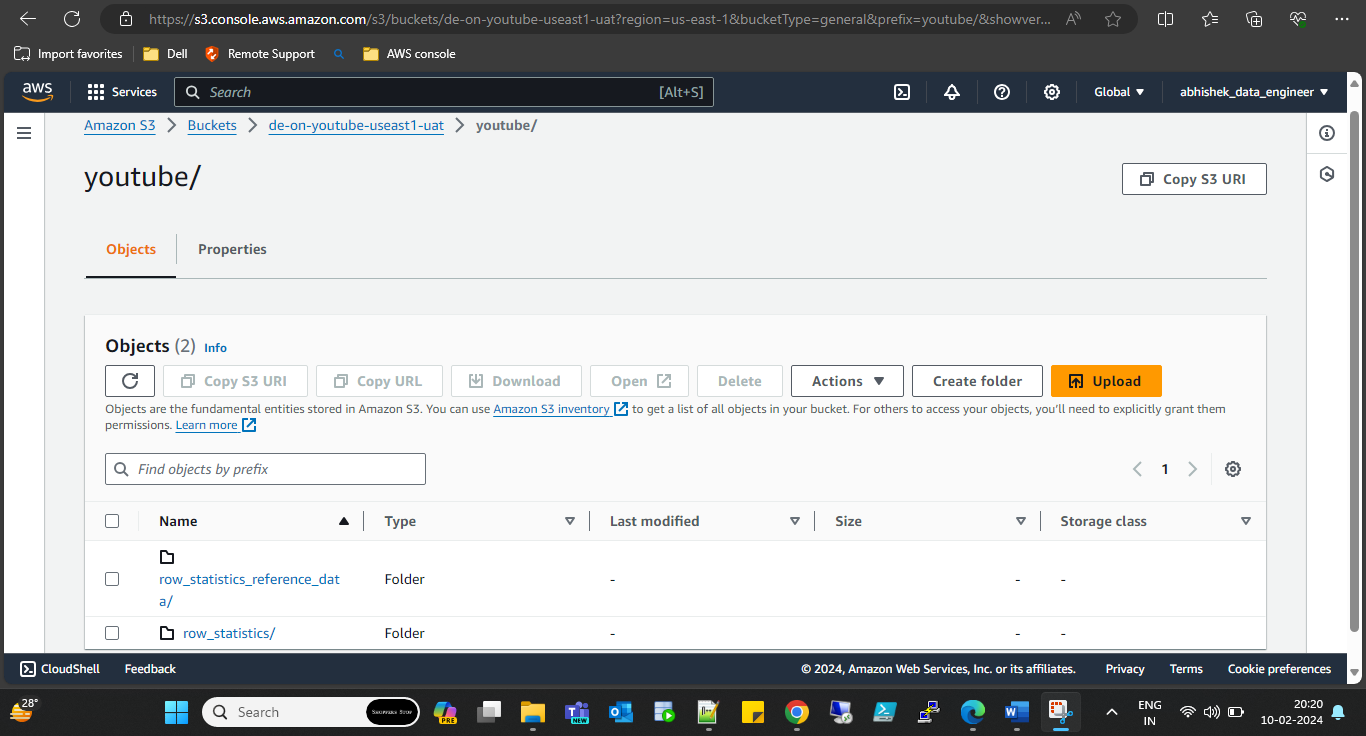
Step1: Download source data from Kaggle url (<https://www.kaggle.com/code/ammar111/youtube-trending-videos-analysis/input?select=CA_category_id.json> ).

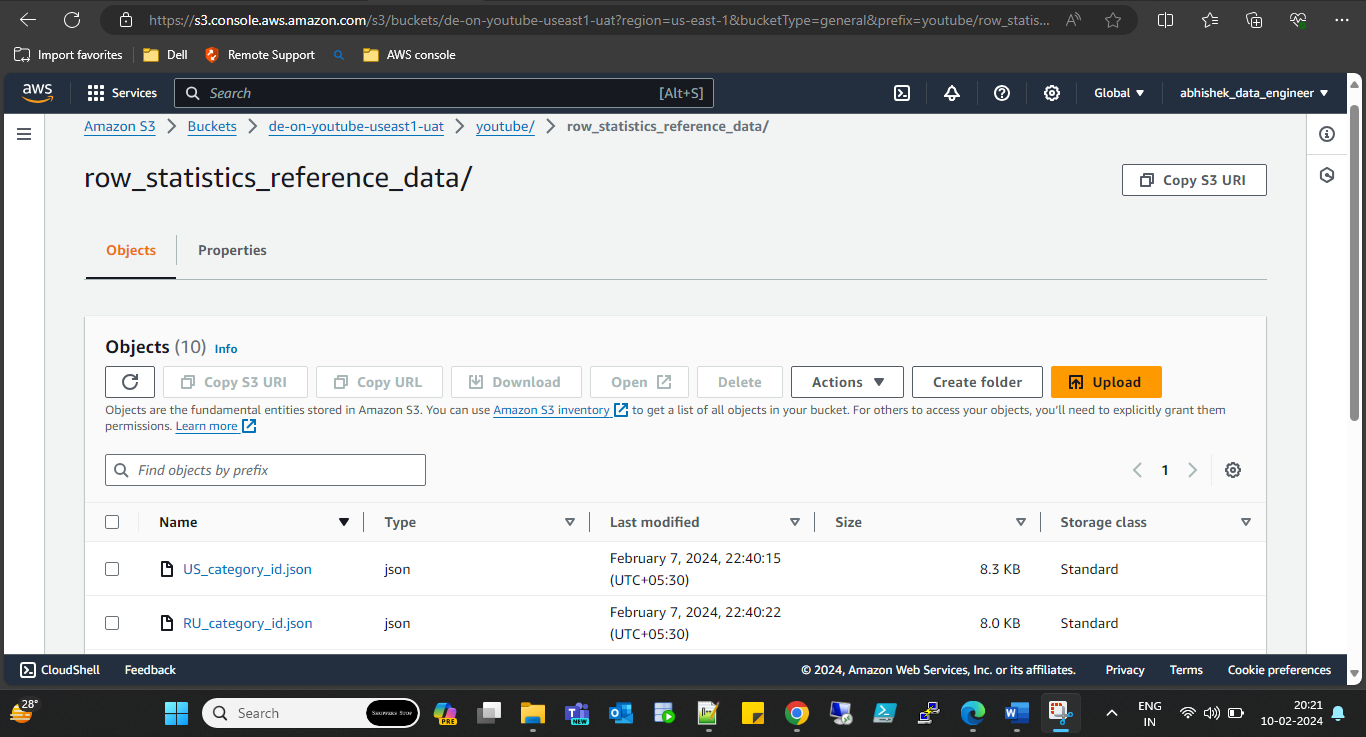


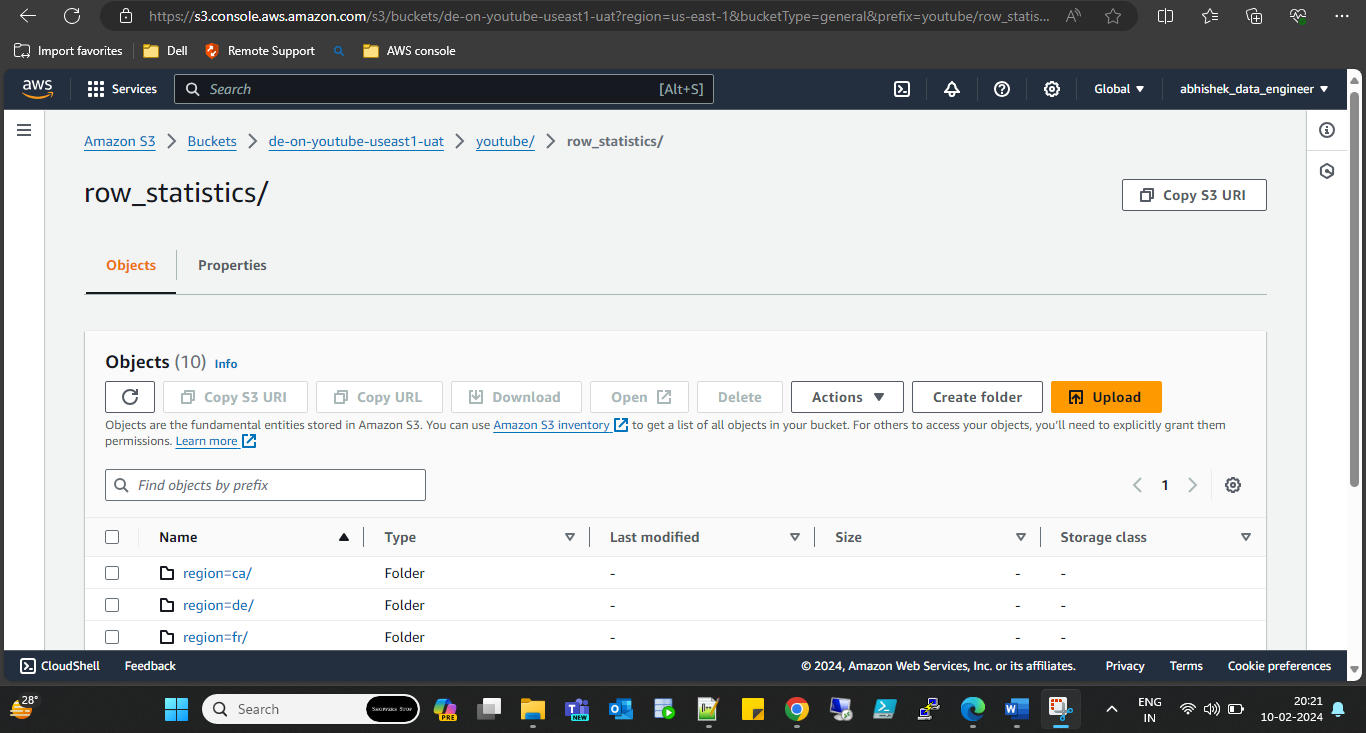
Step2: Store downloaded data from Kaggle to S3 bucket “[de-on-youtube-useast1-uat](https://s3.console.aws.amazon.com/s3/buckets/de-on-youtube-useast1-uat?region=us-east-1&bucketType=general)” and sub buckets “[row\_statistics\_reference\_data/](https://s3.console.aws.amazon.com/s3/buckets/de-on-youtube-useast1-uat?region=us-east-1&bucketType=general&prefix=youtube/row_statistics_reference_data/&showversions=false)” & “[row\_statistics/](https://s3.console.aws.amazon.com/s3/buckets/de-on-youtube-useast1-uat?region=us-east-1&bucketType=general&prefix=youtube/row_statistics/&showversions=false)”



