VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF ELECTRONICS ENGINEERING

TASK-2

Course Name : IoT Domain Analyst Lab

Name : Venkata Siddharth Dhara

Registration No : 19BEC0611

Lab Slot : L11 + L12

Lab Faculty Name : Prakash R

Aim:

Assuming a Smart Agriculture System, where there are three Soil Moisture sensors, two Humidity sensors and three motor controls.

Conditions:

- Simulation is to be done for sensor outputs using python and update every 5 seconds.
- 5 seconds are to be assumed as 1 hour as per Simulation.
- The min value is to be taken as 0% and max as 100%. The humidity level is also 0 to 100%.
- A day's data is to be collected and decisions are to be taken using Pandas (Python, Jupyter Notebook).
- The simulation is to be run for 3 days i.e. 72 hours, and soil moisture levels are to be checked in the morning every day.
- If the average of soil moisture levels is less than 50%< motors are to be switched on.
- Soil moisture levels(3 sensor data) and Humidity levels (2 sensor data) is to be plotted on the ThingSpeak,
- Mean of soil moisture levels and humidity is to be determined and correlation is to found out between soil moisture and humidity.
- The demonstration should cover both the cases –when the soil moisture levels are less than 50% and more than 50 %.

Software used/Tools Required:

- **IDLE – Python Compiler** (for sending data initially to ThingSpeak cloud)
  Python's Integrated Development and Learning Environment is known as IDLE. It makes it simple for programmers to develop Python code. IDLE, like Python Shell, may be used to run a single command as well as develop, edit, and run Python scripts.

- **Jupyter Notebook** (Anaconda) – for data processing
  The Jupyter Notebook is an open-source web tool that lets data scientists create and share documents with live code, equations, computational

output, visualisations, and other multimedia elements, as well as explanatory text.

- **ThingSpeak** (Visualization of data)
  ThingSpeak is a cloud-based IoT analytics tool that lets you gather, visualise, and analyse real-time data streams.

Installation and Simulation Procedure:

1. Firstly, Open IDLE python script and write the code with initializes all the given parameters, and copy the Write API key from the ThingSpeak channel we will be making in the following steps.
2. The code is to be run , and randomized data is formed, and this data is uploaded to ThingSpeak Channel.
3. For the creation of ThingSpeak channel, we need to go to https://thingspeak.com/, and create a channel with the entire fields required, and copy the Write API key as mentioned in the previous step, and when the code is run, we can see data being uploaded to ThingSpeak.
4. Now, Install **Anacondas** from the website link given here: https://docs.anaconda.com/anaconda/install/windows/
5. Finish the installation procedure, by agreeing on the terms and conditions and setting Anaconda a path. Anaconda by itself has a lot of libraries and Python. Therefore, it will take some time to install.
6. After installing, go to START and search for **Anaconda** Prompt and once it opens, type **import pandas.**
7. This imports the **pandas** library if not installed already.
8. Now, go to START again and Open Jupyter notebook.
9. A prompt dialog box appears and redirects to Jupyter Notebook.
10. Click on new, then New Notebook (Python Script 3.7) and it redirects to a new notebook.
11. Now the data analysis can be performed in this notebook.
12. Data is collected, and read as a dataframe and decisions are thus taken using Pandas.

Code and Simulation Snippets:

Python IDLE Script Snippet –

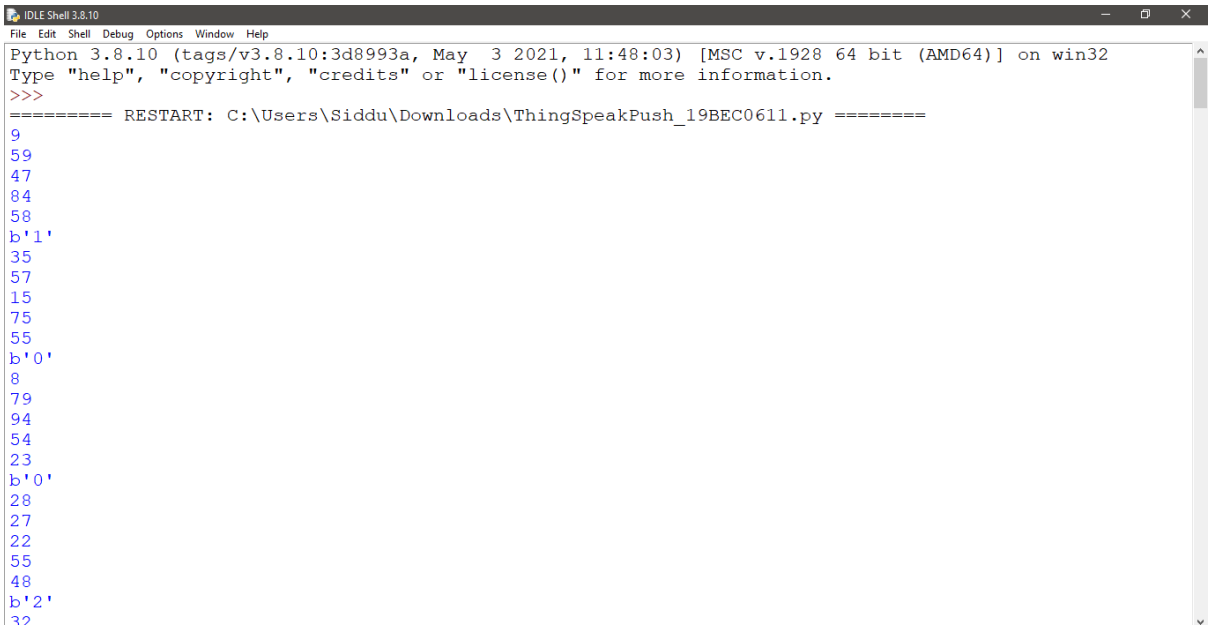
Python Code:

```
import os
import time
import sys
import random
import urllib
import urllib.request
n=72
while (n!=0):
        senva1= random.randrange(0,50)
        senva2= random.randrange(0,100)
        senva3= random.randrange(0,100)
        senva4= random.randrange(0,100)
```

```
        senva5= random.randrange(0,100)
        print(senva1)
        print(senva2)
        print(senva3)
        print(senva4)
        print(senva5)

        b=urllib.request.urlopen('https://api.thingspeak.com/update?api_key=8B
HAVW82IGR00XAV&field1=' + str(senva1)+ '&field2=' + str(senva2)+
'&field3=' + str(senva3)+ '&field4=' + str(senva4)+ '&field5=' + str(senva5))
        time.sleep(5)
        print(b.read())
        n=n-1
print('end')
```

Python Output Snippet (Generation of randomized data for analysis)



