RASPBERRY PI PERFORMANCE MONITORING ON THINGSPEAK CLOUD

Many a times we would need to monitor the performance of any server or device that we are working on. With the dawn of 'Internet of Things', things like such have become attainable. In this article one such application i.e. uploading Raspberry pi performance parameters onto ThingSpeak IoT platform is illustrated. This system sends Raspberry pi CPU temperature, memory available and disk usage to ThingSpeak platform and can be reviewed from anyplace via internet.

THINGS REQUIRED:

1.Raspberry pi (I used Raspbian buster OS)
2.Wi-Fi or LAN connectivity

WHAT IS THINGSPEAK?

ThingSpeak is a platform that provides services and facilitates in building IoT applications. It helps with real-time data collection from sensors, data visualization, plugins and apps for collaborating with web services, social network and many other APIs.

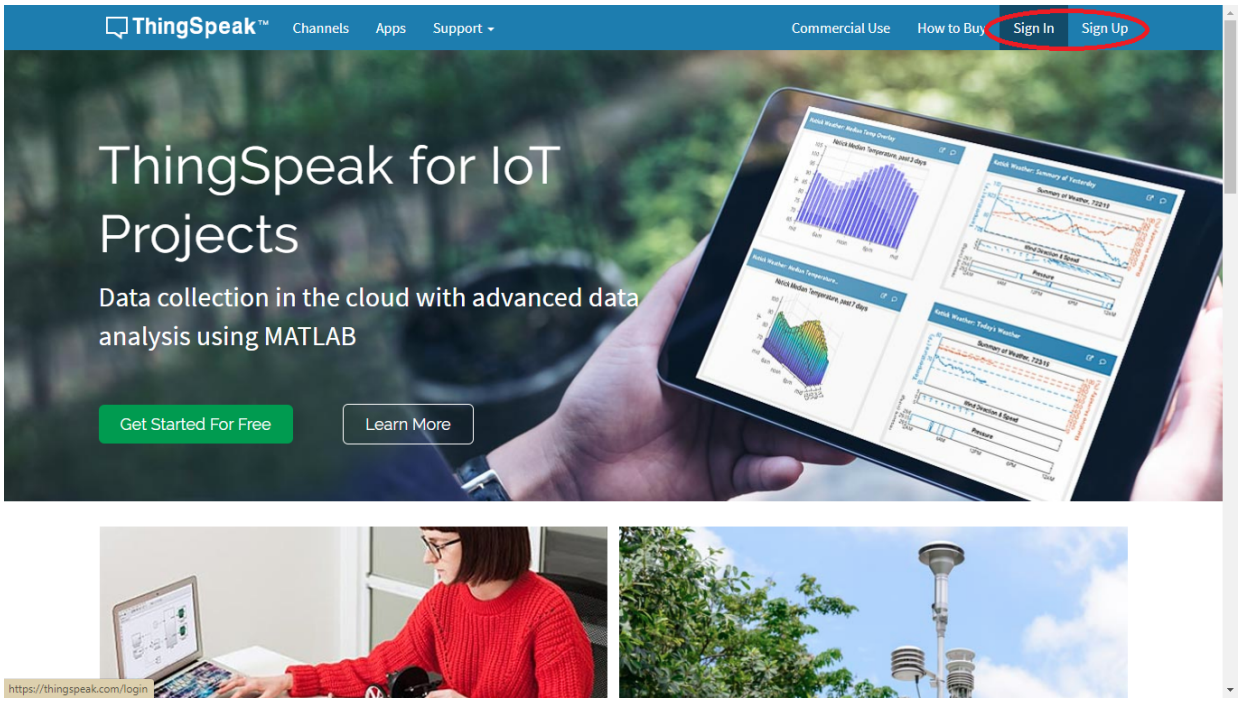
It consists of a channel that stores the data we desire to send to ThingSpeak. It has several attributes like: eight fields for collecting data from a sensor or any other device, an attribute named 'Metadata' that can store JSON, XML and CSV data. Links can also be provided to any external sites, YouTube videos or github that have information about the channel. There are three location fields that store the latitude, longitude and the elevation of the device being interfaced.

