CSCI 5408

DATA MANAGEMENT AND WAREHOUSING

LAB ASSIGNMENT - 5

Submitted By:	Kenil Shaileshkumar Patel (kenil.patel@dal.ca)
Banner ID:	B00954251
Submitted On:	November 04, 2023

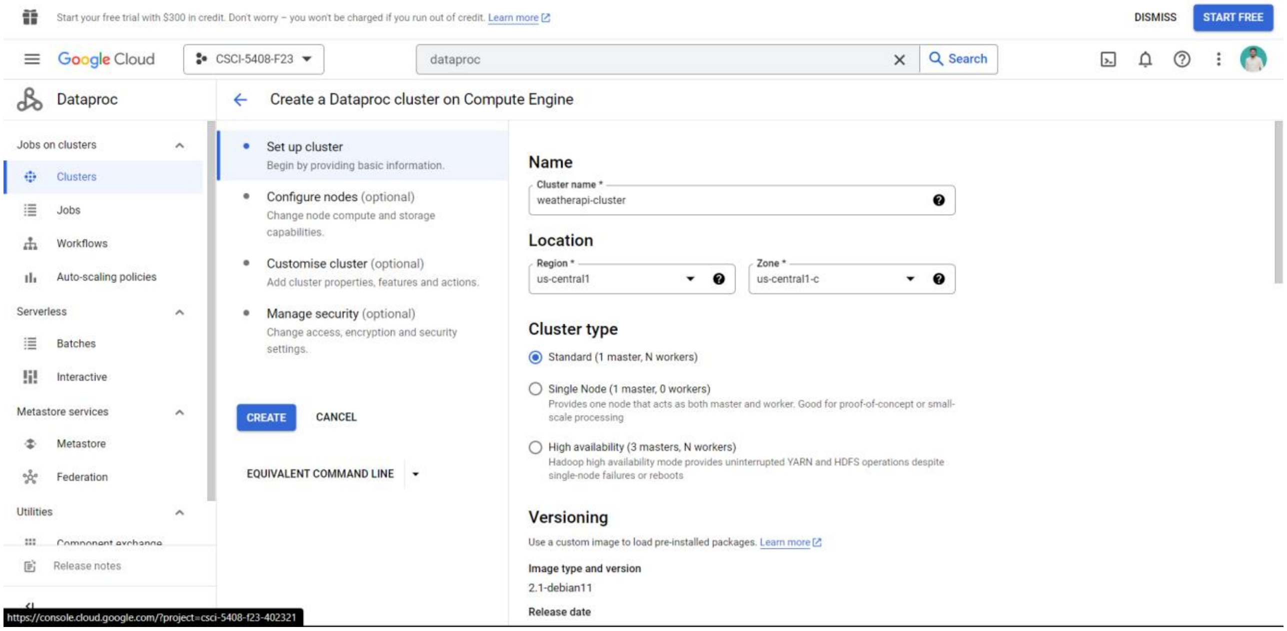
Gitlab Repository Link

https://git.cs.dal.ca/kenil/csci5408_f23_b00954251_kenil_patel/-/tree/main/Lab5

Table of Contents

Setting up Apache Spark on GCP instances

Created a Dataproc cluster on computer engine and also modified the nodes



Create a Dataproc cluster on Compute Engine

- Set up cluster — Begin by providing basic information.
- Configure nodes (optional) — Change node compute and storage capabilities.
- Customise cluster (optional) — Add cluster properties, features and actions.
- Manage security (optional) — Change access, encryption and security settings.