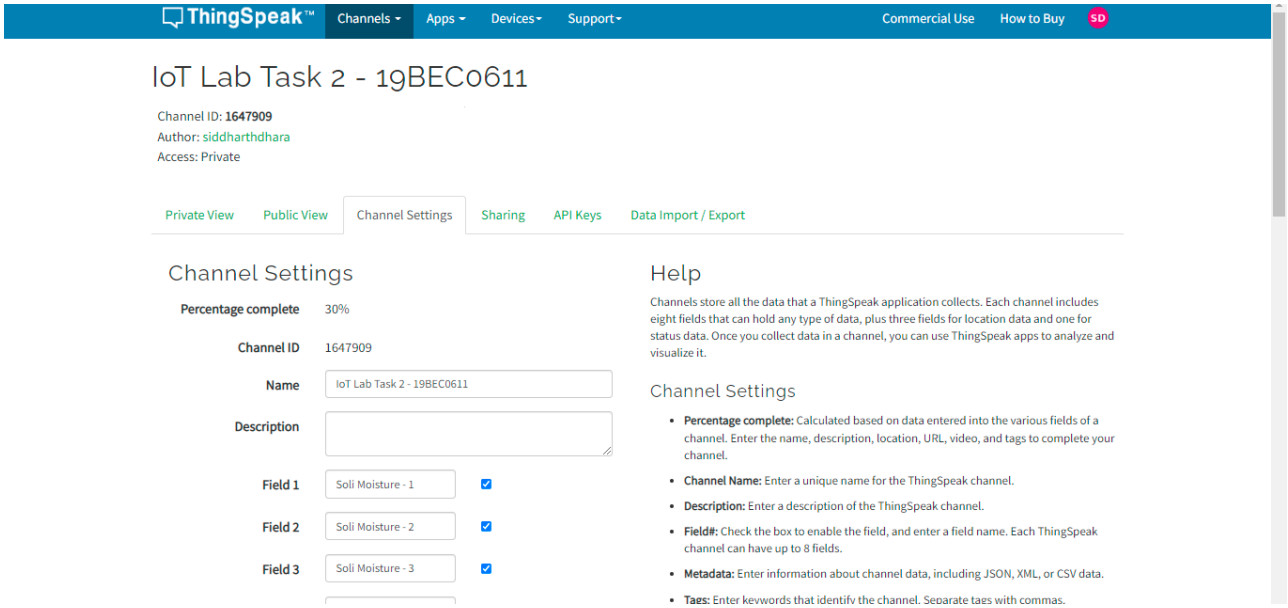
Python Code Explanation-

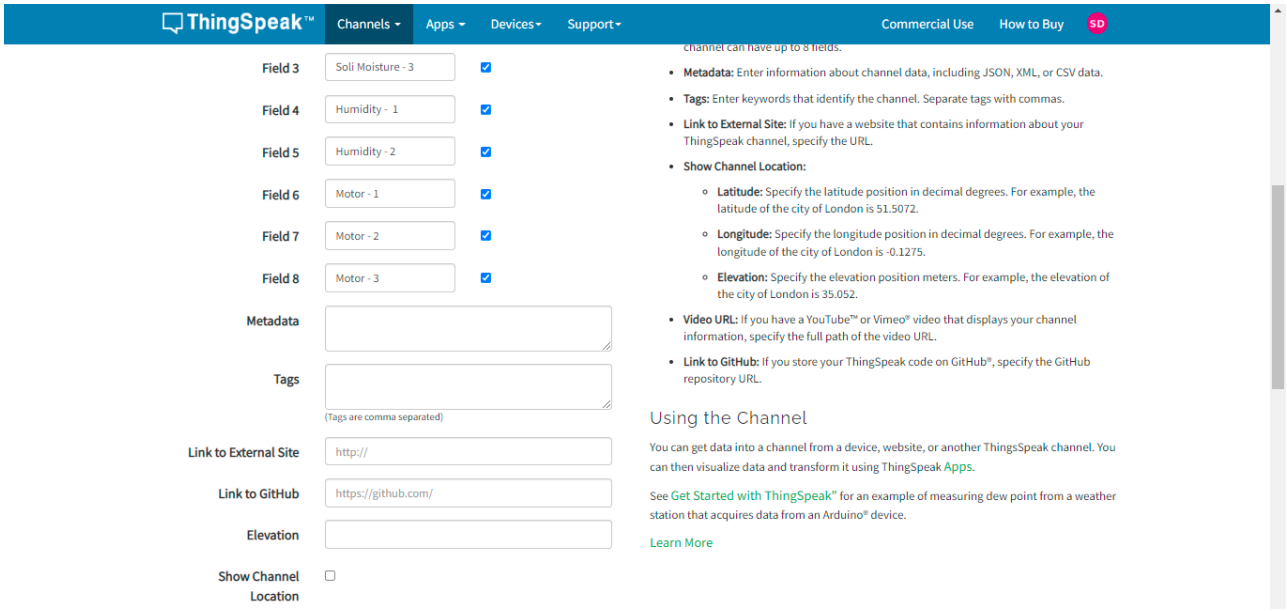
- Firstly, import the libraries necessary for this analysis.
- We are then initializing the variables necessary such as n=72 (signifies 72 hours – 3 days)
- In addition, we run a while loop when n is not zero, which means that randomized data will run from when n is not 0, i.e. from 1 until 72 – so 24 sets of sensor data is created.
- This set of data created is given a range between 0 to 100, and printed.
- A command is used tor specifying the Write API Key of the ThingSpeak tool, so that this data can be analysed graphically on a channel in the cloud tool.
- These 24 sets of data contain the sensor data of three soil moisture sensors and two humidity sensors as required.

ThingSpeak Cloud Platform

- A channel is created with 8 fields symbolizing the three soil moisture sensors, 2 humidity sensors.
- 3 fields are also initialized for the Motor control (which will be decided by Pandas (using Jupyter Notebook, by finding the average/mean of the data)