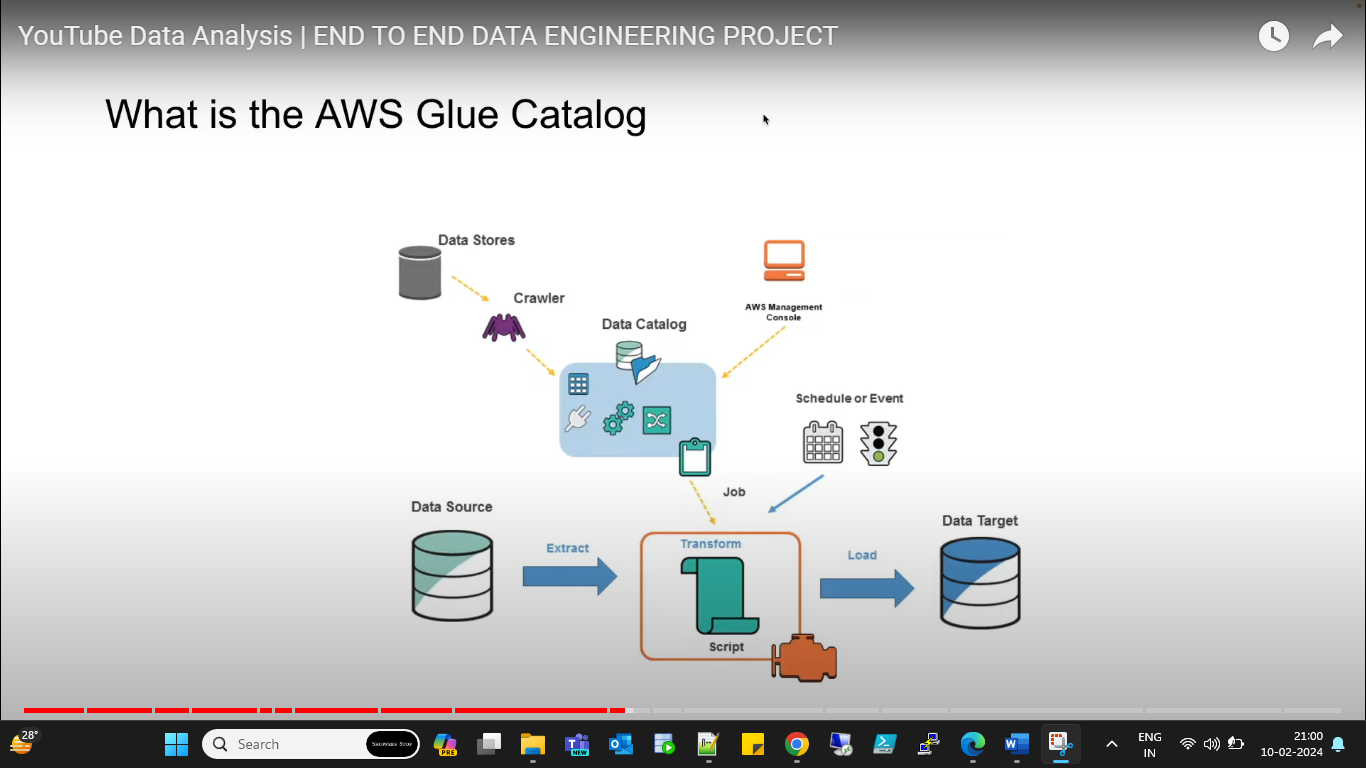






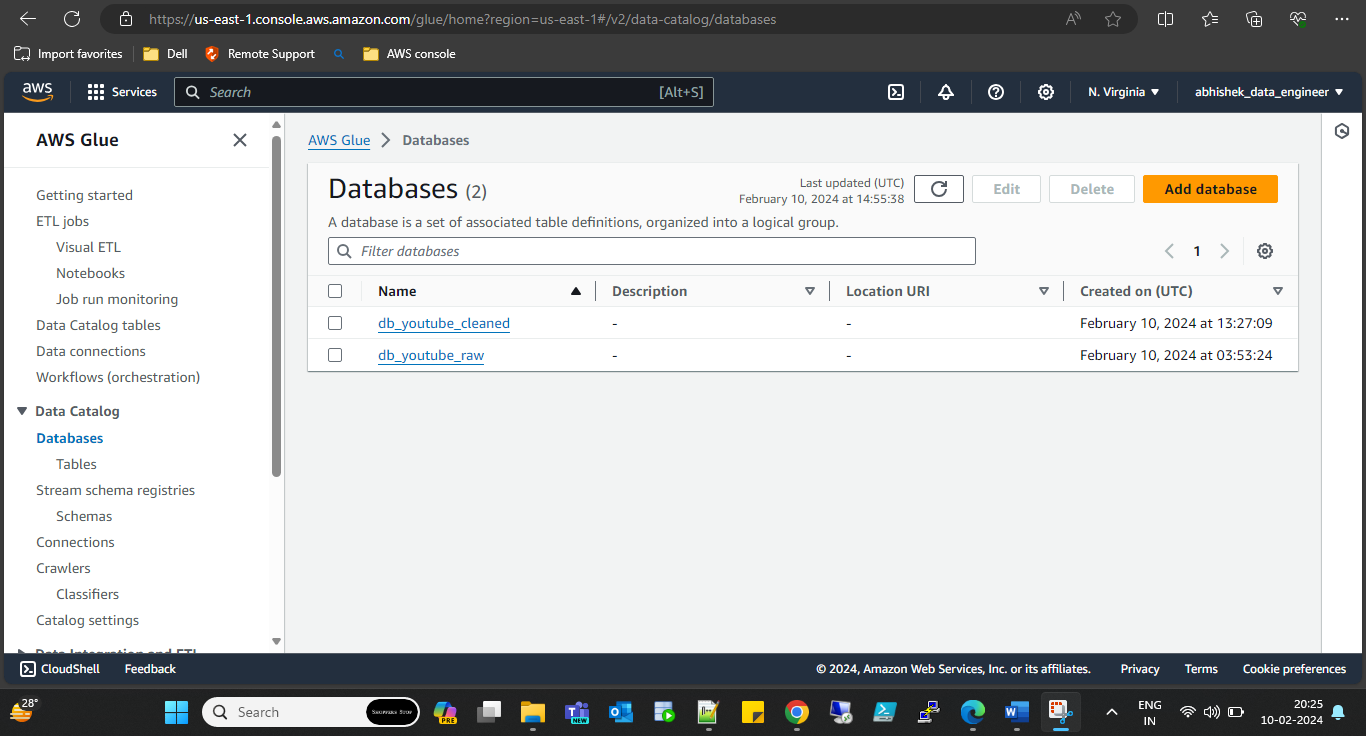


Step3: create catalog of the json data using aws glue

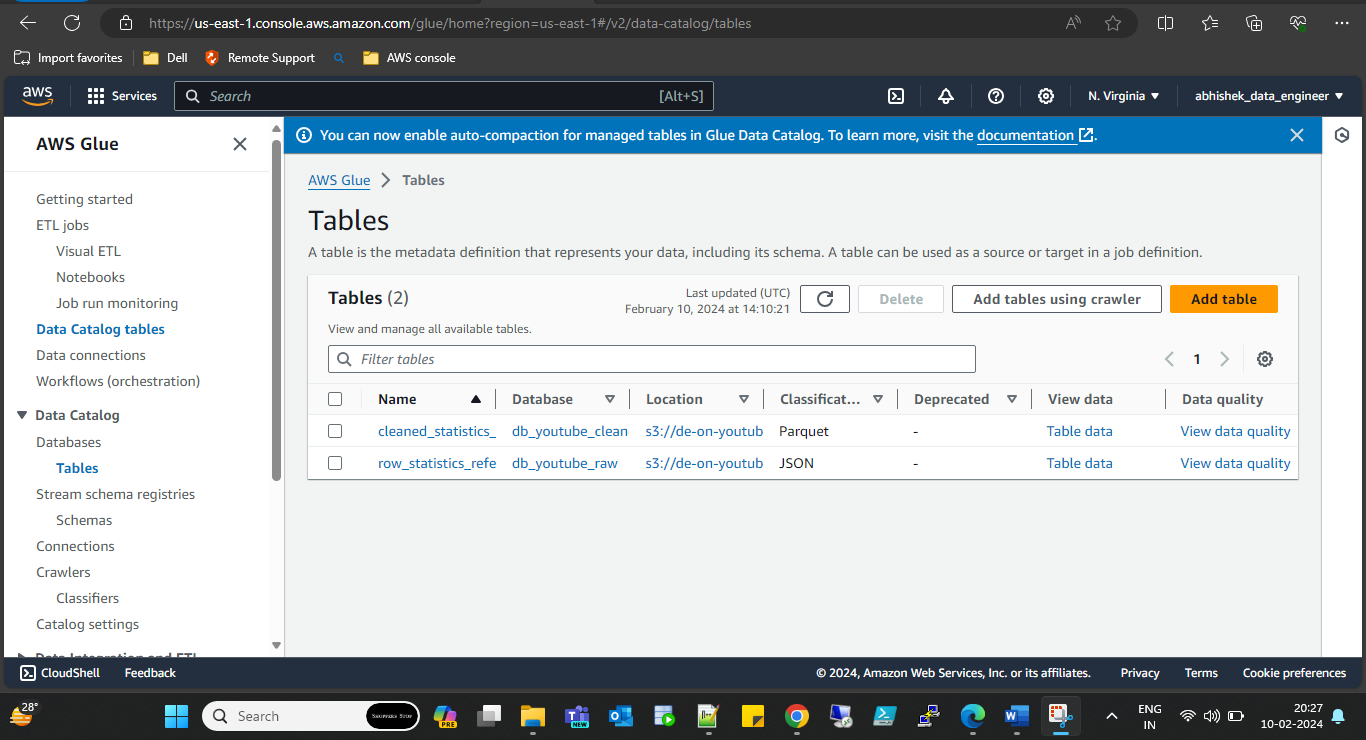




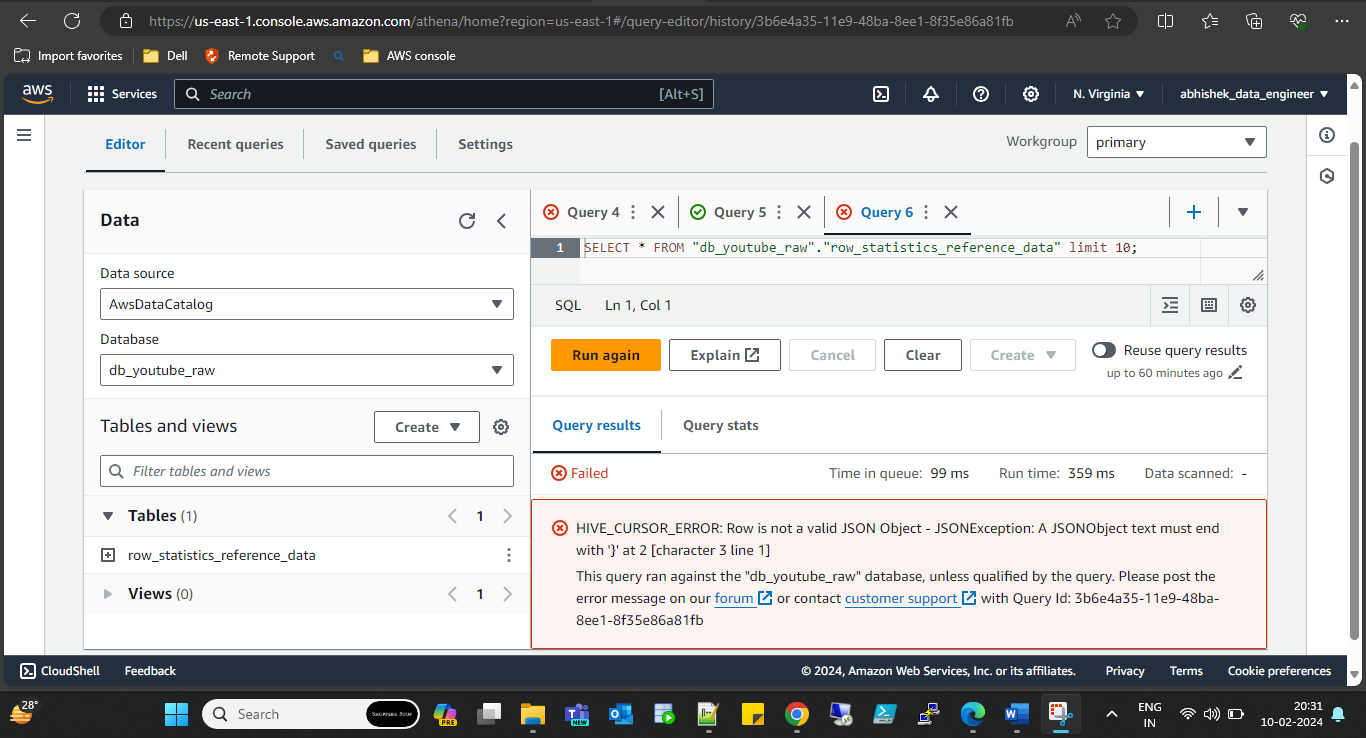
* Database: [db\_youtube\_cleaned](https://us-east-1.console.aws.amazon.com/glue/home?region=us-east-1#/v2/data-catalog/databases/view/db_youtube_cleaned?catalogId=966906844441)



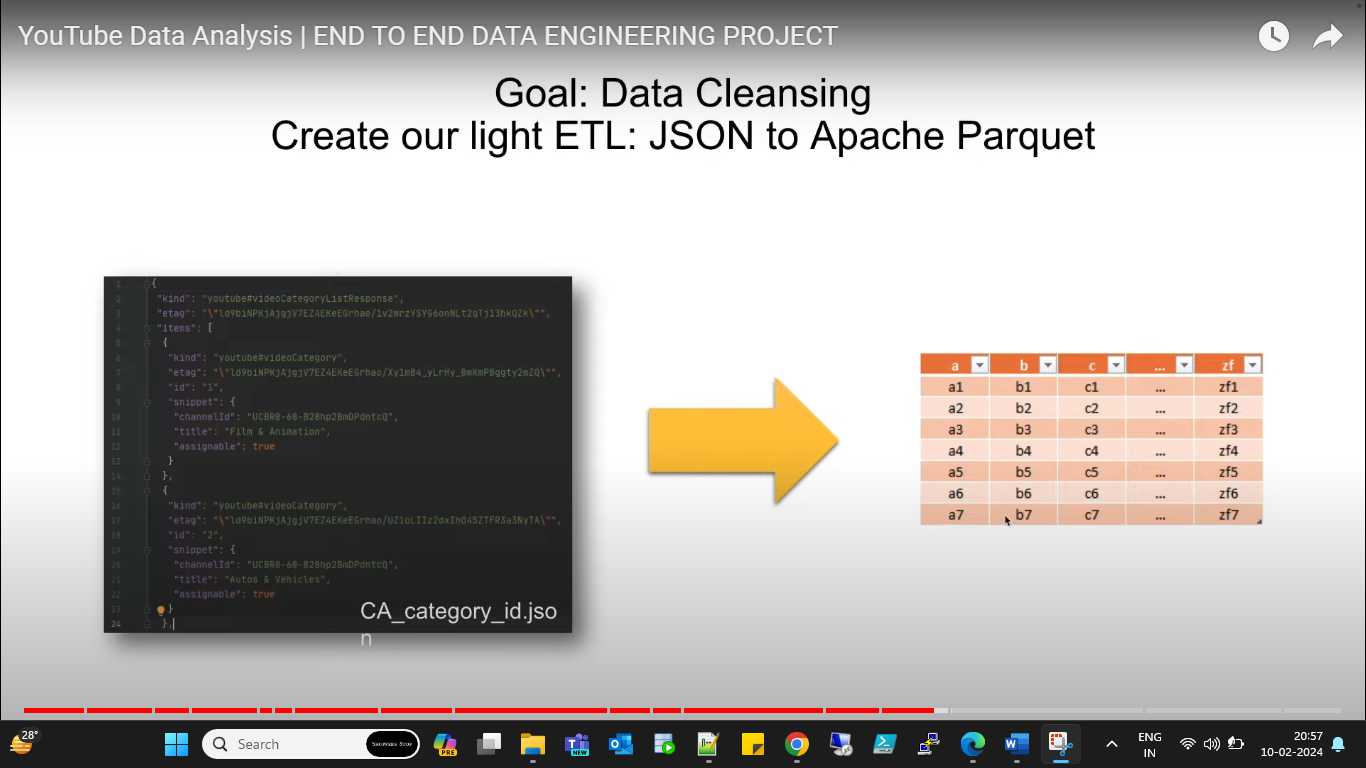
* Tables: [cleaned\_statistics\_reference\_data](https://us-east-1.console.aws.amazon.com/glue/home?region=us-east-1#/v2/data-catalog/tables/view/cleaned_statistics_reference_data?database=db_youtube_cleaned&catalogId=966906844441&versionId=latest)