The dashboard shows all existing channels with channel name and useful options associated with it. Channel's private and public view can be monitored, channel settings can be modified from dashboard itself also API key and data sharing options are at hand.

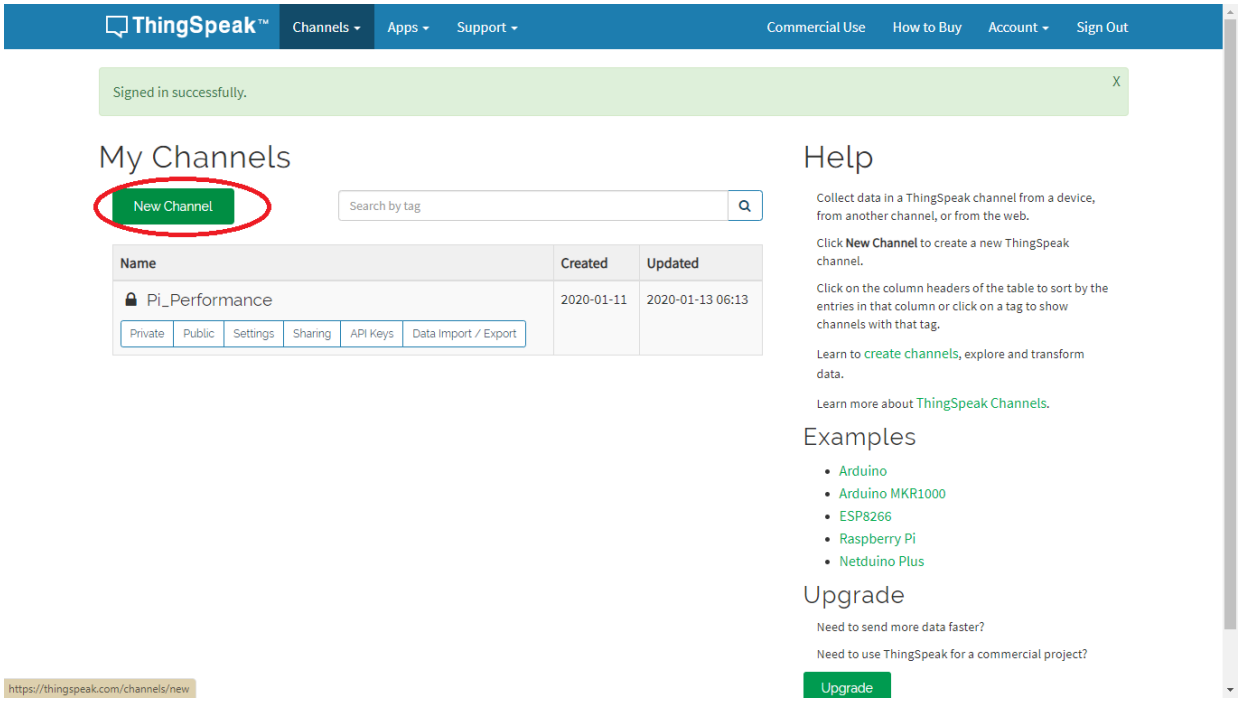
RASPBERRY PI PERFORMANCE MONITORING:

RASPBERRY PI PERFORMANCE MONITORING SYSTEM

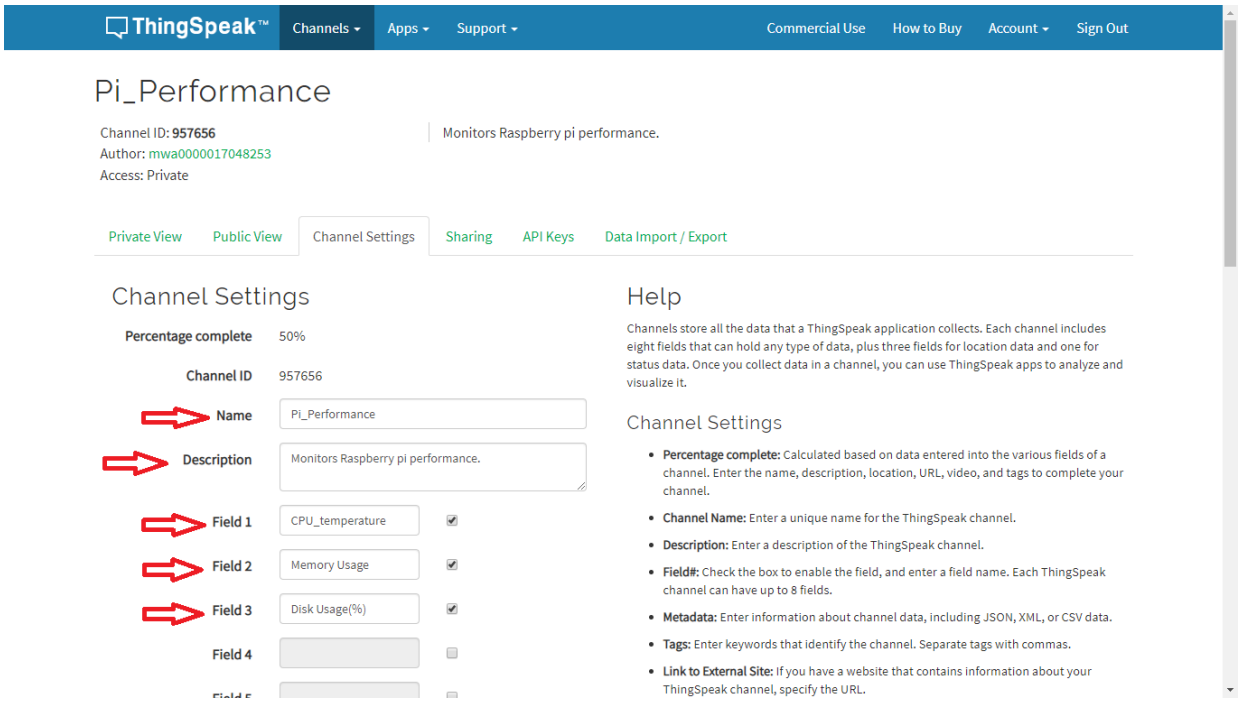
1. First off, sign up with ' https://thingspeak.com '. You can also sign in with your Mathworks account.