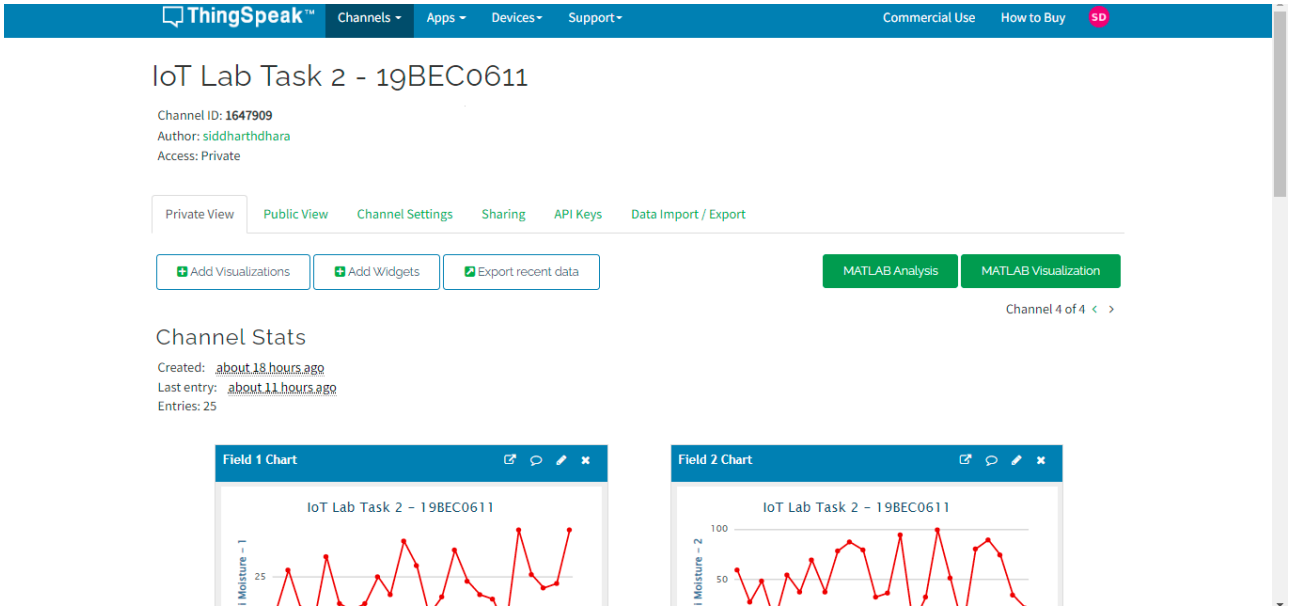
ThingSpeak Channel Creation



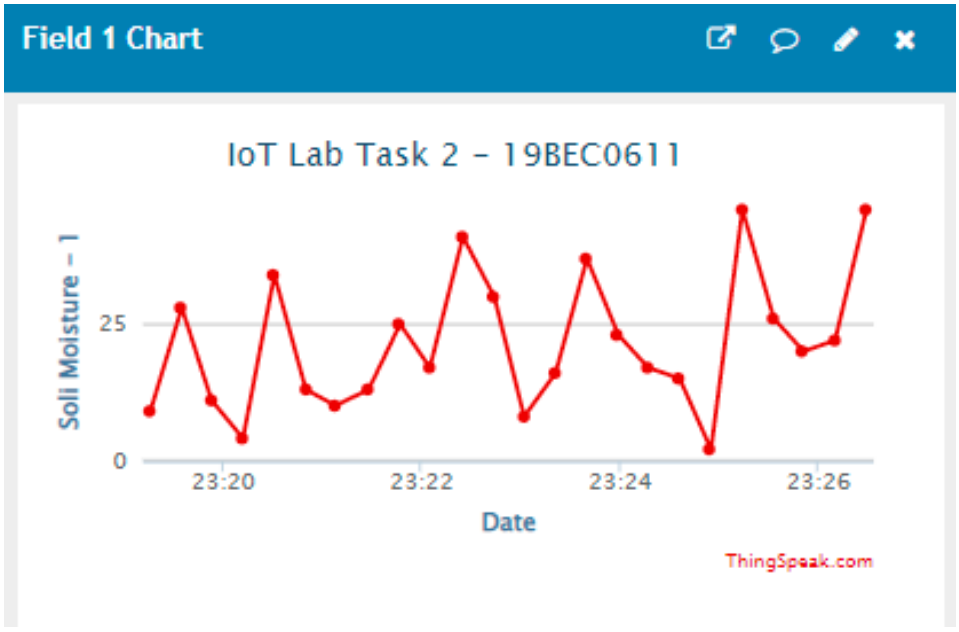


This channel is then saved and thus created. Now the below snippets will demonstrate the data sent by the Python code which is mentioned above.