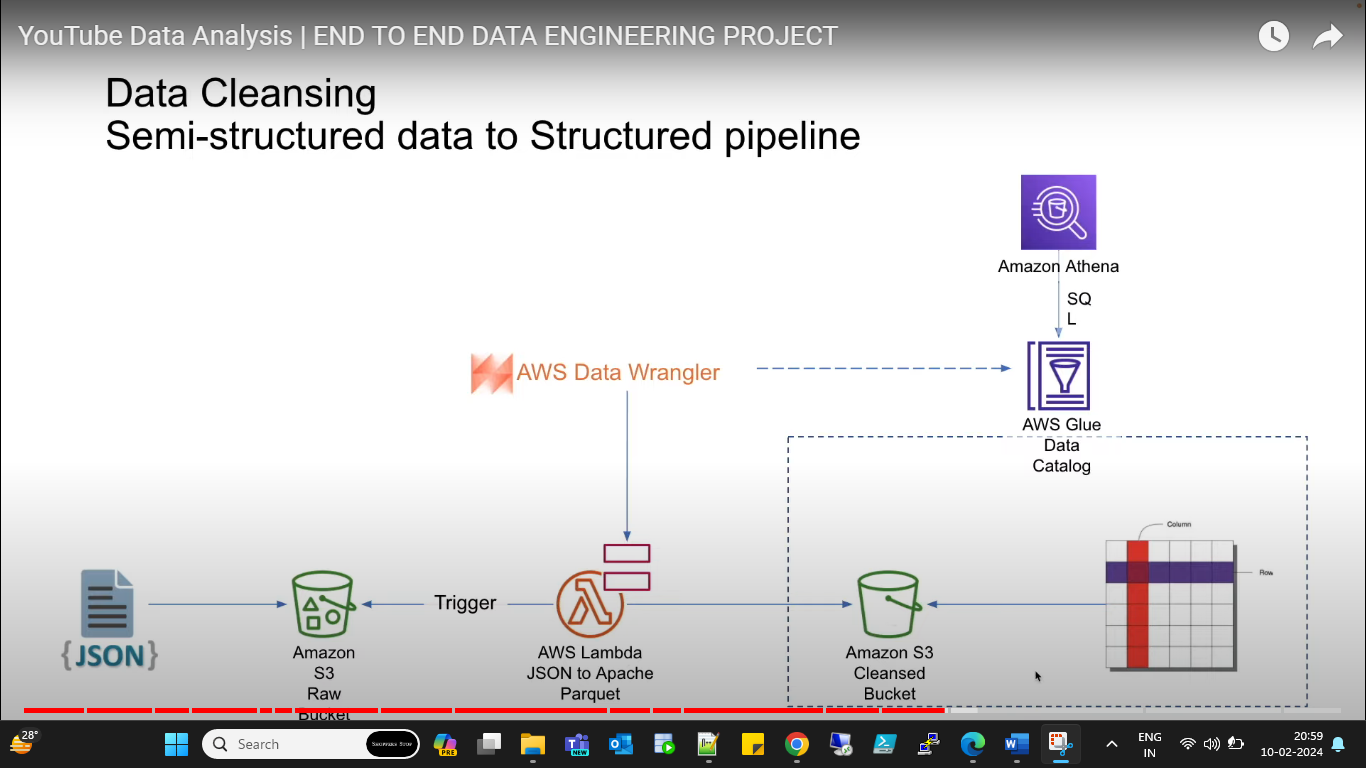


Step4: Try to process the data using aws Anthena but we can not process directly because file type is ‘json’.

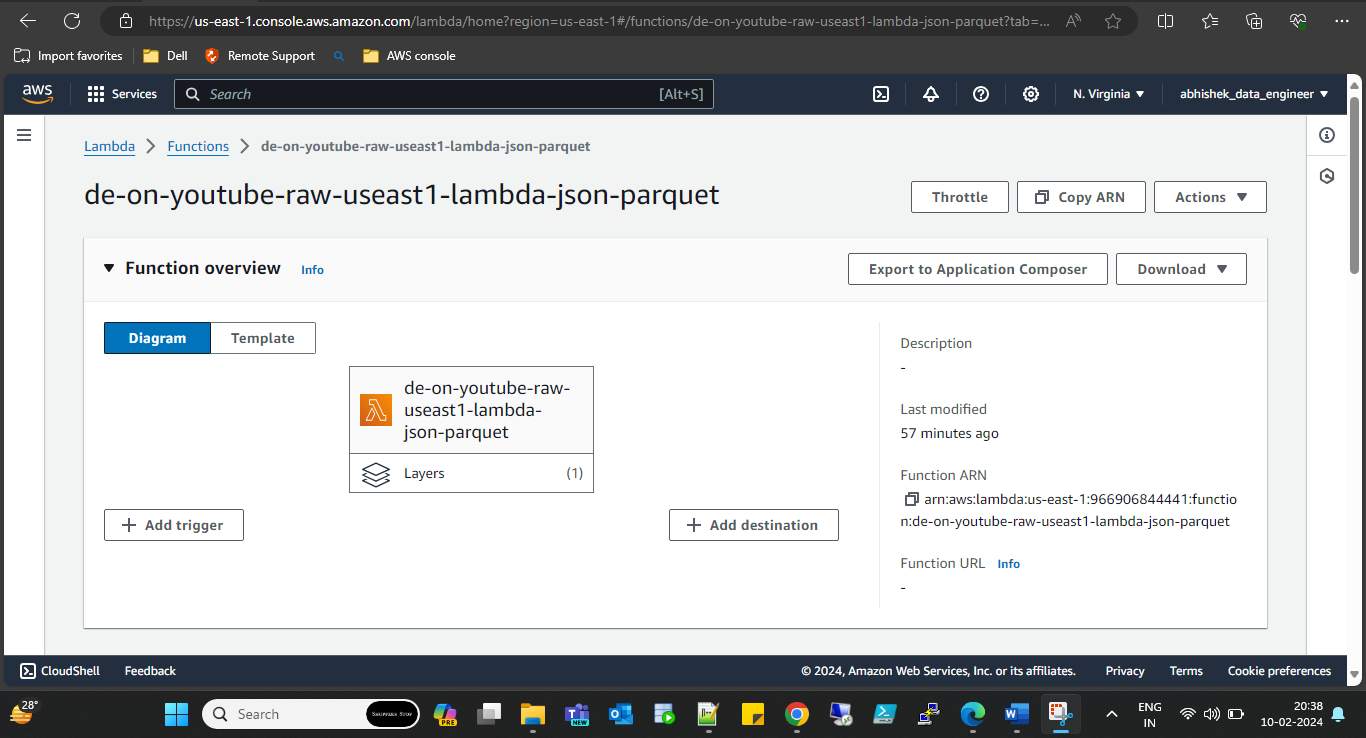


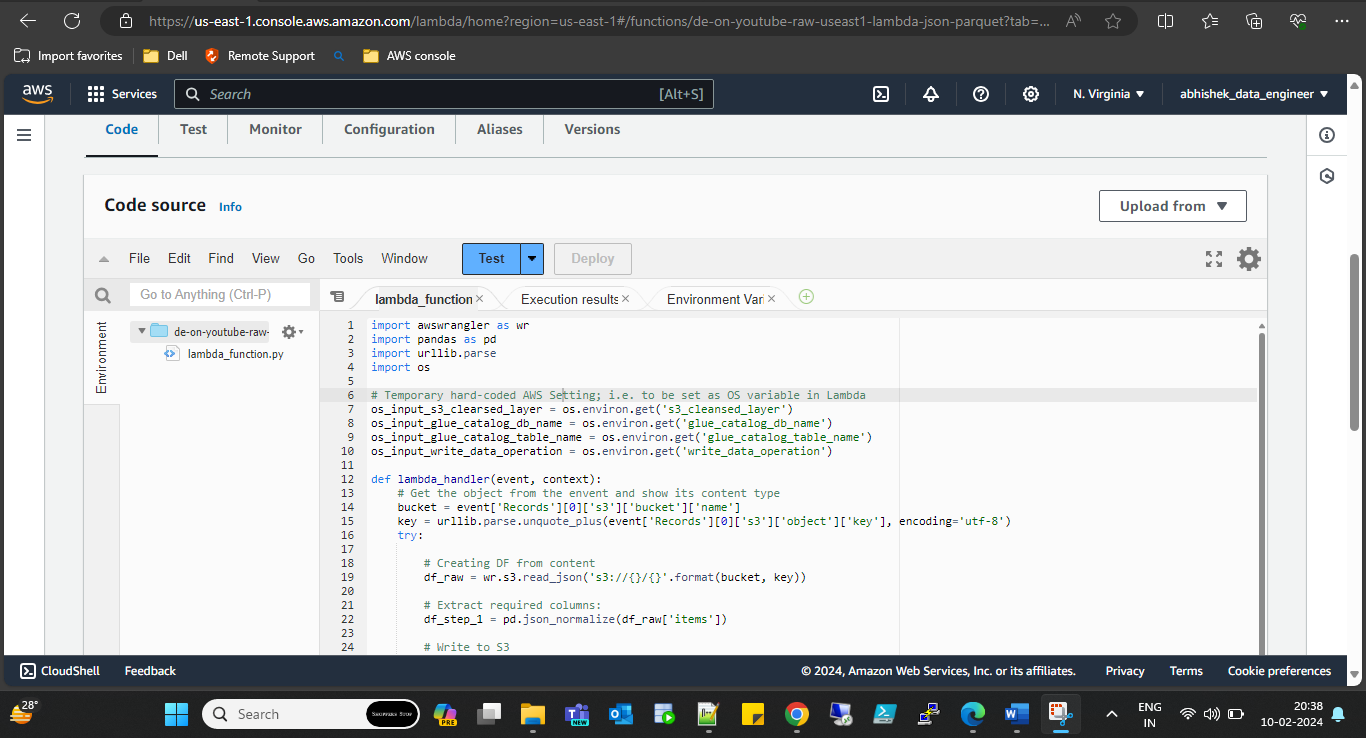
Hence we need to do ETL process on json file using aws lambda function to convert in query able format.





Step5: create aws lambda function to execute python to convert the json file to parquet fomat.