CREATE CANCEL

EQUIVALENT COMMAND LINE

Name

Cluster name *: weatherapi-cluster

Location

Region *: us-central1

Zone *: us-central1-c

Cluster type

● Standard (1 master, N workers)

○ Single Node (1 master, 0 workers)
Provides one node that acts as both master and worker. Good for proof-of-concept or small-scale processing

○ High availability (3 masters, N workers)
Hadoop high availability mode provides uninterrupted YARN and HDFS operations despite single-node failures or reboots

Versioning

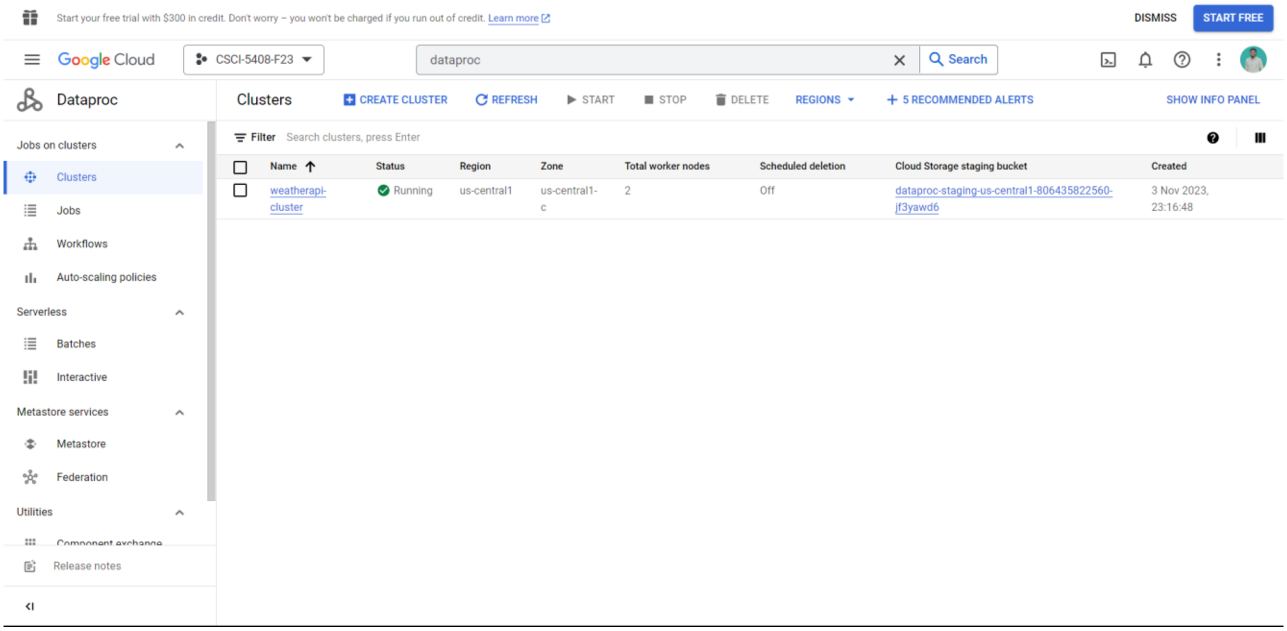
Use a custom image to load pre-installed packages. Learn more

Image type and version
2.1-debian11

Release date

Figure 1: Setting Up the Cluster.

The cluster for the weatherapi is created and it is successfully running.