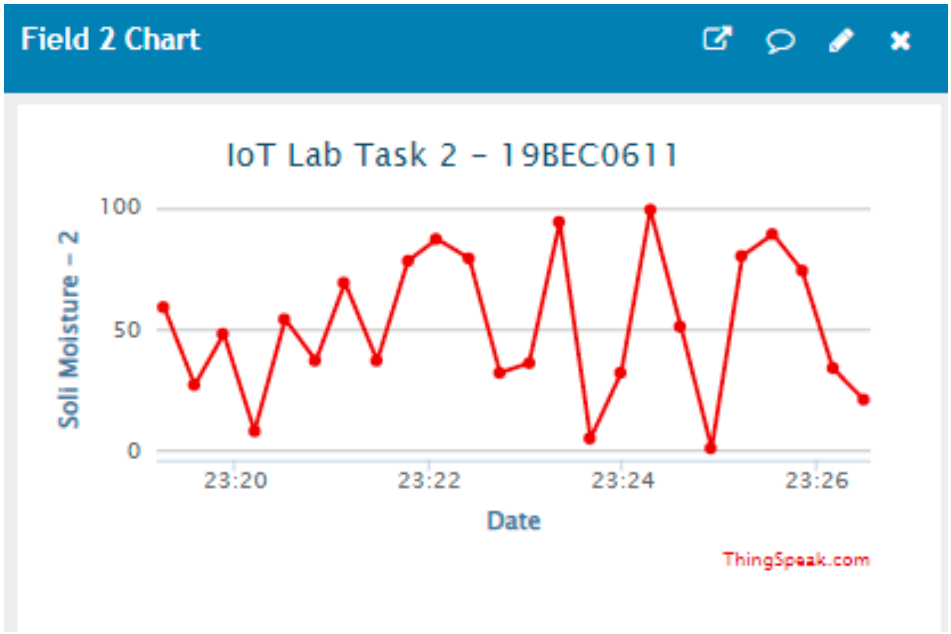
ThingSpeak Channel Snippet



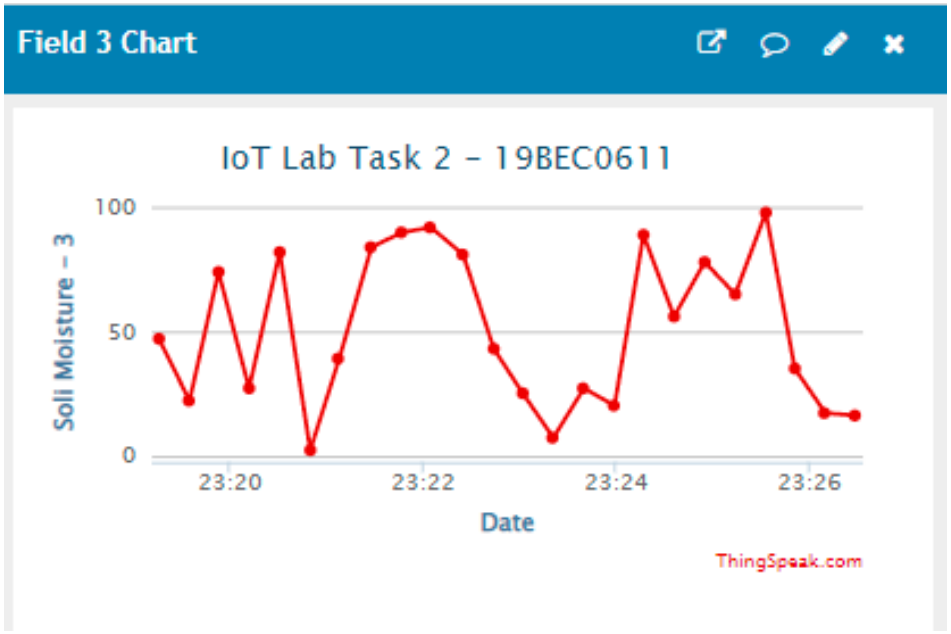
Soil Moisture Sensor – 1



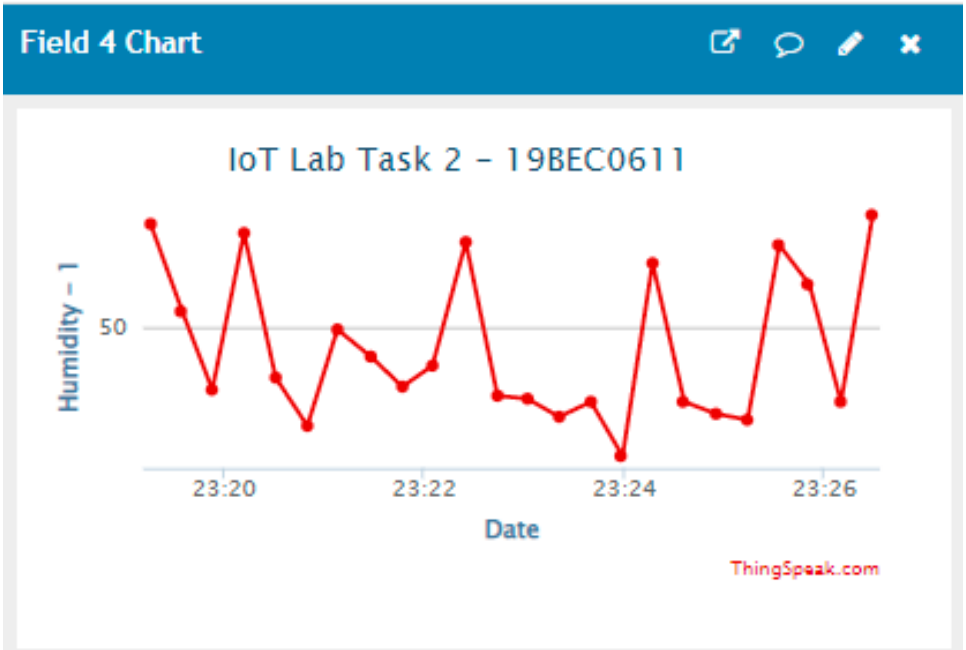
Soil Moisture Sensor – 2



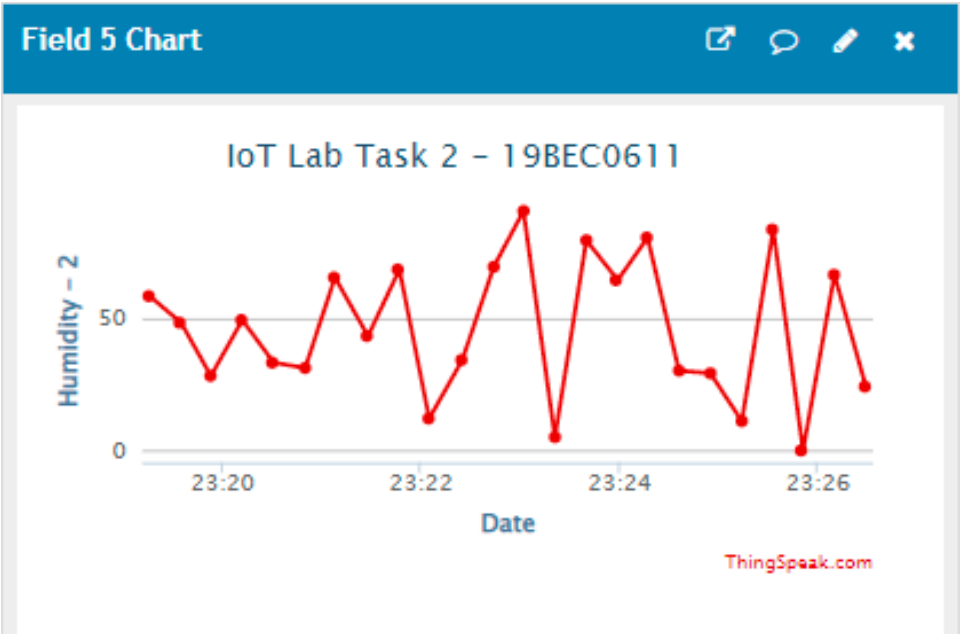
Soil Moisture Sensor – 3



Humidity Sensor -1



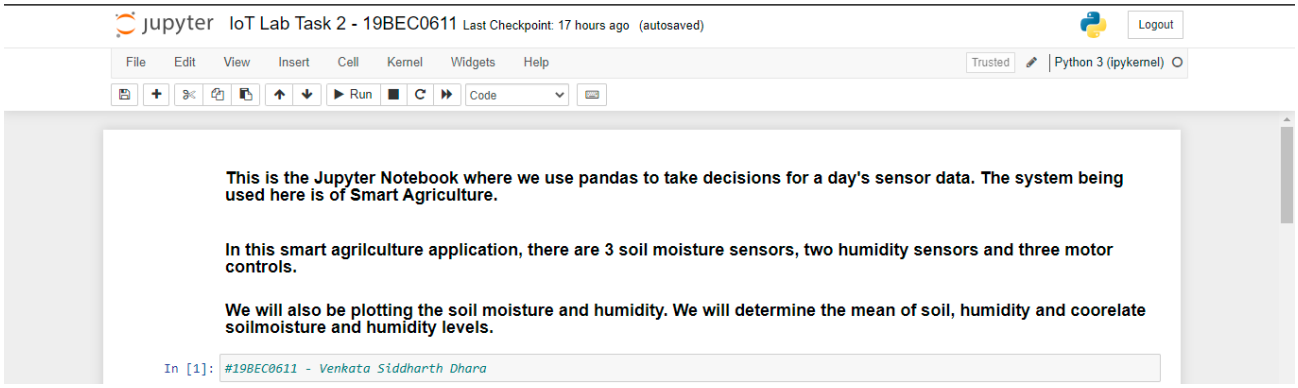
Humidity Sensor -2



Randomized Data created by Python Compiler

- This data will now be downloaded as a .csv file from the ThingSpeak channel, and read as a dataframe.
- This data is now analysed using Pandas (Jupyter Notebook).

Jupyter Notebook Snippet:



Code and Data Analysis Snippets –

Import the necessary libraries for this operation

```
In [15]: import pandas as pd
         import matplotlib as plt
         import os
         import time
         import sys
         import random
         import urllib
         import urllib.request
```

Reading the .csv as a dataframe

In [16]:
```
df=pd.read_csv("C:\\Users\\Siddu\\Downloads\\IoT_Lab_DA2\TS_feed_19BEC0611.csv")
```

Printing the dataframe

In [17]:
```
print(df)
```

```
                 created_at  entry_id  field1  field2  field3  field4  field5  \
0   2022-02-05 17:49:16 UTC         1       9      59      47      84      58
1   2022-02-05 17:49:35 UTC         2      28      27      22      55      48
2   2022-02-05 17:49:53 UTC         3      11      48      74      29      28
3   2022-02-05 17:50:12 UTC         4       4       8      27      81      49
4   2022-02-05 17:50:31 UTC         5      34      54      82      33      33
5   2022-02-05 17:50:50 UTC         6      13      37       2      17      31
6   2022-02-05 17:51:08 UTC         7      10      69      39      49      65
7   2022-02-05 17:51:28 UTC         8      13      37      84      40      43
8   2022-02-05 17:51:47 UTC         9      25      78      90      30      68