**Python code:**

import awswrangler as wr

import pandas as pd

import urllib.parse

import os

# Temporary hard-coded AWS Setting; i.e. to be set as OS variable in Lambda

os\_input\_s3\_clearsed\_layer = os.environ.get('s3\_cleansed\_layer')

os\_input\_glue\_catalog\_db\_name = os.environ.get('glue\_catalog\_db\_name')

os\_input\_glue\_catalog\_table\_name = os.environ.get('glue\_catalog\_table\_name')

os\_input\_write\_data\_operation = os.environ.get('write\_data\_operation')

def lambda\_handler(event, context):

# Get the object from the envent and show its content type

bucket = event['Records'][0]['s3']['bucket']['name']

key = urllib.parse.unquote\_plus(event['Records'][0]['s3']['object']['key'], encoding='utf-8')

try:

# Creating DF from content

df\_raw = wr.s3.read\_json('s3://{}/{}'.format(bucket, key))

# Extract required columns:

df\_step\_1 = pd.json\_normalize(df\_raw['items'])

# Write to S3

wr\_response = wr.s3.to\_parquet(

df = df\_step\_1,

dataset = True,

database = os\_input\_glue\_catalog\_db\_name,

table = os\_input\_glue\_catalog\_table\_name,

mode = os\_input\_write\_data\_operation

)

return wr\_response

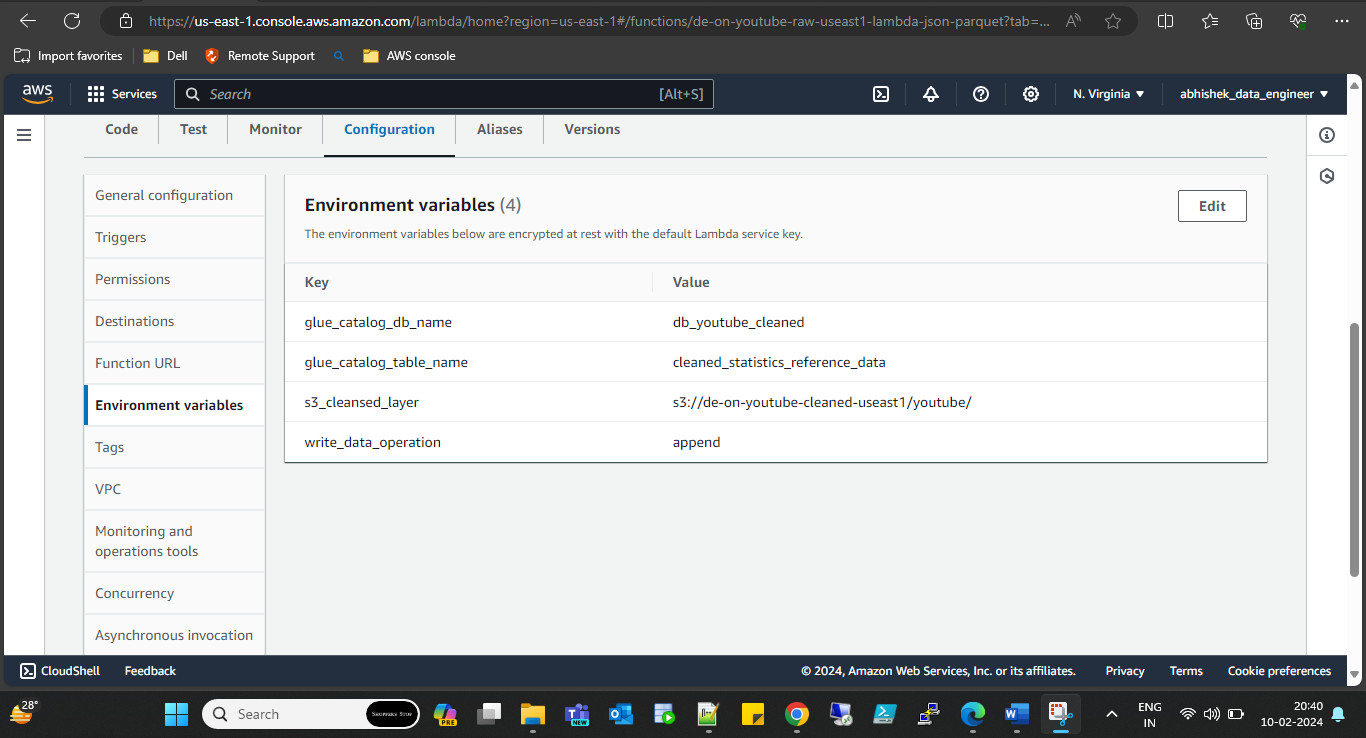
except Exception as e:

print(e)

print('Error getting object {} from bucket {}. Make sure they exist and your bucket is in the same region as this function.'.format(key, bucket))

raise e

Step7: set the environment variable to fetch or create target location.



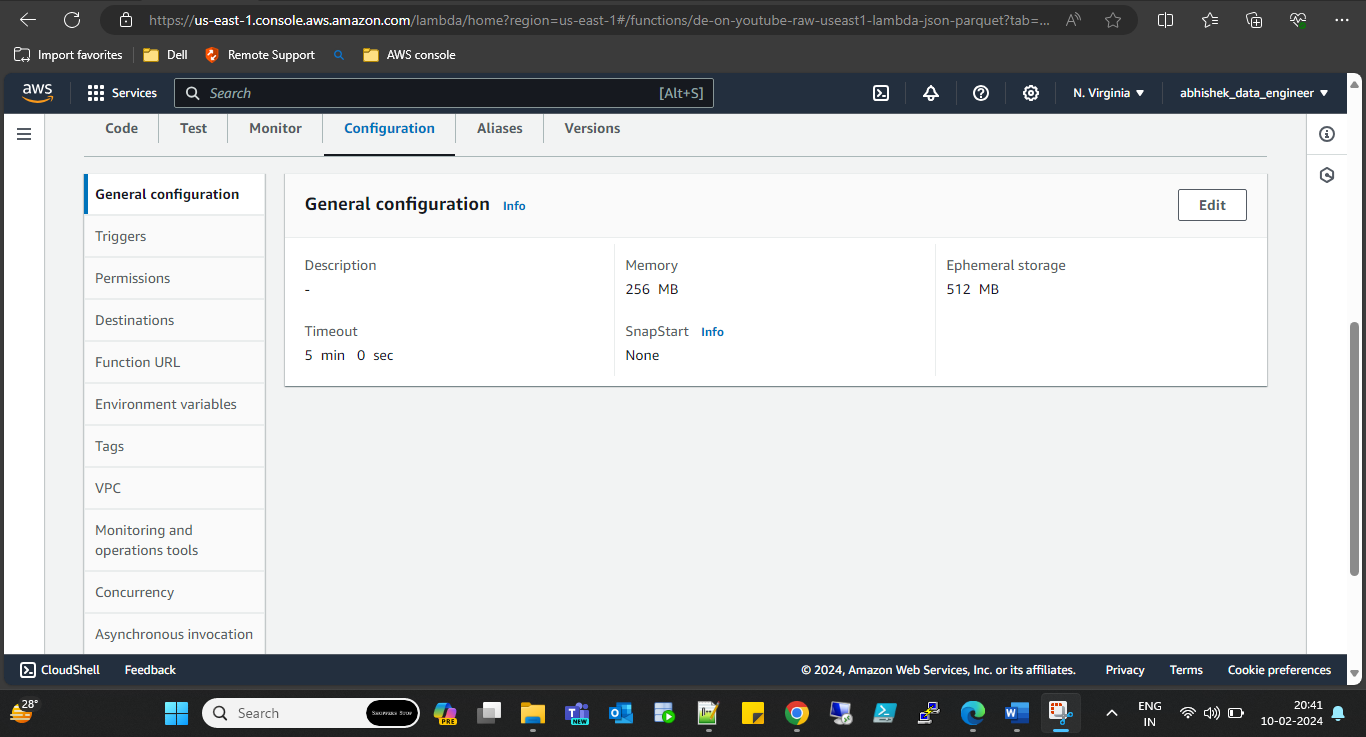
glue\_catalog\_db\_name=db\_youtube\_cleaned

glue\_catalog\_table\_name=cleaned\_statistics\_reference\_data

s3\_cleansed\_layer="your bucket name"