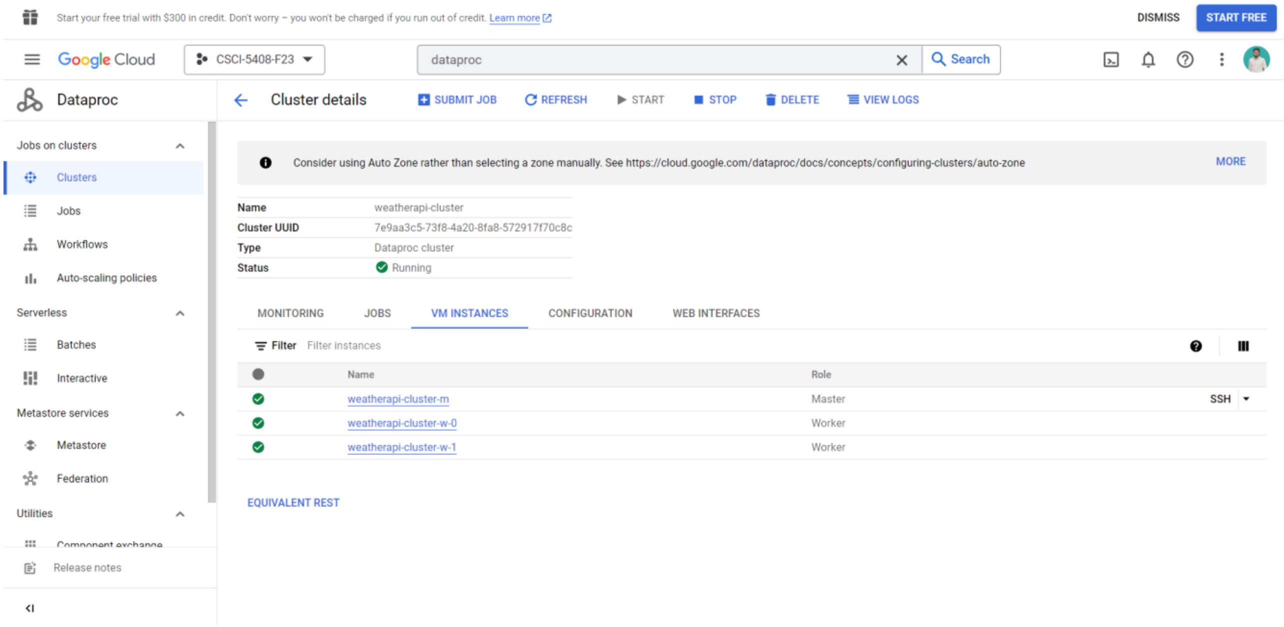


Clusters

Name	Status	Region	Zone	Total worker nodes	Scheduled deletion	Cloud Storage staging bucket	Created
weatherapi-cluster	✓ Running	us-central1	us-central1-c	2	Off	dataproc-staging-us-central1-806435822560-jf3yawd6	3 Nov 2023, 23:16:48

Figure 2: weatherapi-cluster is running.

In the cluster, there are three VM instances in which one master node and other two worker node.



Cluster details

Consider using Auto Zone rather than selecting a zone manually. See https://cloud.google.com/dataproc/docs/concepts/configuring-clusters/auto-zone

Name	weatherapi-cluster
Cluster UUID	7e9aa3c5-73f8-4a20-8fa8-572917f70c8c
Type	Dataproc cluster
Status	✓ Running

