



## CPE 4040: DATA COLLECTION AND ANALYSIS

### Lab 4: Raspberry Pi Weather Station Application on ThingSpeak

#### Learning Objectives:

1. Learn how to interface and read data from a temperature sensor.
2. Learn how to communicate with the popular ThingSpeak cloud platform.
3. Understand how to make HTTP client-server connection with Raspberry Pi.
4. Understand how to create a cloud server application with Raspberry Pi.
5. Learn how to control output of Raspberry Pi remotely from the cloud service.

#### Hardware Requirement:

1. Temperature sensor (either DHT-11, DHT-22, or DHT-20).
2. Breadboard, LED and resistors.

#### What is ThingSpeak?

ThingSpeak™ is an IoT analytics platform service that allows you to aggregate, visualize, and analyze live data streams in the cloud. You can send data to ThingSpeak from your devices, create instant visualization of live data, and send alerts to the subscribers.

About ThingSpeak: <https://thingspeak.com>

#### Lab Procedure:

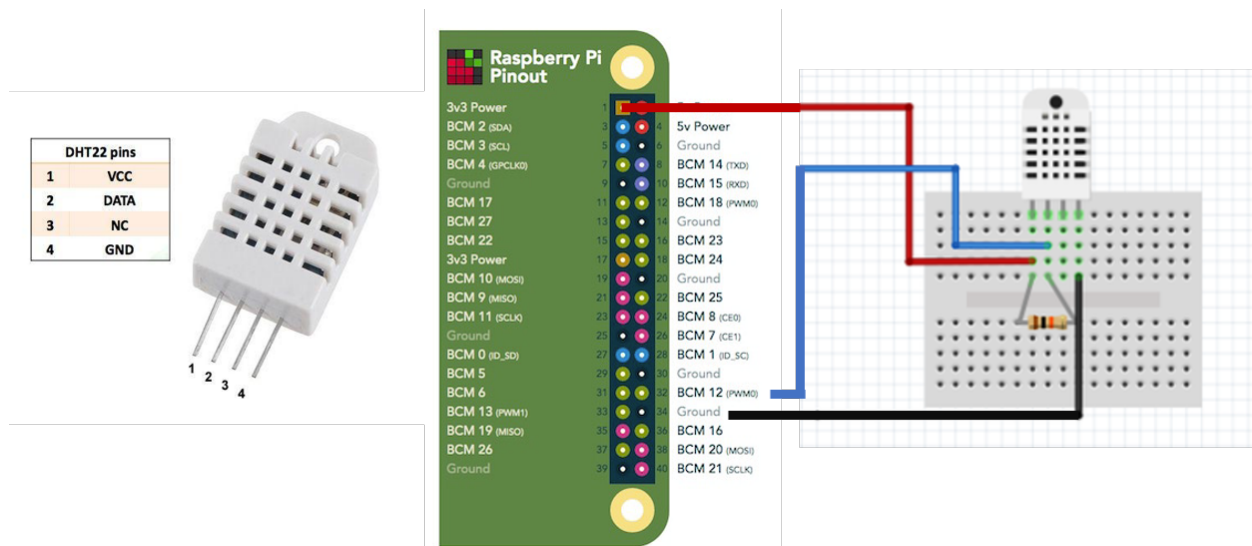
1. Plug the power adapter to Raspberry Pi and power up the system. Open Remote Desktop Connection (or SSH connection) on your laptop and connect to Raspberry Pi.

#### Section 1: Connect and Test the Temperature Sensor

2. Open a terminal window and type `pip3 list` to see which Python packages are installed in your device.
3. Install the following packages if they are not installed already: ***urllib3***, ***python-http-client***, ***requests***. **Question:** What are those three packages intended for? Do Google searches and give a brief description for each package.

Type `pip3 install <package name>` to install the packages

4. Connect the temperature sensor to Raspberry Pi with the below connection diagram. Note that there should be a resistor (around 10kΩ) placed between Vcc and DATA pins of the sensor.