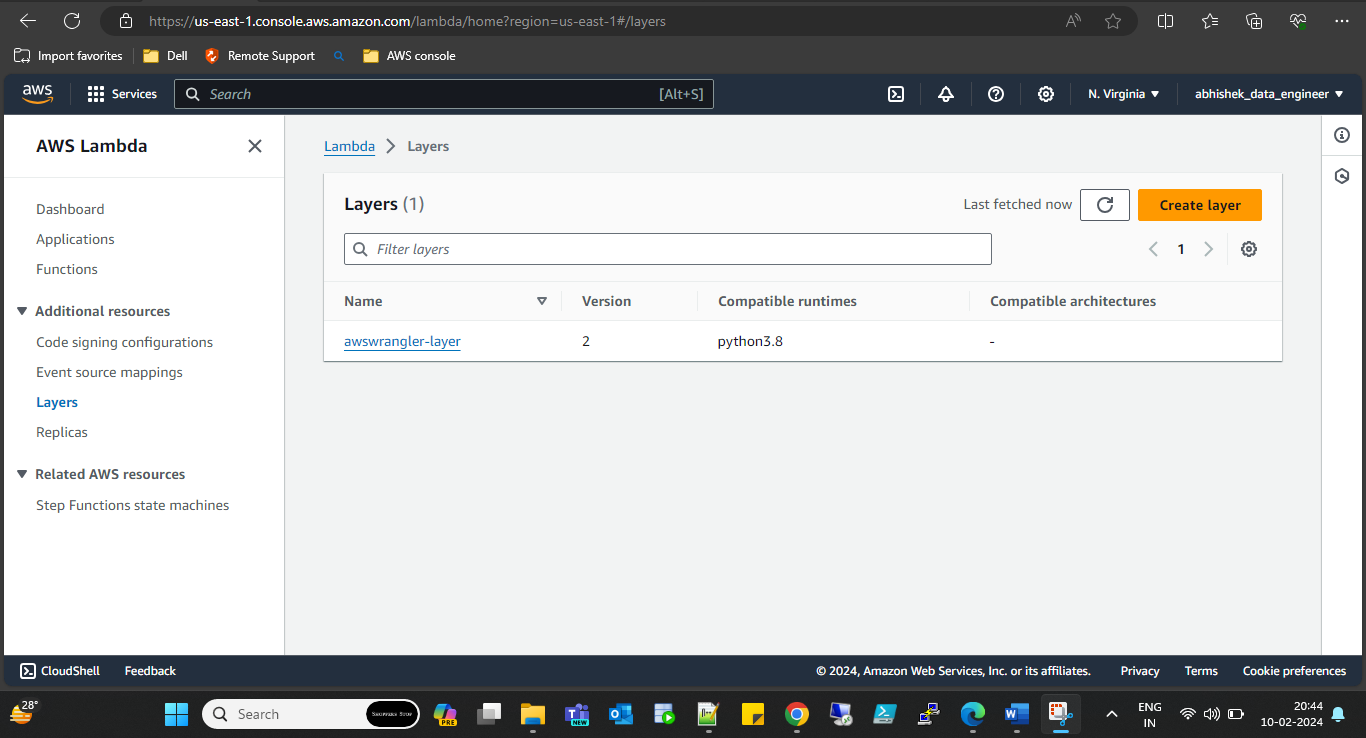
write\_data\_operation=append

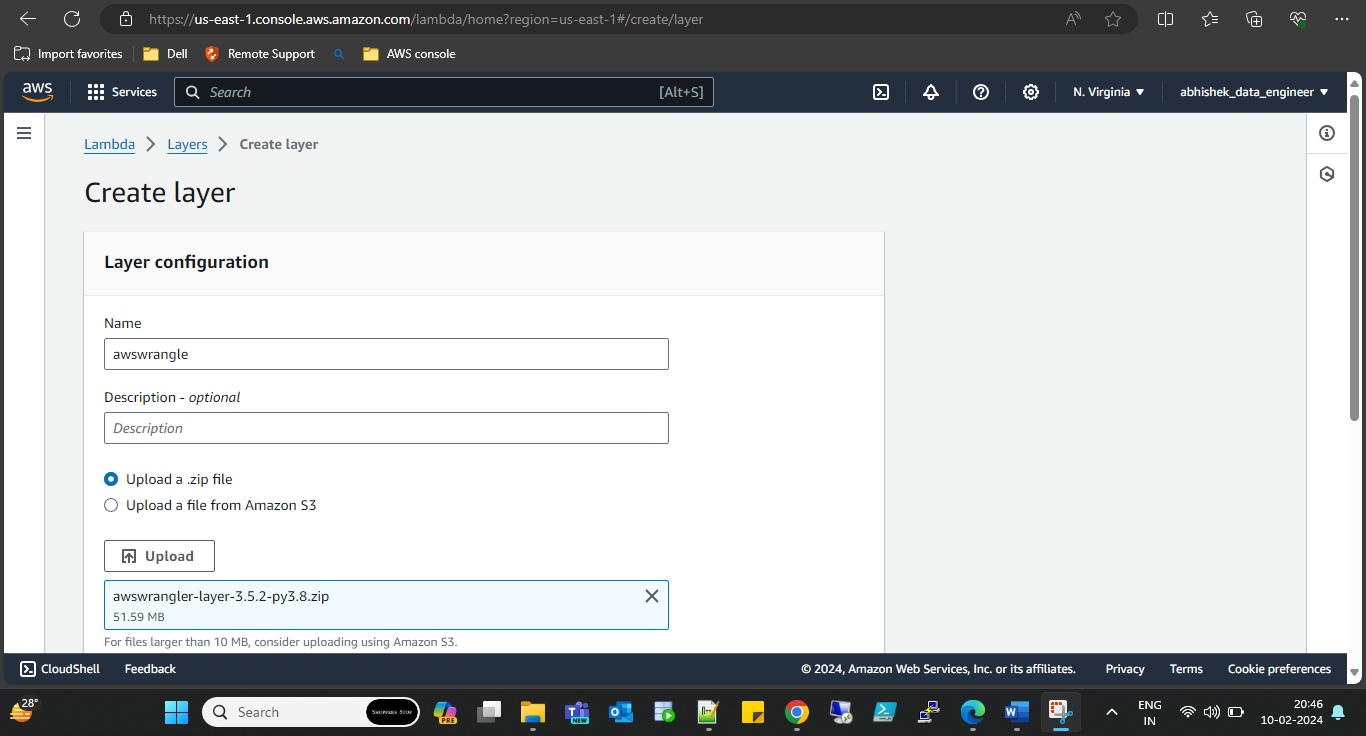
Increase time of execution for lambda function.

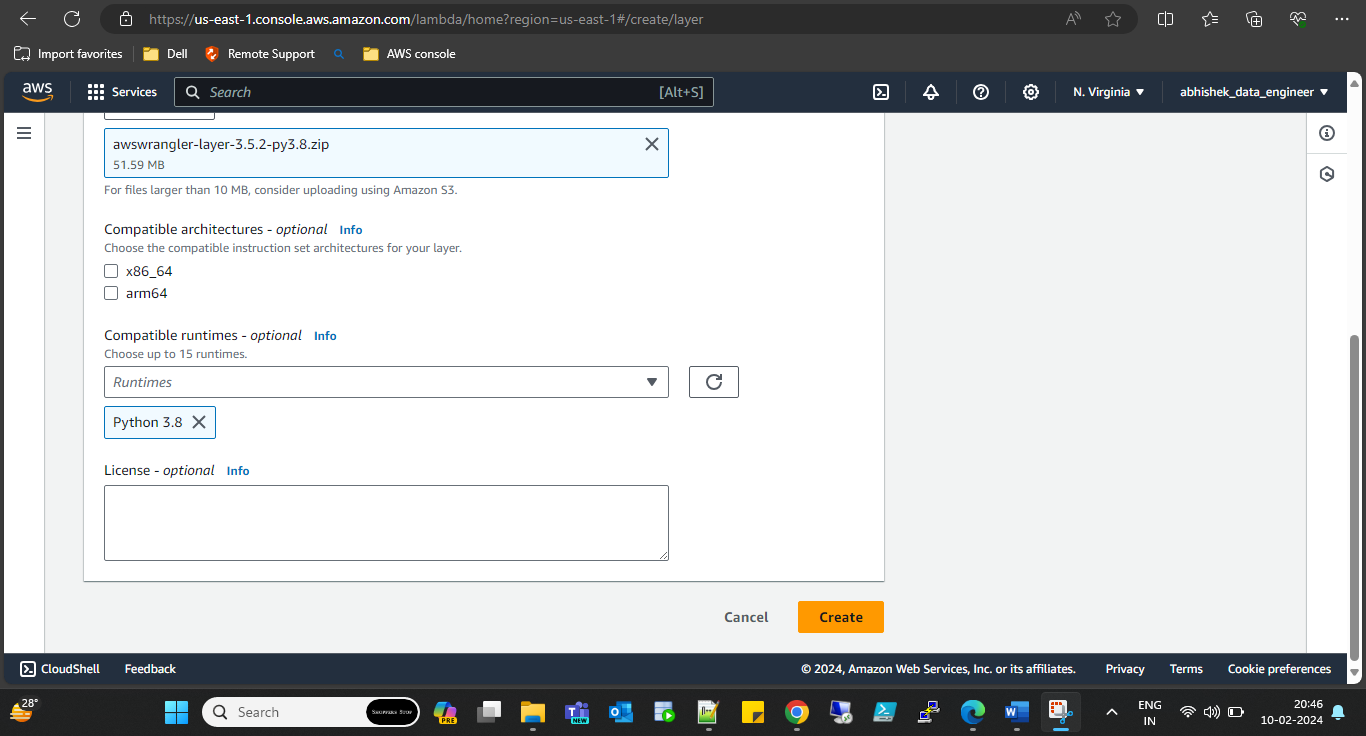


Step8: while running lambda function got an error of “awswrangle” module does not exist. For this we need create and then add layer.