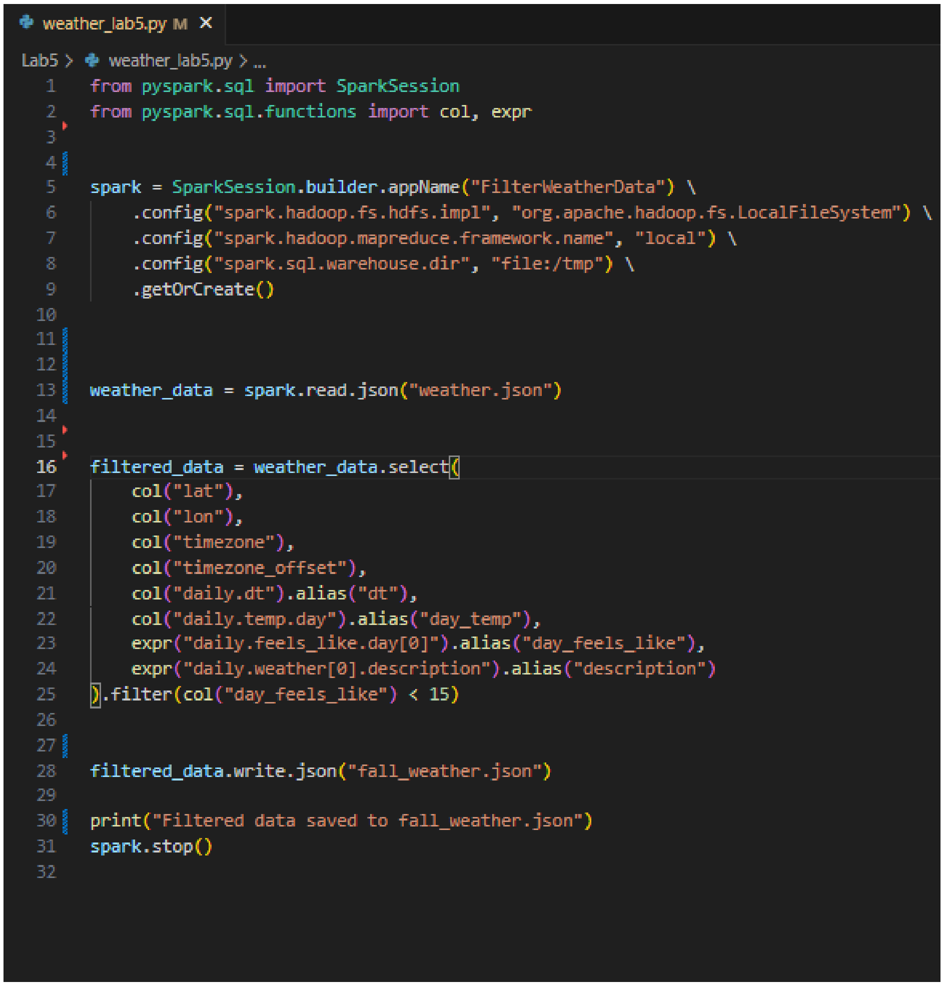
●	Name	Role
✓	weatherapi-cluster-m	Master
✓	weatherapi-cluster-w-0	Worker
✓	weatherapi-cluster-w-1	Worker

Figure 3: VM instances in the cluster.

Spark Program (Python)

Write Spark program to filter the response data where the daily “feels_like” temperature for the next 5-days is less than 15°C during the “day” time. Exclude the current, minutely, and hourly fields.

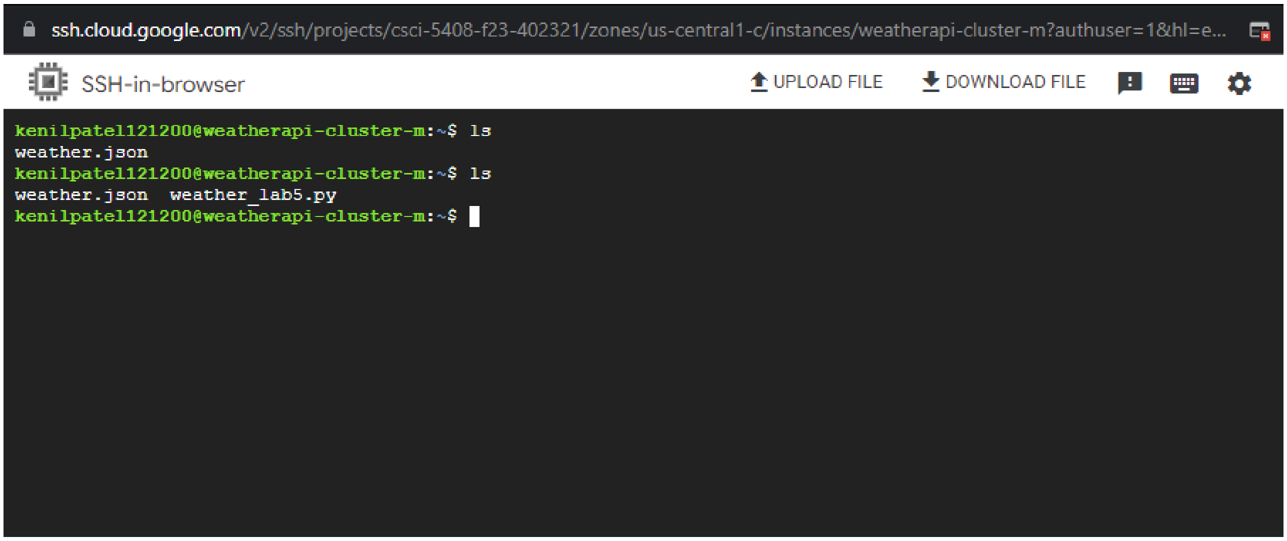
Below is the Python program which loads the data from the JSON file and processes the data. It filters the data where the daily “feels_like” temperature for the next 5 days is less than 15°C during the “day” time.



```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, expr


spark = SparkSession.builder.appName("FilterWeatherData") \
    .config("spark.hadoop.fs.hdfs.impl", "org.apache.hadoop.fs.LocalFileSystem") \
    .config("spark.hadoop.mapreduce.framework.name", "local") \
    .config("spark.sql.warehouse.dir", "file:/tmp") \
    .getOrCreate()



weather_data = spark.read.json("weather.json")


filtered_data = weather_data.select(
    col("lat"),
    col("lon"),
    col("timezone"),
    col("timezone_offset"),
    col("daily.dt").alias("dt"),
    col("daily.temp.day").alias("day_temp"),
    expr("daily.feels_like.day[0]").alias("day_feels_like"),
    expr("daily.weather[0].description").alias("description")
).filter(col("day_feels_like") < 15)


filtered_data.write.json("fall_weather.json")

print("Filtered data saved to fall_weather.json")
spark.stop()
```