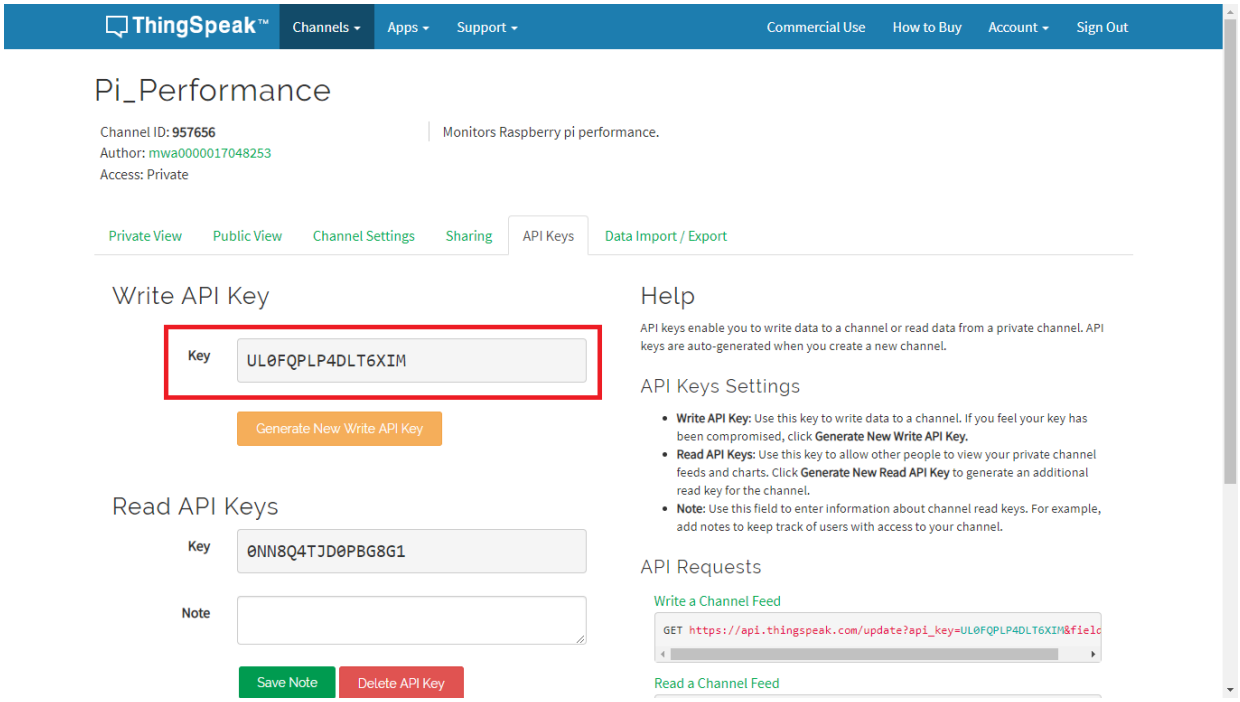
2. Head to New Channel to create a new channel.



3. Fillout the fields, giving channel a name, description and the parameters to be monitored.
My channel's name is 'Pi_performance' and it monitors 3 parameters: CPU temperature, available memory and disk usage.



4. Copy API key which has to get into the python code.



5. Python code:

This system is coded in Python 3 platform on Raspberry pi model B. Assuming you have already installed 'pip' in Python 3, let's dive in. The modules used come pre-installed with Python 3.

'http.client' defines classes which implement the client side of the HTTP and HTTPS protocols. It is normally used with the module 'urllib.request' to handle URLs that use HTTP and HTTPS.

'psutil'(python system and process utilities) is basically a cross-platform library that helps retrieving information on running

processes and system utilization (CPU, memory, disks, network, sensors) in Python. It will facilitate with the parameters we select to monitor pi's performance.
Code:

```
import http.client
import urllib
import time
import psutil
key = "YOUR API KEY FROM THINGSPEAK" # Put your API Key here
def perf_monitor():
    while True:
        temp = int(open('/sys/class/thermal/thermal_zone0/temp').read()) / 1e3 # Get Raspberry
Pi CPU temp
```

> *# This is how we get pi CPU temperature.*

```
        mem = psutil.virtual_memory()
        mem_available = mem.available # Get available CPU memory
```

> *# psutil.virtual_memory() returns metrics such as total memory, available memory, memory in use, buffers,etc. This code extracts available CPU memory.*

```
        x = psutil.disk_usage('/')
        disk_us = x.used # Get used disk space
```

> *# psutil.disk_usage(path) returns metrics about the partition which contains the given path. It returns total, used space in bytes and also the percentage usage. This code extracts used disk space in bytes.*

```
        params = urllib.parse.urlencode({'field1':
temp,'field2':mem_available,'field3':disk_us,'key':key })
```

> *# Returns 'field1=temp&field2=mem_available&field3=disk_us&key=YOUR API KEY FROM THINGSPEAK'*

```
        headers = {"Content-type": "application/x-www-form-urlencoded","Accept": "text/plain"}
        conn = http.client.HTTPConnection("api.thingspeak.com:80")
```