5. Install the DHT library which contains the device driver for DHT-11 or DHT-22 :

```
pip3 install adafruit-circuitpython-dht
```

```
sudo apt-get install libgpiod2
```

**Question:** What is the library libgpiod2 for?

6. Create a file (*digitalOut.py*) with the python code below to get sensor readings from your DHT-11 or DHT-22. **Note:** If you have DHT-20, which is a newer version of the sensor with I2C interface, use the driver code, *dht20\_sensor\_driver.py*, that comes with this assignment.

```
import time
import board
import adafruit_dht

# Initialize the DHT device, with data pin connected to:
dhtDevice = adafruit_dht.DHT22(board.D12)

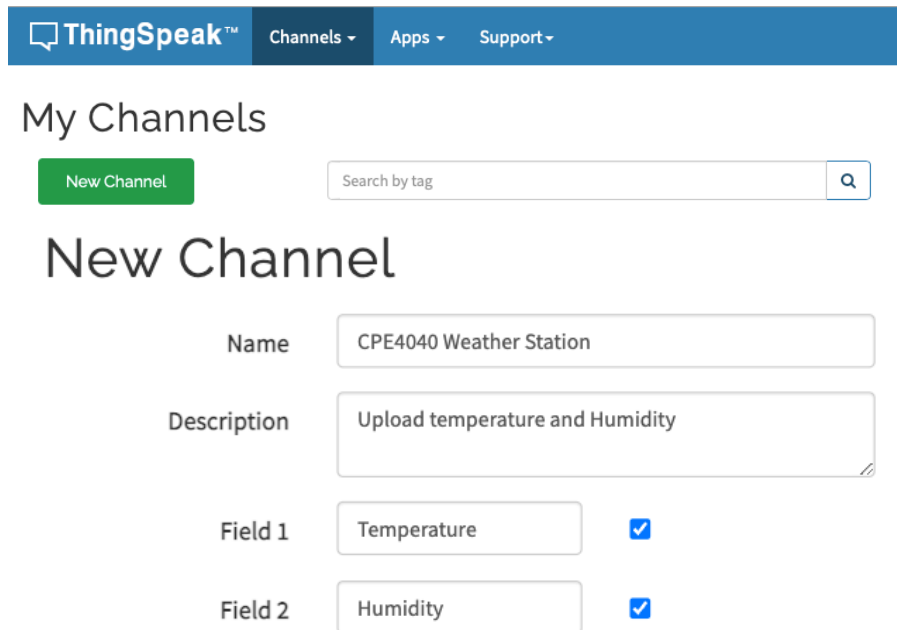
# Print the temperature and humidity values to the terminal
temperature = dhtDevice.temperature
humidity = dhtDevice.humidity
print(f"Temp: {temperature:.1f}C, Humidity: {humidity}%")
```

7. Check whether the temperature and humidity readings are reasonable. **Add a small piece of code to convert temperature values from C (Celsius) to F (Fahrenheit).**

## Section 2: Create a ThingSpeak Channel for the MQTT Device

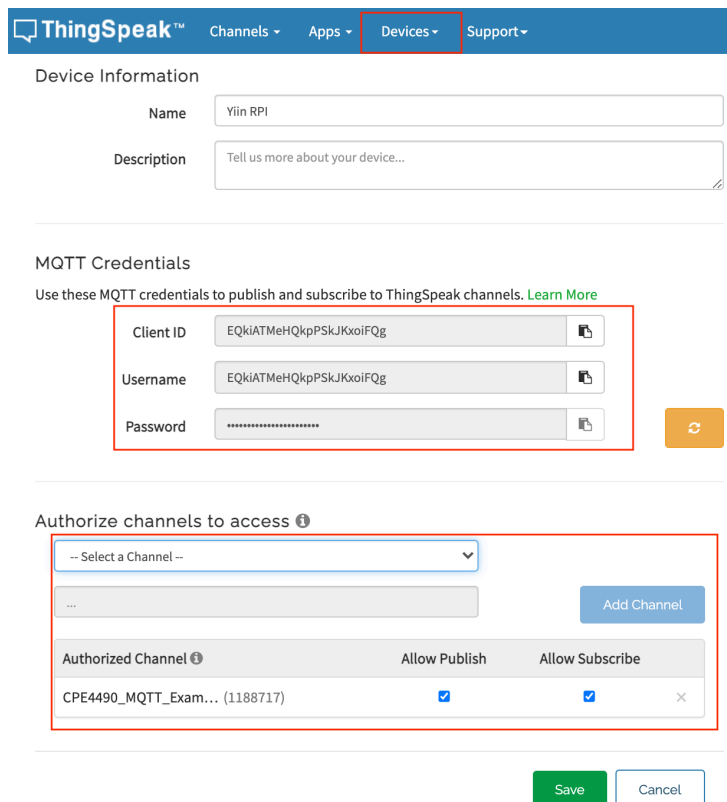
8. Sign up a free ThingSpeak account. If you already have a MathWorks (MATLAB) account, go ahead and use your login name and password. Otherwise, create one using your KSU email.