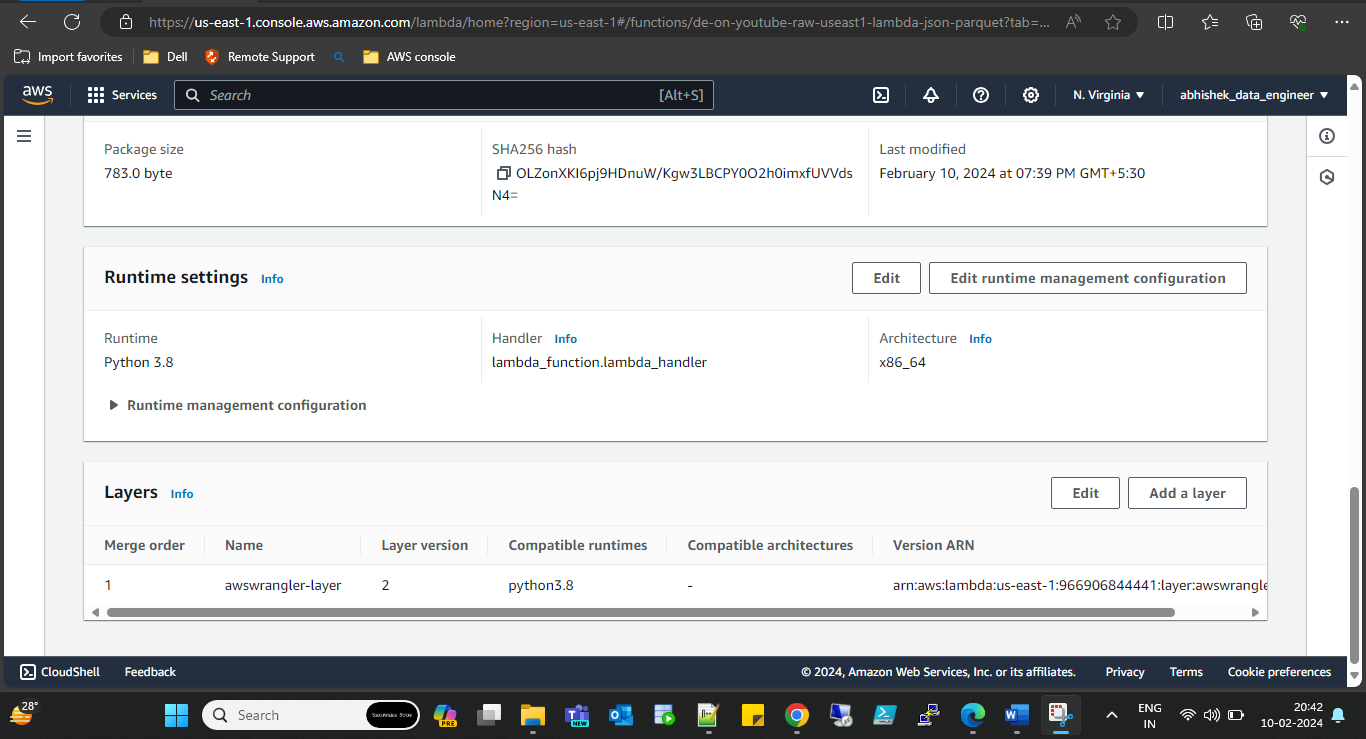
**Creating layer:**



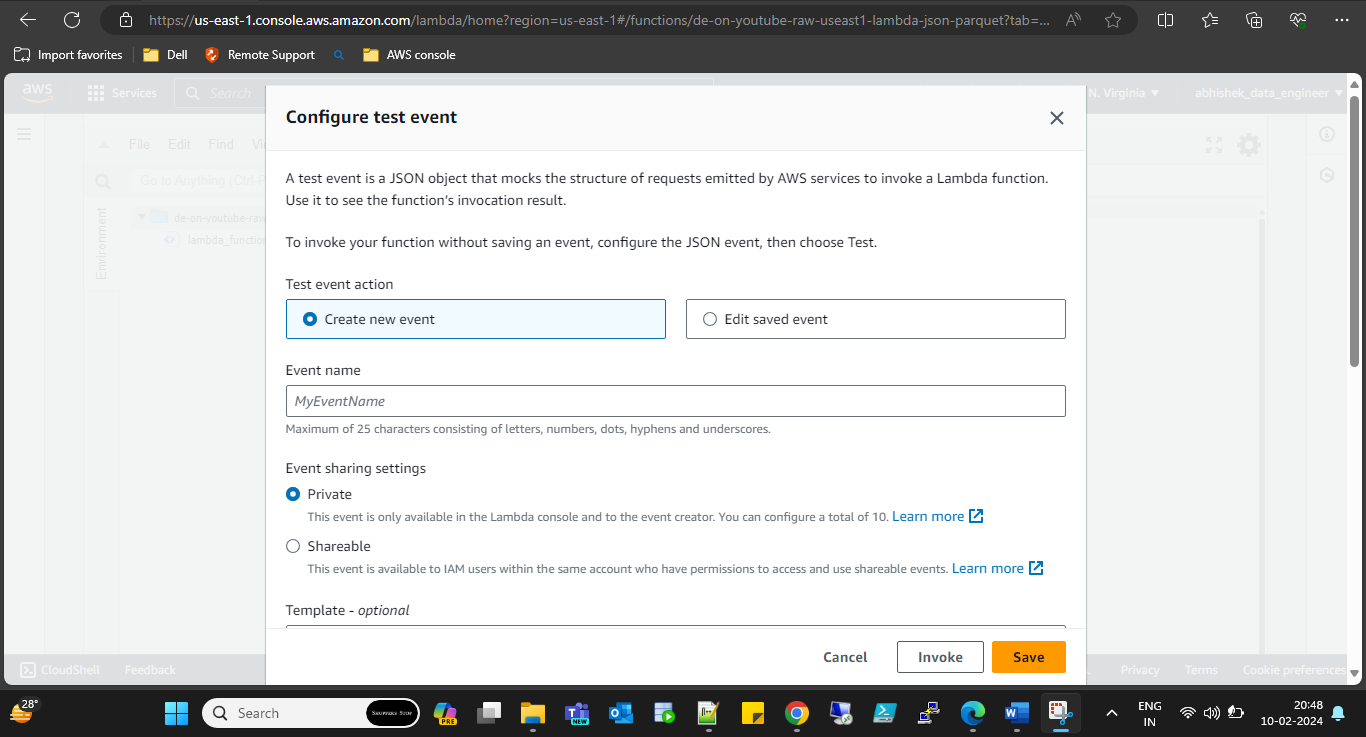


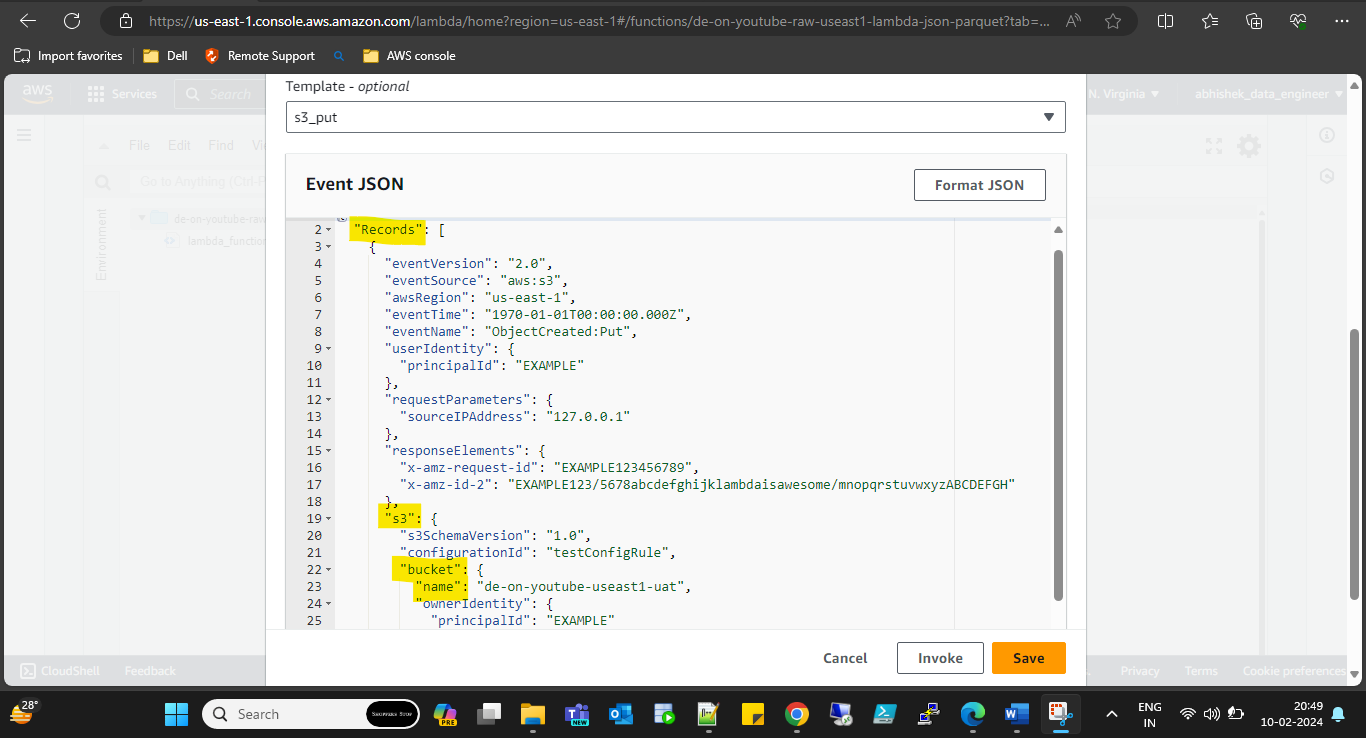


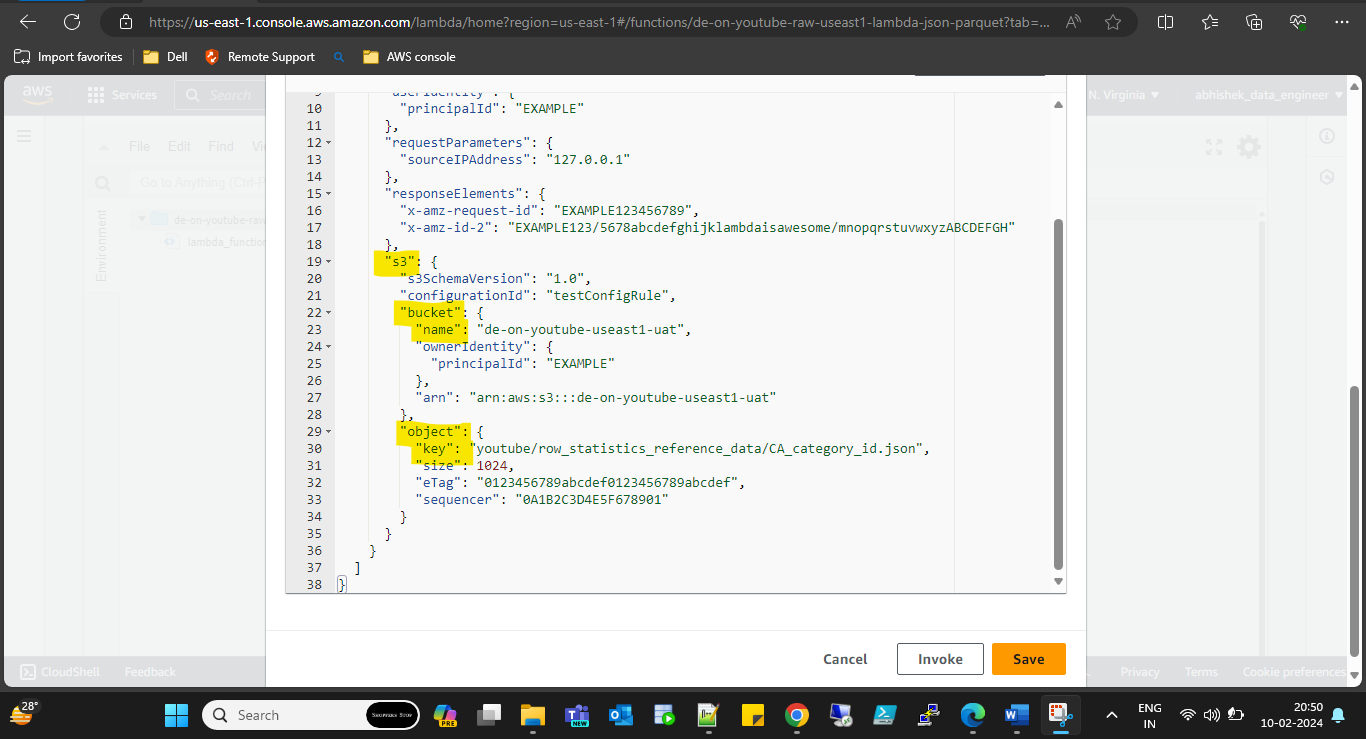
**Adding layer with aws lambda function:**



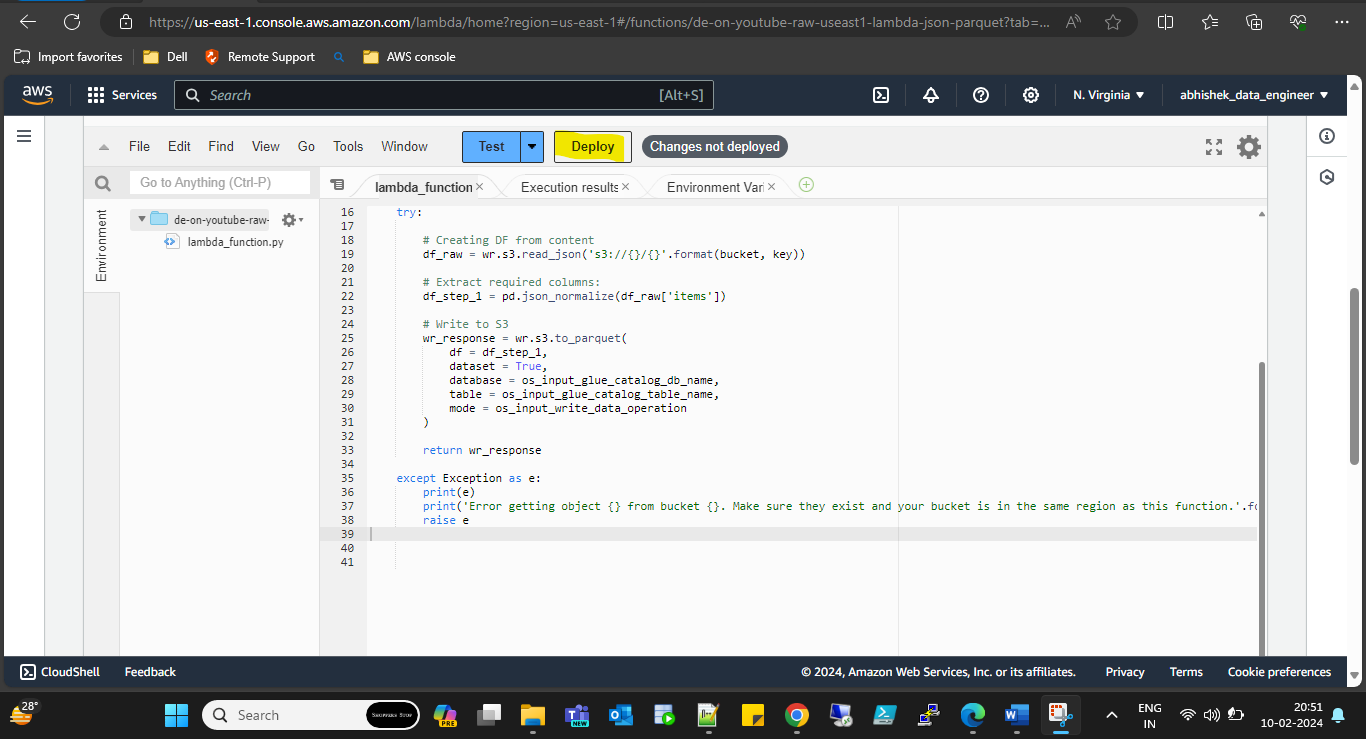