> *# This line represents one transaction with an HTTP server with 80 being the default port number.*

```
        try:
            conn.request("POST", "/update", params, headers)
            response = conn.getresponse()
            print (temp)
            print(mem_available)
            print(disk_pc)
            print (response.status, response.reason)
            data = response.read()
            conn.close()
        except:
            print( "connection failed")
            break
```

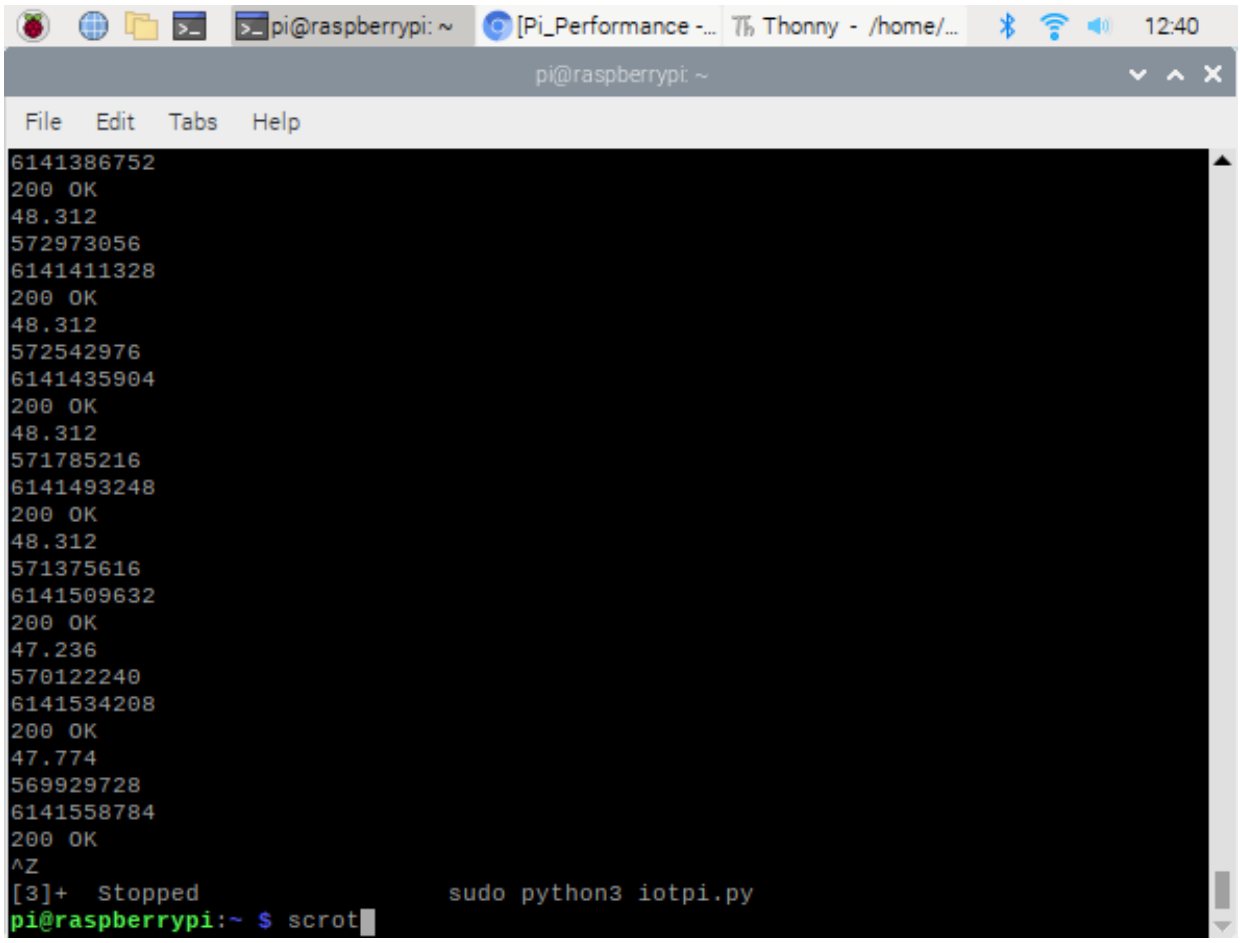
> *# system sleeps for 15 seconds here*

```
if __name__ == "__main__":
    while True:
        perf_monitor()
        time.sleep(16)
```