Figure 4: Spark Python Code

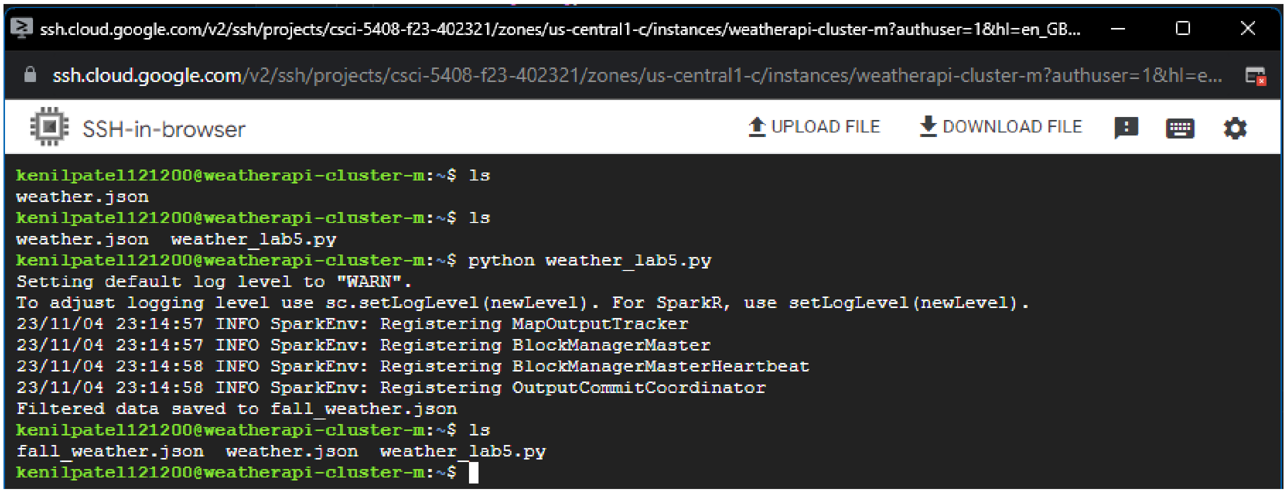
I have uploaded weather.json and weather_lab5.py file.


ssh.cloud.google.com/v2/ssh/projects/csci-5408-f23-402321/zones/us-central1-c/instances/weatherapi-cluster-m?authuser=1&hl=e...
SSH-in-browser
UPLOAD FILE
DOWNLOAD FILE

```
kenilpatel121200@weatherapi-cluster-m:~$ ls
weather.json
kenilpatel121200@weatherapi-cluster-m:~$ ls
weather.json  weather_lab5.py
kenilpatel121200@weatherapi-cluster-m:~$
```

Figure 5: File Uploading.