9   2022-02-05 17:52:05 UTC        10      17      87      92      37      12
10  2022-02-05 17:52:25 UTC        11      41      79      81      78      34
11  2022-02-05 17:52:44 UTC        12      30      32      43      27      69
12  2022-02-05 17:53:02 UTC        13       8      36      25      26      90
13  2022-02-05 17:53:21 UTC        14      16      94       7      20       5
14  2022-02-05 17:53:40 UTC        15      37       5      27      25      79
15  2022-02-05 17:53:59 UTC        16      23      32      20       7      64
16  2022-02-05 17:54:17 UTC        17      17      99      89      71      80
17  2022-02-05 17:54:36 UTC        18      15      51      56      25      30
18  2022-02-05 17:54:55 UTC        19       2       1      78      21      29
19  2022-02-05 17:55:14 UTC        20      46      80      65      19      11
20  2022-02-05 17:55:33 UTC        21      26      89      98      77      83
21  2022-02-05 17:55:51 UTC        22      20      74      35      64       0
22  2022-02-05 17:56:10 UTC        23      22      34      17      25      66
23  2022-02-05 17:56:29 UTC        24      46      21      16      87      24

    field6  field7  field8
0      NaN     NaN     NaN
1      NaN     NaN     NaN
2      NaN     NaN     NaN
3      NaN     NaN     NaN
4      NaN     NaN     NaN
5      NaN     NaN     NaN
6      NaN     NaN     NaN
7      NaN     NaN     NaN
8      NaN     NaN     NaN
9      NaN     NaN     NaN
10     NaN     NaN     NaN
11     NaN     NaN     NaN
12     NaN     NaN     NaN
13     NaN     NaN     NaN
14     NaN     NaN     NaN
15     NaN     NaN     NaN
16     NaN     NaN     NaN
17     NaN     NaN     NaN
18     NaN     NaN     NaN
19     NaN     NaN     NaN
20     NaN     NaN     NaN
21     NaN     NaN     NaN
22     NaN     NaN     NaN
23     NaN     NaN     NaN
```