9. In order to upload data to the ThingSpeak Cloud, you have to create a **Channel** first. A Channel allows you to send and receive data from the ThingSpeak server. Click the “New Channel” button and enter a Name and Description for your new channel. In Field 1 and Field 2, fill in Temperature and Humidity, respectively, as shown in the figure below.



The screenshot shows the ThingSpeak web interface. At the top is a blue navigation bar with the ThingSpeak logo and links for Channels, Apps, and Support. Below the navigation bar is a section titled 'My Channels' with a green 'New Channel' button and a search bar. The 'New Channel' form is displayed, featuring a 'Name' field with the text 'CPE4040 Weather Station', a 'Description' field with the text 'Upload temperature and Humidity', and two 'Field' sections. 'Field 1' is labeled 'Temperature' and 'Field 2' is labeled 'Humidity', both with checkboxes that are checked.

10. Go to “Devices” in the menu bar. This is where you create an MQTT device (your Raspberry Pi). Give the device a name, add the channel that you just created, then click “Add Channel”. The MQTT credentials of this device will show up (Client ID, Username and Password). Save them in a separate text file as we will use them in Step 13.



The screenshot shows the ThingSpeak 'Devices' page. The navigation bar has 'Devices' highlighted. Under 'Device Information', the 'Name' field contains 'Yiin RPI' and the 'Description' field contains 'Tell us more about your device...'. Below this is the 'MQTT Credentials' section, which includes a note to use these credentials to publish and subscribe to ThingSpeak channels, with a 'Learn More' link. The credentials are displayed in three fields: 'Client ID' (EQkiATMeHQkPPSkJKxoIFQg), 'Username' (EQkiATMeHQkPPSkJKxoIFQg), and 'Password' (a masked field). To the right of these fields is a blue 'Add Channel' button. Below the credentials is the 'Authorize channels to access' section, which includes a dropdown menu to select a channel. Below the dropdown is a table with columns 'Authorized Channel', 'Allow Publish', and 'Allow Subscribe'. The table contains one row with the channel 'CPE4490\_MQTT\_Exam... (1188717)' and both 'Allow Publish' and 'Allow Subscribe' checkboxes checked. At the bottom right are 'Save' and 'Cancel' buttons.

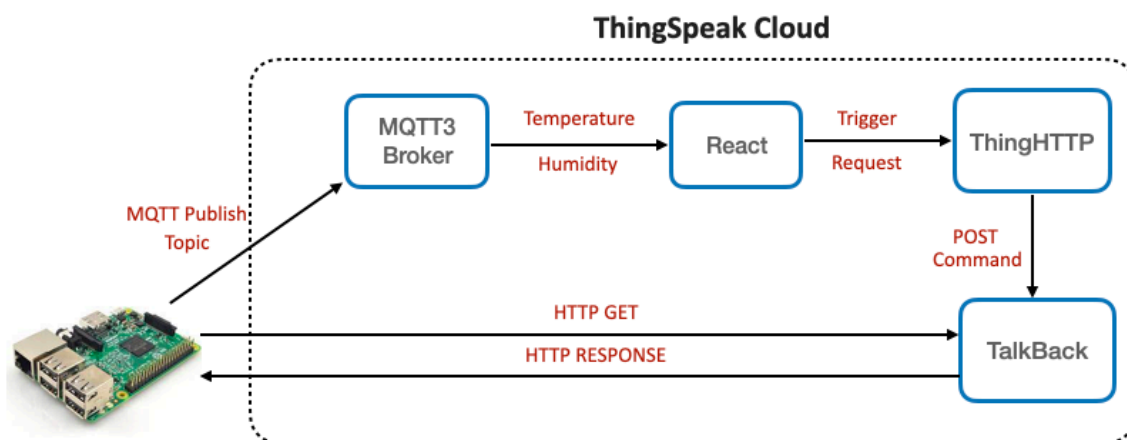
11. In "Channel Settings", scroll down and check the box "Show Channel Location". You can find the latitude and longitude of your location from Google Maps, and paste it into the corresponding boxes that are now available under Channel Settings. When you are done, save the channel and you will be redirected to the main page of the newly created channel. Write down "Channel ID", which will be used shortly.
12. Go to "API Keys" in the manual bar and write down the key in the "Write API Key" section that will be used to configure your code.
13. Open the reference Python program, ***publish.py***, and modify the code as instructed. When running the code, you will start to see sensor data posting to your ThingSpeak channel so you can monitor in real time.