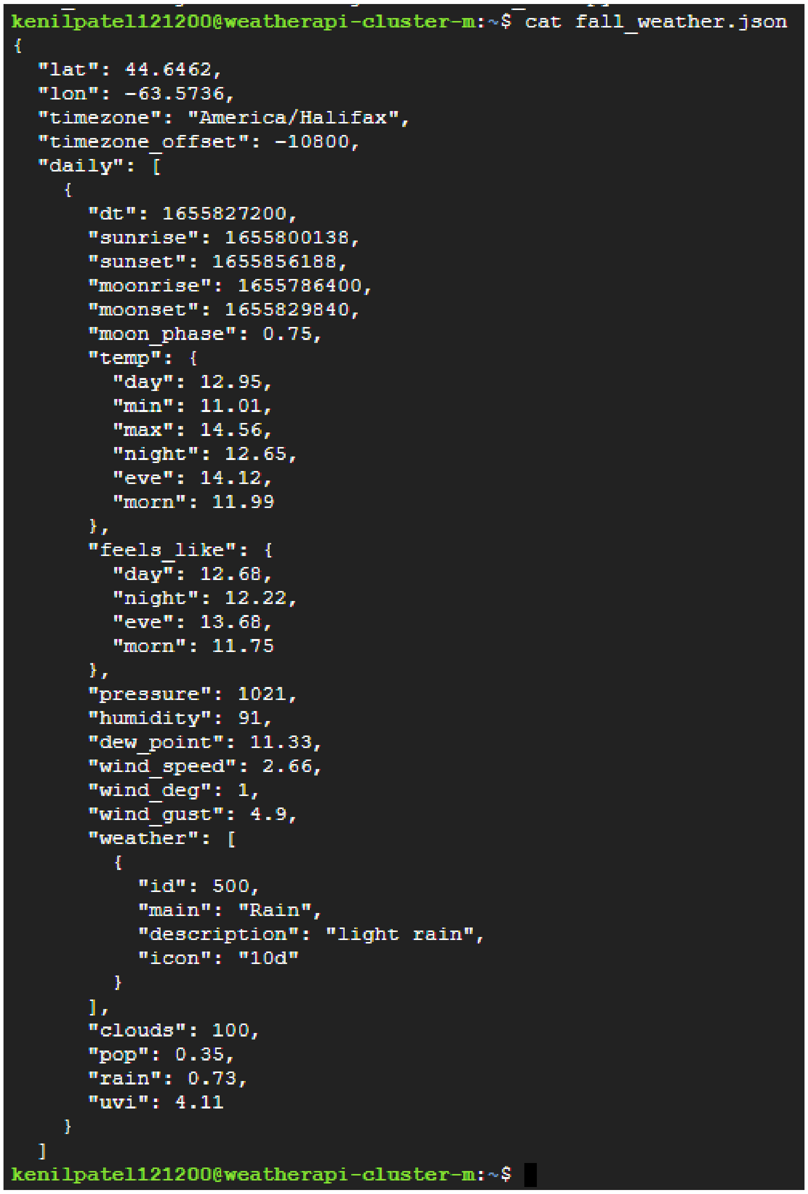
I am running the python file and the fall_weather.json file is generated.