Finding the mean of the Soil Moisture Sensor Data

In [18]:
```
df_mean1 = df["field1"].mean()
df_mean2 = df["field2"].mean()
df_mean3 = df["field3"].mean()
```

Displaying/Printing the mean of the Soil Moisture Sensor Data

In [20]:
```
print(df_mean1)
print(df_mean2)
print(df_mean3)
```

```
21.375
51.291666666666664
50.666666666666664
```

- Clearly, two sensors have the average soil moisture level above 50% and one of them has a soil moisture level below 50%.
- So Motor 1 has to be on, and motors 2 and 3 have to be off (according to the condition).
- For this, in the next step, I have used an if-else loop to decide which motor has to be on and which ones have to be off.

Deciding whether the motors have to be on or not, using the soil moisture mean data for the three sensors

In [19]:
```
if(df_mean1>=50):
    senva1=0
    print("Motor 1 is off")
else:
    senva1=1
    print("Motor 1 is on")
if(df_mean2>=50):
    senva2=0
    print("Motor 2 is off")
else:
    senva2=0
    print("Motor 2 is on")
if(df_mean3>=50):
    senva3=0
    print("Motor 3 is off")
else:
    senva3=1
    print("Motor 3 is on")
```

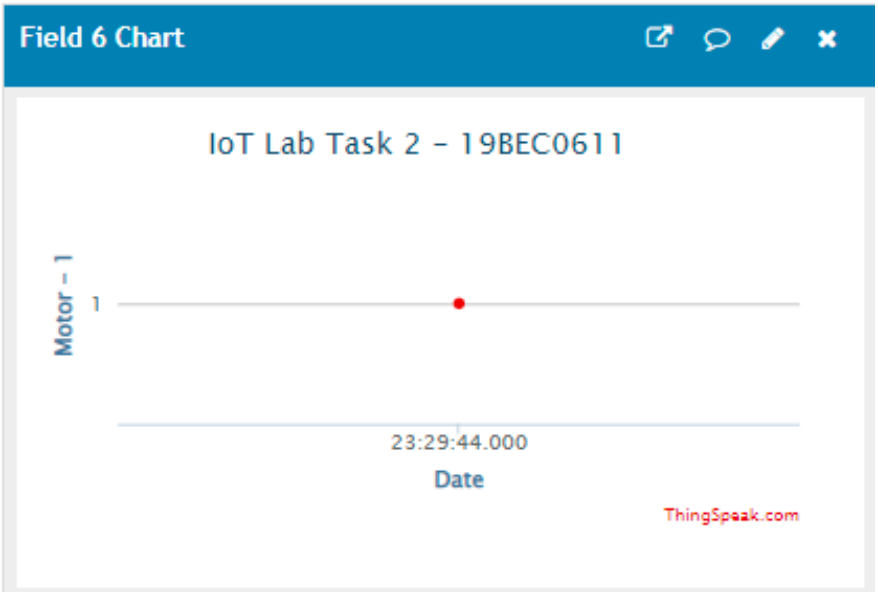
```
Motor 1 is on
Motor 2 is off
Motor 3 is off
```

Sending the motor data to ThingSpeak Channel (Fields 6,7,8 for the motor data – one each for the three Soil Moisture Sensors)