**Note:** For channel ID, username and password, you will use the ones obtained in Step 10. In the *while* loop, include the modification that you did in Step 7 to convert the temperature unit to Fahrenheit.

14. Connect an LED to GPIO12 of Raspberry Pi with a serial resistor (the same thing you did in Lab 3).

### Section 3: Sending Alert from Your Weather Station

When the temperature exceeds 75°F, an alert will be sent to Raspberry Pi and the LED will be turned on. This will be achieved using three ThingSpeak applications, as shown in the flowchart below.



15. To generate an action when the temperature exceeds 75°F, we will use the **TalkBack** app. Go to Apps on the ThingSpeak home page and select TalkBack. Select the "New TalkBack" button, then enter a name to select your channel in the "Log to Channel" option. Save and exit.

TalkBack is where Raspberry Pi periodically accesses and polls commands. To generate commands for the result of events, we use the **React** app in ThingSpeak.

16. We will create two new Reacts: one for the temperature exceeding above 75°F and one for the temperature dropping below 75°F. Both are similarly defined as in the figure below.

Apps / React / New

React Name: Temp\_over\_75F

Condition Type: Numeric

Test Frequency: On Data Insertion

Condition: If channel  
CPE4040 Weather Station (1662449)

field: 1 (Temperature)

is greater than or equal to

75

Action: ThingHTTP

You need to add a ThingHTTP request  
[Add ThingHTTP request](#)

Options: ☒ Run action only the first time the condition is met  
☐ Run action each time condition is met

17. When the condition is met in **React**, a trigger will be sent to **ThingHTTP** to generate a POST command to **TalkBack**. You will set up two separate ThingHTTP requests. For each request, fill in the configuration settings as illustrated in the figure below. **Note:** In the URL section, you will need to enter your TalkBack ID, and in the Body section, you will enter the your TalkBack API Key.

- URL: `https://api.thingspeak.com/talkbacks/<TALKBACK_ID>/commands.json`
- Method: POST
- Content Type: `application/x-www-form-urlencoded`
- HTTP Version: 1.1
- Host: `api.thingspeak.com`
- Body: `api_key=<TALKBACK_API_KEY>&command_string=<COMMAND>`
- For the body, **COMMAND** should either be **TURN\_ON** (warning) or **TURN\_OFF** (normal).

Name:	TempWarning
API Key:	TIGMB9RV895C71R9
	<a href="#">Regenerate API Key</a>
URL:	https://api.thingspeak.com/talkbacks/31470/commands
HTTP Auth Username:	
HTTP Auth Password:	
Method:	POST
Content Type:	application/x-www-form-urlencoded
HTTP Version:	1.1
Host:	api.thingspeak.com
Headers:	
Body:	api_key=TUOAAAMH7MU195QI5&command_string=TURN_ON
Parse String:	
Created:	2019-03-05 6:40 pm

Name:	TempNormal
API Key:	GUAB52IUC30U00XF
	<a href="#">Regenerate API Key</a>
URL:	https://api.thingspeak.com/talkbacks/31470/commands
HTTP Auth Username:	
HTTP Auth Password:	
Method:	POST
Content Type:	application/x-www-form-urlencoded
HTTP Version:	1.1
Host:	api.thingspeak.com
Headers:	
Body:	api_key=TUOAAAMH7MU195QI5&command_string=TURN_OFF
Parse String:	
Created:	2019-03-05 6:57 pm

18. After setting up ThingSpeak, open the reference Python program, *tempAlert.py*, and modify the code as instructed. It will make HTTP connection to poll commands from the TalkBack app to turn on or off the LED.

**Note:** Again, do not forget to modify the TalkBack ID and the API Keys to your own.

When you execute the code, the LED should be turned on when the temperature goes above 75°F and turned off when it drops below 75°F. You can place your finger on the sensor or use a hair dryer to accelerate the heat-up!

You should run this code on another terminal while *publish.py* is still running.

## Lab Report:

Please include the following in your report:

1. A photo of your hardware setup.
2. Screenshots from step 6 to step 18.
3. A video that clearly shows the LED was turned on or off, in response to the temperature change.
4. The three Python programs you created: *digitalOut.py*, *publish.py*, and *tempAlert.py*.
5. Place the device in your room with the code running for at least **one hour with one minute data update interval**. Save and submit the temperature and humidity data in an Excel or CSV file. Show the plots from the Weather Station (that is, in the ThingSpeak Channel) with the whole data set.

Submit the report via D2L Assignment, in PDF format.