ssh.cloud.google.com/v2/ssh/projects/csci-5408-f23-402321/zones/us-central1-c/instances/weatherapi-cluster-m?authuser=1&hl=en_GB...
ssh.cloud.google.com/v2/ssh/projects/csci-5408-f23-402321/zones/us-central1-c/instances/weatherapi-cluster-m?authuser=1&hl=e...
SSH-in-browser
UPLOAD FILE
DOWNLOAD FILE

```
kenilpatel121200@weatherapi-cluster-m:~$ ls
weather.json
kenilpatel121200@weatherapi-cluster-m:~$ ls
weather.json  weather_lab5.py
kenilpatel121200@weatherapi-cluster-m:~$ python weather_lab5.py
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
23/11/04 23:14:57 INFO SparkEnv: Registering MapOutputTracker
23/11/04 23:14:57 INFO SparkEnv: Registering BlockManagerMaster
23/11/04 23:14:58 INFO SparkEnv: Registering BlockManagerMasterHeartbeat
23/11/04 23:14:58 INFO SparkEnv: Registering OutputCommitCoordinator
Filtered data saved to fall_weather.json
kenilpatel121200@weatherapi-cluster-m:~$ ls
fall_weather.json  weather.json  weather_lab5.py
kenilpatel121200@weatherapi-cluster-m:~$
```

Figure 6: Running the Python file.

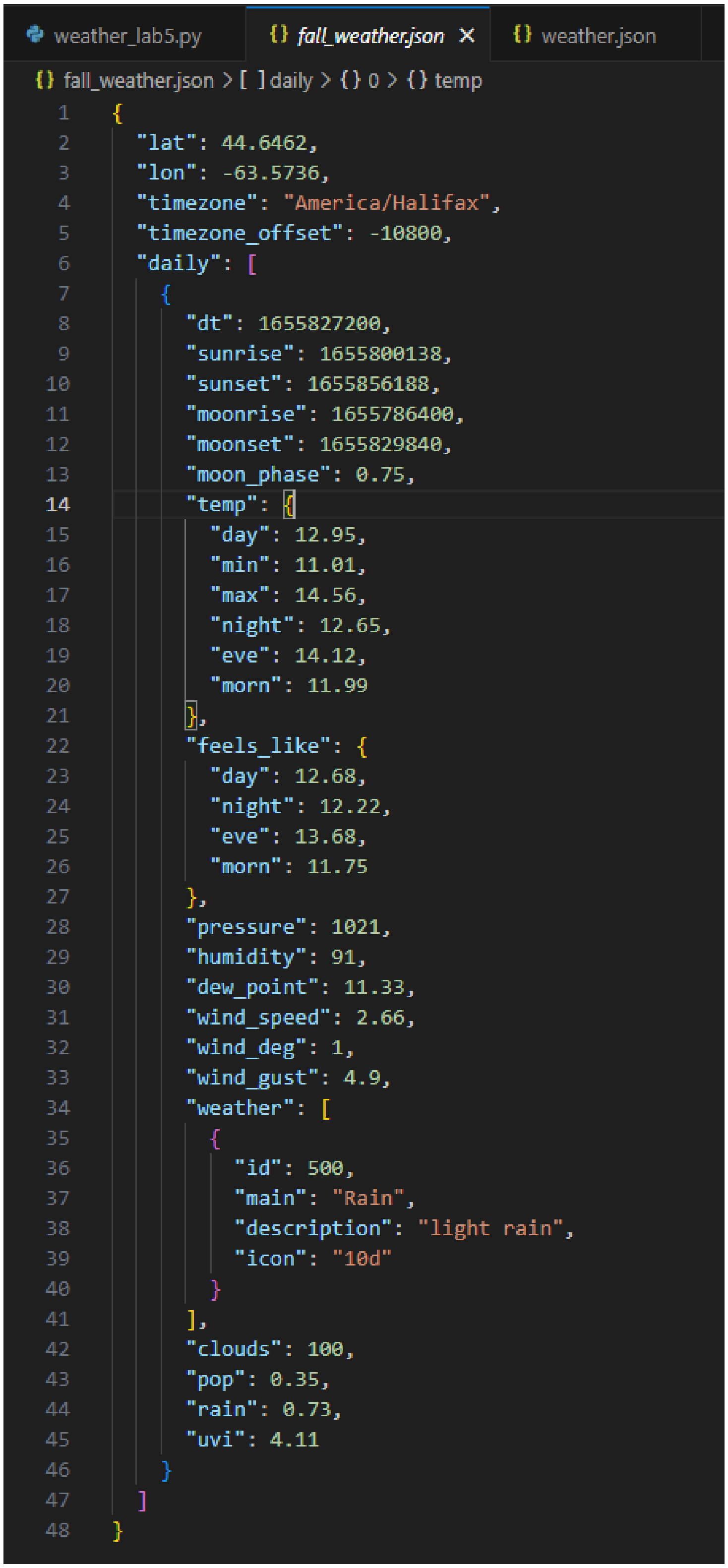
Displaying the fall_weather.json in command prompt.



```
kenilpatel121200@weatherapi-cluster-m:~$ cat fall_weather.json
{
  "lat": 44.6462,
  "lon": -63.5736,
  "timezone": "America/Halifax",
  "timezone_offset": -10800,
  "daily": [
    {
      "dt": 1655827200,
      "sunrise": 1655800138,
      "sunset": 1655856188,
      "moonrise": 1655786400,
      "moonset": 1655829840,
      "moon_phase": 0.75,
      "temp": {
        "day": 12.95,
        "min": 11.01,
        "max": 14.56,
        "night": 12.65,
        "eve": 14.12,
        "morn": 11.99
      },
      "feels_like": {
        "day": 12.68,
        "night": 12.22,
        "eve": 13.68,
        "morn": 11.75
      },
      "pressure": 1021,
      "humidity": 91,
      "dew_point": 11.33,
      "wind_speed": 2.66,
      "wind_deg": 1,
      "wind_gust": 4.9,
      "weather": [
        {
          "id": 500,
          "main": "Rain",
          "description": "light rain",
          "icon": "10d"
        }
      ],
      "clouds": 100,
      "pop": 0.35,
      "rain": 0.73,
      "uvi": 4.11
    }
  ]
kenilpatel121200@weatherapi-cluster-m:~$
```