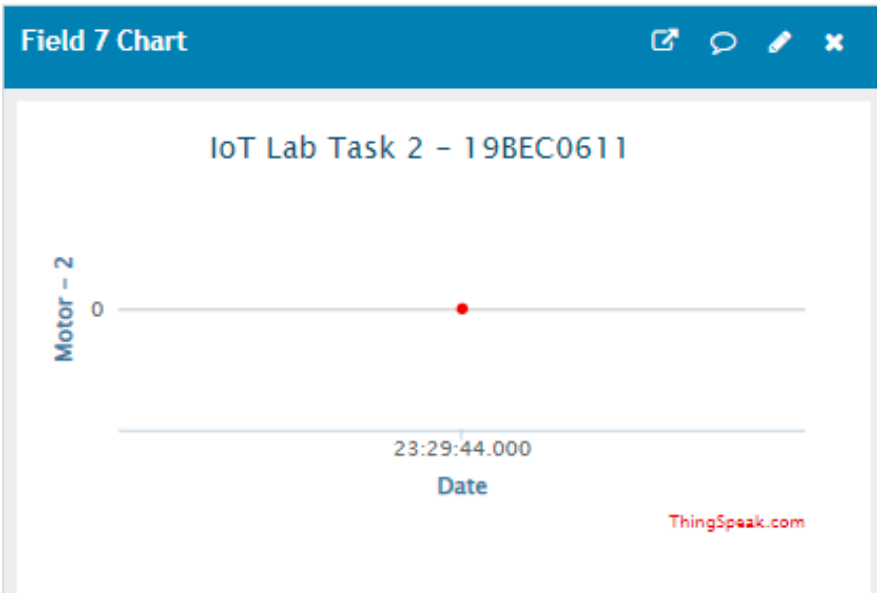
```
In [21]: b=urllib.request.urlopen('https://api.thingspeak.com/update?api_key=8BHAVW82IGR00XAV&field6=' + '1'+'&field7=' + '0'+'&field8=' +
```

ThingSpeak Snippets for Motor Data

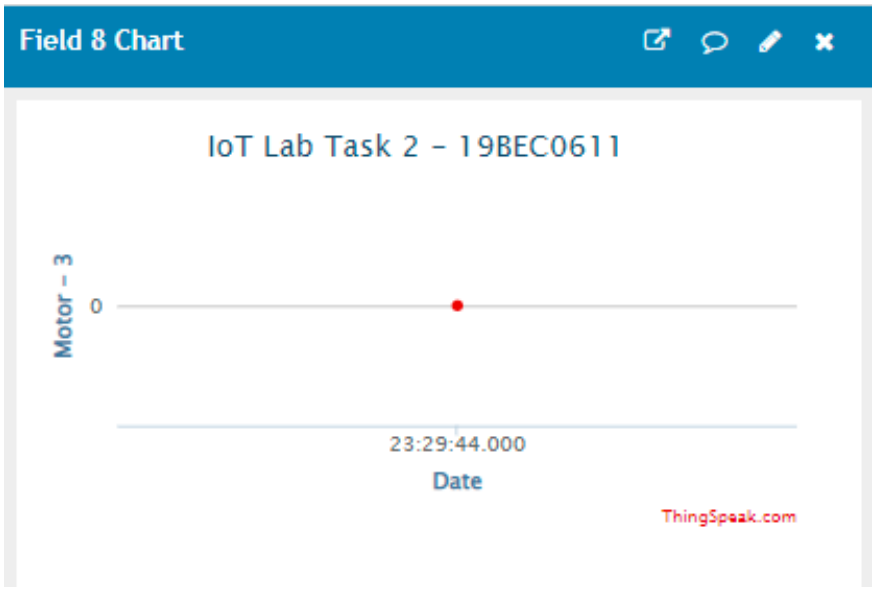
Motor -1 (ON)



Motor -2 (OFF)



Motor -3 (OFF)



Finding the correlation between the Soil Moisture and Humidity sensors

In [22]:
```
# correlation between soil moisture sensor-1 and humidity sensor-1
print(df['field1'].corr(df['field4']))

# correlation between soil moisture sensor-2 and humidity sensor-1
print(df['field2'].corr(df['field4']))

# correlation between soil moisture sensor-3 and humidity sensor-1
print(df['field3'].corr(df['field4']))

# correlation between soil moisture sensor-1 and humidity sensor-2
print(df['field1'].corr(df['field5']))

# correlation between soil moisture sensor-2 and humidity sensor-2
print(df['field2'].corr(df['field5']))

# correlation between soil moisture sensor-3 and humidity sensor-2
print(df['field3'].corr(df['field5']))
```

```
0.09265730921169035
0.17398303184253286
0.15955672161916118
-0.09663569090738247
-0.17060999277168362
0.02504618304580059
```

Verification:

For verification, I also uploaded this on my Github.

The link for the DA to verify the document - https://github.com/siddharthdhara17/IoTDomainAnalyst_Lab2

Conclusion

Thus, I have successfully performed the analysis for the set of sensor data generated by Python. I have performed different operations on the data. In addition, with the help of this analysis I could find out a lot of information from the data.

Depending on the conditions given, analysis has been done, and whenever the soil moisture levels are less than 50%, that respective motor will switch on; this analysis has been done using ThingSpeak, Pandas and Python.

Result

Thus, the output is generated and verified.

Thank You!