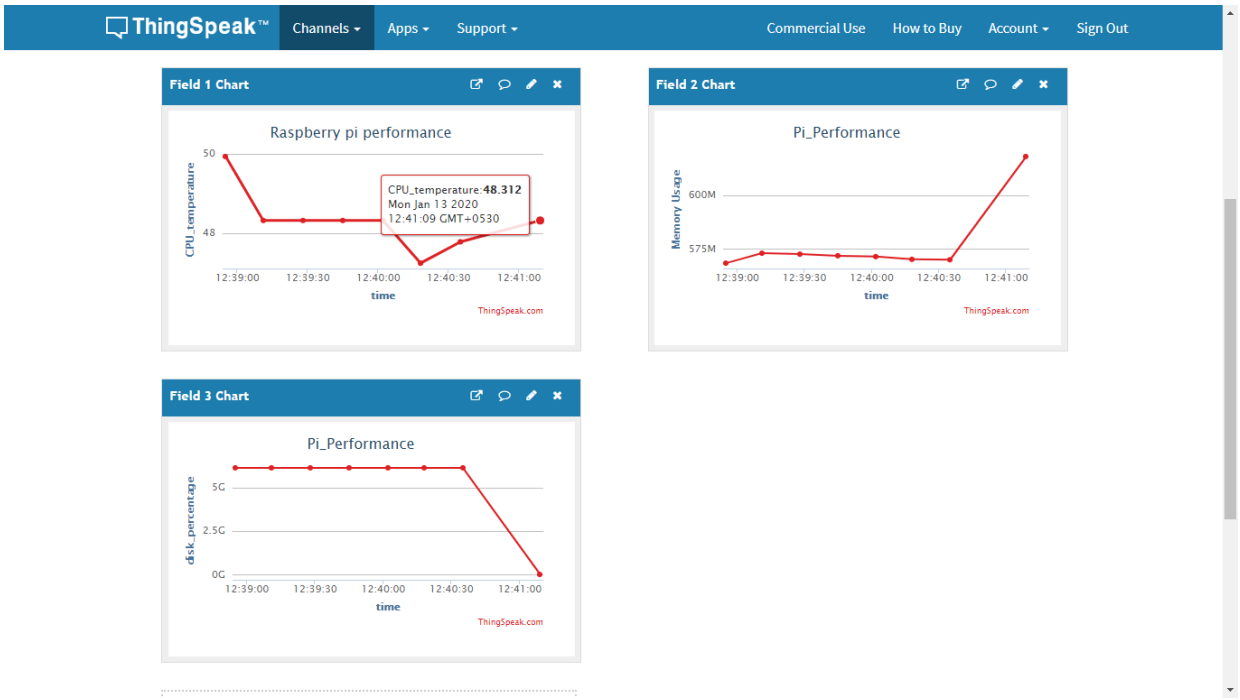
You can either run this code directly in IDLE by clicking on 'Run' or in terminal using the command:

```
sudo python3 filename.py
```

Output as viewed on terminal. It displays CPU temperature, available memory and disk used.



6. The output can be monitored from anywhere using ‘ThingSpeak platform’. The figure below shows the uploaded data.



7. The subsequent link shows my work and the above picture.
https://thingspeak.com/channels/957656

FURTHER POSSIBILITIES:

- This system can be interfaced with an LCD or OLED to locally display the performance parameters.
- Data can be visualized in many other ways using MATLAB.
- Data can be exported for additional analysis or processing.