

**Name: Vansh Rahate**

**SUID: 547161826**

## IOT- Assignment 3

### STEPS:

To develop the cloud-based IoT system, I used the ThingSpeak platform with Python to simulate virtual environmental stations. Here's a breakdown of the steps:

#### ThingSpeak Channel Setup

- A ThingSpeak channel contained three data fields for Temperature, Humidity and CO2 measurements.
- The system required both the Channel ID and Write API key for MQTT or HTTP data push functionality.

#### Virtual Sensor Creation

- Developed a Python script to simulate random sensor data (Temperature: -50 to 50°C, Humidity: 0–100%, CO2: 300–2000 ppm).
- The paho-mqtt library proved initially useful for ThingSpeak data publication yet I had to switch to ThingSpeak's HTTP API with requests due to Colab restrictions.

#### Data Collection and Visualization

- Sent multiple data points using the API.
- Retrieved sensor values via ThingSpeak's REST API.
- Visualized the data using matplotlib in Colab.

#### Output & GitHub Submission

- Captured screenshots of channel output and plots.
- Uploaded code and notebook to GitHub with a structured README.

### Screenshots

ThingSpeak Channels Apps Devices Support Commercial Use How to Buy

Virtual Station  
Channel ID: 2852082  
Author: mwa000038485878  
Access: Private

Private View Public View Channel Settings Sharing API Keys Data Import / Export

Channel Settings  
Percentage Complete: 30%

Channel ID: 2852082

Name: Virtual Station

Description:

Field 1: Temperature ☒

Field 2: Humidity ☒

Field 3: CO2 ☒

Field 4:

Field 5:

Field 6:

Field 7:

Field 8:

Metadata:

Help  
Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once you collect data in a channel, you can use ThingSpeak apps to analyze and visualize it.

Channel Settings

- **Percentage complete:** Calculated based on data entered into the various fields of a channel. Enter the name, description, location, URL, video, and tags to complete your channel.
- **Channel Name:** Enter a unique name for the ThingSpeak channel.
- **Description:** Enter a description of the ThingSpeak channel.
- **Fields:** Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- **Metadata:** Enter information about channel data, including JSON, XML, or CSV data.
- **Tags:** Enter keywords that identify the channel. Separate tags with commas.
- **Link to External Site:** If you have a website that contains information about your ThingSpeak channel, specify the URL.
- **Show Channel Location:**
  - **Latitude:** Specify the latitude position in decimal degrees. For example, the latitude of the city of London is 51.5072.
  - **Longitude:** Specify the longitude position in decimal degrees. For example, the longitude of the city of London is 0.1275.
  - **Elevation:** Specify the elevation position meters. For example, the elevation of the city of London is 30.05.
- **Video URL:** If you have a YouTube® or Vimeo® video that displays your channel information, specify the full path of the video URL.
- **Link to GitHub:** If you store your ThingSpeak code on GitHub®, specify the GitHub repository URL.

```
IOT_Assign3.ipynb
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text

Collecting paho-mqtt
Downloading paho-mqtt-2.1.0-py3-none-any.whl.metadata (23 kB)
Downloading paho-mqtt-2.1.0-py3-none-any.whl (67 kB)
Installing collected packages: paho-mqtt
Successfully installed paho-mqtt-2.1.0

import random
import requests
import time

# Your Write API Key from ThingSpeak
write_api_key = "U6ASC4VGEC1DA3PF"

def generate_sensor_data():
    temperature = round(random.uniform(-50, 50), 2)
    humidity = round(random.uniform(0, 100), 2)
    co2 = round(random.uniform(300, 2000), 2)
    return temperature, humidity, co2

# Send a few sample entries (not infinite loop to avoid Colab timeout)
for i in range(5):
    temp, hum, co2 = generate_sensor_data()
    url = f"https://api.thingspeak.com/update?api_key={write_api_key}&field1={temp}&field2={hum}&field3={co2}"
    response = requests.get(url)
    if response.status_code == 200:
        print(f"[Sent] Temp: {temp} °C | Humidity: {hum}% | CO2: {co2} ppm | Entry ID: {response.text}")
    else:
        print("Failed to send data.")
    time.sleep(15)

[Sent] Temp: 2.27 °C | Humidity: 51.05% | CO2: 1836.66 ppm | Entry ID: 21
[Sent] Temp: 14.69 °C | Humidity: 19.7% | CO2: 1942.8 ppm | Entry ID: 22
[Sent] Temp: -9.38 °C | Humidity: 69.18% | CO2: 559.55 ppm | Entry ID: 23
[Sent] Temp: 39.49 °C | Humidity: 66.0% | CO2: 1876.85 ppm | Entry ID: 24
[Sent] Temp: -40.96 °C | Humidity: 15.72% | CO2: 1225.65 ppm | Entry ID: 25

1m 15s completed at 2:26 PM
```

ThingSpeak

Channels Apps Devices Support

Commercial Use How to Buy

Virtual Station

Channel ID: 2892082  
Author: mwa000036845878  
Access: Private

Private View Public View Channel Settings Sharing API Keys Data Import / Export

Write API Key

Key U6ASC4VGEC1DA3PF

Generate New Write API Key

Read API Keys

Key OU3AB0V98CIY5NGX

Note

Save Note Delete API Key

Add New Read API Key

Help

API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.

API Keys Settings

- Write API Key:** Use this key to write data to a channel. If you feel your key has been compromised, click **Generate New Write API Key**.
- Read API Keys:** Use this key to allow other people to view your private channel feeds and charts. Click **Generate New Read API Key** to generate an additional read key for the channel.
- Note:** Use this field to enter information about channel read keys. For example, add notes to keep track of users with access to your channel.

API Requests

Write a Channel Feed

GET https://api.thingspeak.com/update?api\_key=U6ASC4VGEC1DA3PF&field1=0

Read a Channel Feed

GET https://api.thingspeak.com/channels/2892082/feeds.json?api\_key=OU3AB0V98CIY5NGX

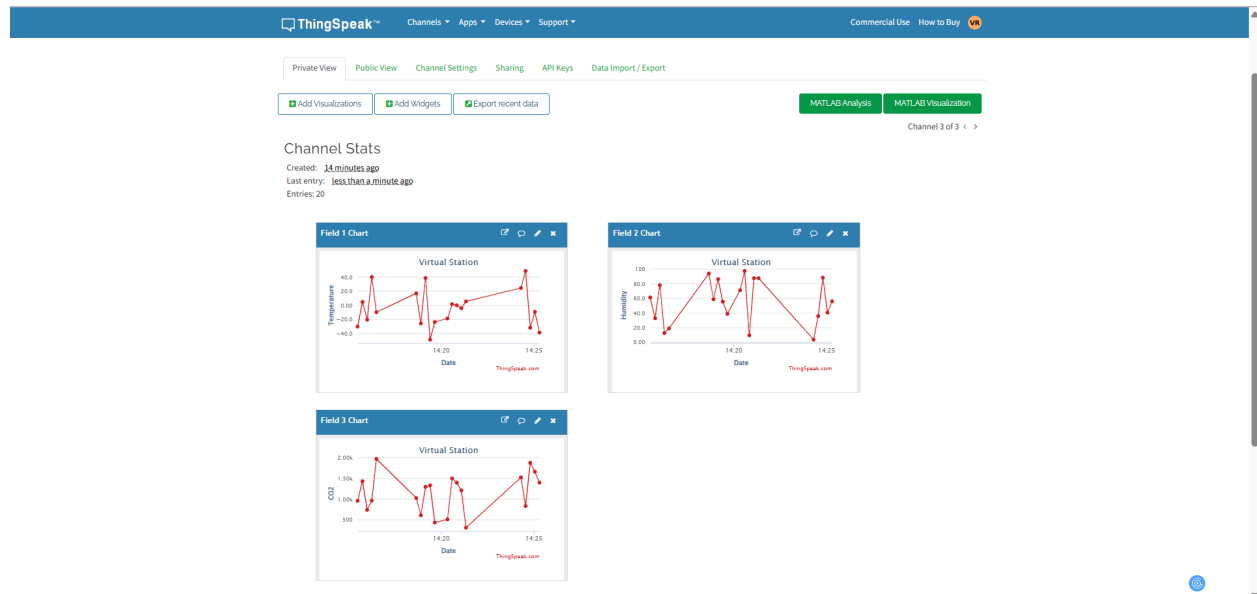
Read a Channel Field

GET https://api.thingspeak.com/channels/2892082/fields/1.json?api\_key=OU3AB0V98CIY5NGX

Read Channel Status Updates

GET https://api.thingspeak.com/channels/2892082/status.json?api\_key=OU3AB0V98CIY5NGX

Learn More



### Reflection:

This assignment was a great hands-on experience that helped me better understand how IoT systems work with cloud platforms. I initially tried using MQTT with the paho-mqtt library, but ran into issues running it on Google Colab. After some trial and error, I switched to using HTTP requests with ThingSpeak's API, which worked much better in that environment.

One thing I learned is how important it is to adapt when things don't go as planned. Debugging API errors and figuring out how to structure the requests correctly helped me understand how cloud-based communication really works. Visualizing the data in Colab was also really satisfying, it made the whole system feel real.

Overall, I feel more confident working with IoT data, APIs, and cloud platforms now, and I enjoyed the process of solving real-world problems during this project.

### Github:

[https://github.com/Vansh3117/IOT\\_Cloud\\_Assign3](https://github.com/Vansh3117/IOT_Cloud_Assign3)