Step9: create test event to test lambda function. Gives S3 bucket and file path which want to read.





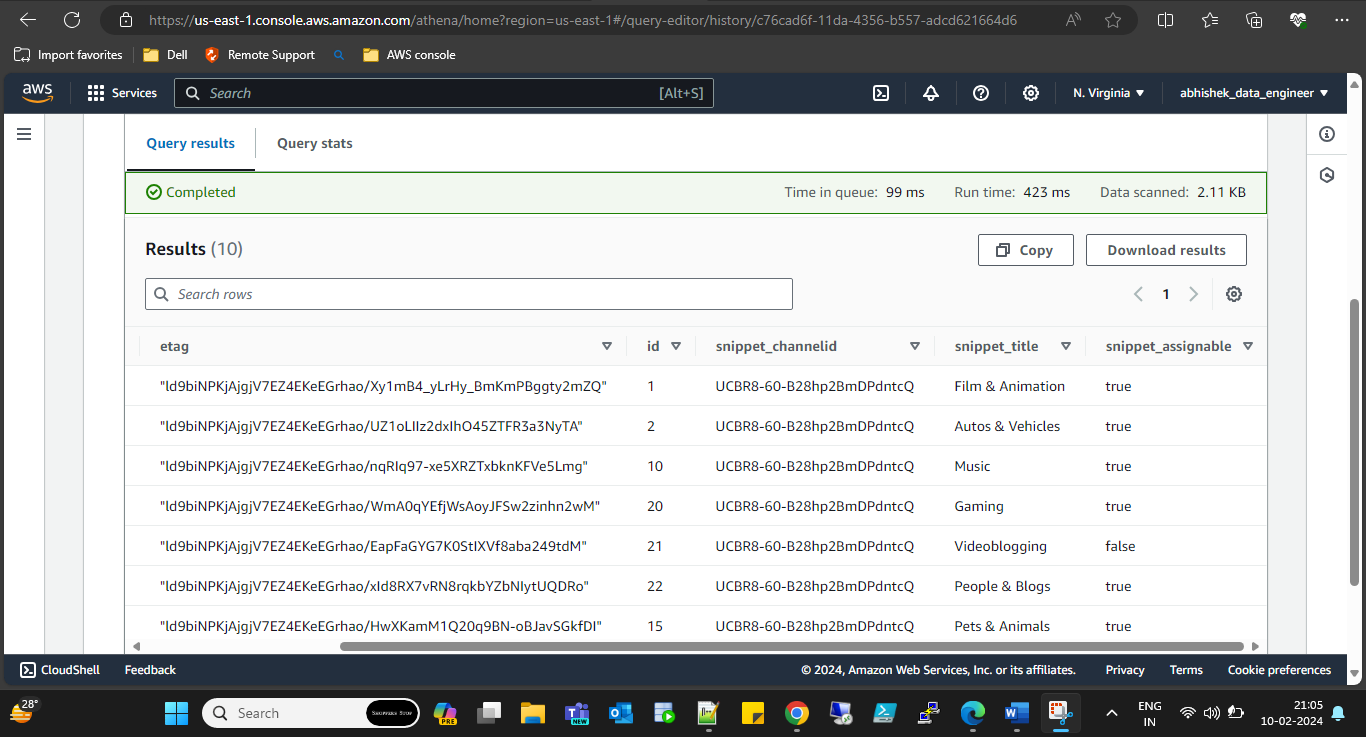


Step10: click of deploy to deploy code for testing.



Step11: cleaned data uploaded into table & database in aws glues which we created.





**Final Architecture diagram of project.**