Figure 7: Displaying Filtered Data.

Saved the filtered data i.e. fall_weather.json in local computer.



```
{
  "lat": 44.6462,
  "lon": -63.5736,
  "timezone": "America/Halifax",
  "timezone_offset": -10800,
  "daily": [
    {
      "dt": 1655827200,
      "sunrise": 1655800138,
      "sunset": 1655856188,
      "moonrise": 1655786400,
      "moonset": 1655829840,
      "moon_phase": 0.75,
      "temp": {
        "day": 12.95,
        "min": 11.01,
        "max": 14.56,
        "night": 12.65,
        "eve": 14.12,
        "morn": 11.99
      },
      "feels_like": {
        "day": 12.68,
        "night": 12.22,
        "eve": 13.68,
        "morn": 11.75
      },
      "pressure": 1021,
      "humidity": 91,
      "dew_point": 11.33,
      "wind_speed": 2.66,
      "wind_deg": 1,
      "wind_gust": 4.9,
      "weather": [
        {
          "id": 500,
          "main": "Rain",
          "description": "light rain",
          "icon": "10d"
        }
      ],
      "clouds": 100,
      "pop": 0.35,
      "rain": 0.73,
      "uvi": 4.11
    }
  ]
}
```

Figure 8: Filtered Data in the Local

References

[1] Mysql.com. [Online]. Available: https://dev.mysql.com/doc/workbench/en/wb-forward-engineering-live-server.html. [Accessed: 18-Oct-2023].

[2] “Google Cloud Platform,” Google.com. [Online]. Available: https://cloud.google.com/?hl=en. [Accessed: 18-Oct-2023].

[3] “Apache SparkTM - Unified Engine for large-scale data analytics,” Apache.org. [Online]. Available: https://spark.apache.org/. [Accessed: 04-Nov-2023].

[4] “Python,” Python.org. [Online]. Available: https://www.python.org/. [Accessed: 04-Nov-2023].