Introduction to Computer Networks (ENME506)

Done by T-06:

* Mariam Magdy Maher 1000659
* Ahmed Sherif Elarabi 1002199
* Ahmed Khaled Hegazy 1002701
* Farida Hisham Elhelaly 1000960
* Omar Fadel 100 2097
* Omar Yasser 1002195



Weather Station Project

# 1.Intro:

Weather or Climate is an important part of human life.   
Sensors are essential components not only applicable to the industries for process control but also in daily life for safety of buildings and traffic flow measuring, environmental parameters measurement.

IoT means Internet of Things. It provides inter-networking of physical devices, buildings, vehicles and other components like sensors and actuators.

Humidity, Temperature and Rain detection are three basic parameters to build any Weather Station and to measure environmental conditions. This semester we ae building this project using multiple weather measure sensors and a Raspberry Pi 4 board. This [IoT based Project](http://circuitdigest.com/internet-of-things-iot-projects) aims to show the current Humidity, Temperature and Rain detection parameters on the Internet server using Raspberry Pi, which makes it a Raspberry Pi Weather Station.

The two sensors used in this project are the DHT22 which measures the temperature & Humidity, and Fc-37 which is a rain detector sensor. The physical quantities measured in this project are only temperature, humidity and rain detection. The readings received from these sensors will be sent to a ThingSpeak server for live monitoring and an acknowledgment email will be sent along too.

# 

Diagram

Description automatically generated

Figures 1 & 2. Schematic diagrams representing the components connections

# A picture containing graphical user interface Description automatically generated

Figure 3. Block diagram example for the overall system

# 2 Hardware components included:

## 2.1 Development board (Raspberry Pi 4):

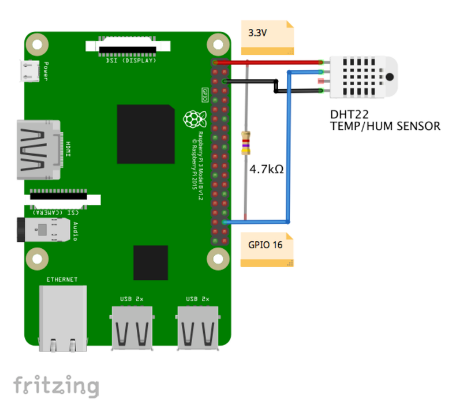
Raspberry PI is a card-sized ARM powered Linux computer development board. There are in total of 5 types of various board with different specification, for the proposed Weather forecasting system Raspberry PI 4 model B is used as the main development board which is shown in Figure 4.

A picture containing text, electronics, circuit

Description automatically generatedThe Raspberry Pi 4 Model B is the latest board launched by the Raspberry Pi Foundation in June 2019. This model has the latest high-performance quad-Core **64-bit Broadcom 2711**, **Cortex A72** processor clocked at 1.5GHz speed.

Figure 4. Raspberry PI model 4 board.

## 2.2 [DHT22](https://www.adafruit.com/product/385)Temperature & Humidity Sensor:

The first sensor to be installed will be the DHT22 for capturing air temperature and relative humidity data. The [ADAFRUIT site](https://learn.adafruit.com/dht/overview) provides great information about those sensors.

Below, some information retrieved from there:

**Overview**

The low-cost DHT temperature & humidity sensors are very basic and slow but are great for hobbyists who want to do some basic data logging. The DHT sensors are made of two parts, a capacitive humidity sensor, and a thermistor. There is also a very basic chip inside that does some analog to digital conversion and spits out a digital signal with the temperature and humidity. The digital signal is easy to be read using any microcontroller.

DHT22 Main characteristics:

* Low cost
* 3 to 5V power and I/O
* 2.5mA max current use during conversion (while requesting data)
* Good for 0-100% humidity readings with 2-5% accuracy
* Good for -40 to 125°C temperature readings ±0.5°C accuracy
* No more than 0.5 Hz sampling rate (once every 2 seconds)
* Body size 15.1mm x 25mm x 7.7mm
* 4 pins with 0.1" spacing

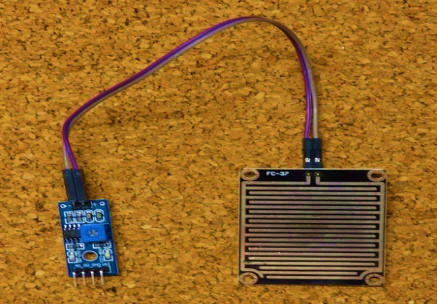
Price: EGP115.00

## 2.3 FC37 Rain Detector Sensor:

The rain sensor is used to detect water and it can detect beyond of what a humidity sensor do. It will display “It’s raining” when it detects water, which is in our case is rain.

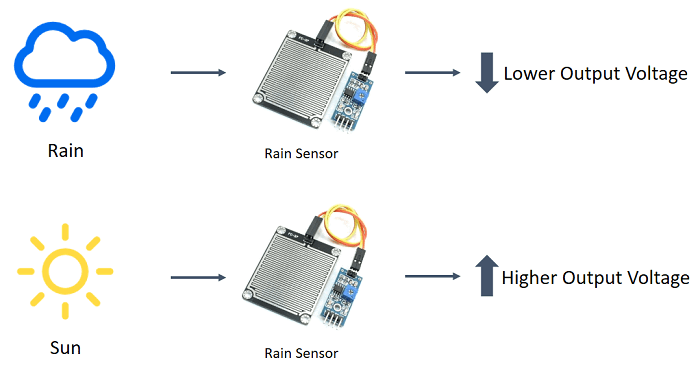
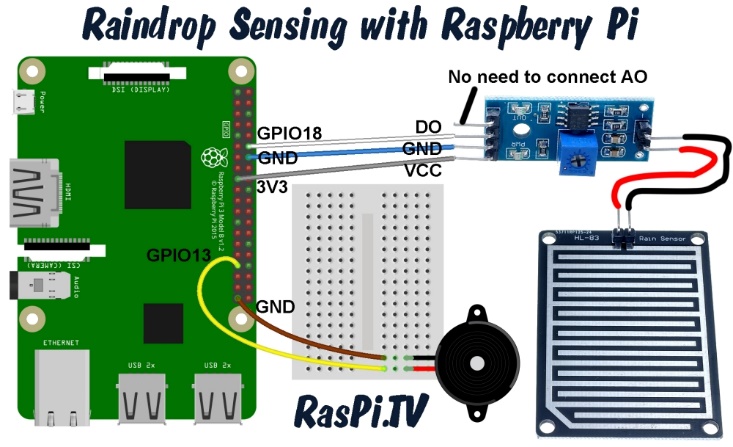
It has a built-in potentiometer for sensitivity adjustment of the digital output (D0). It also has a power LED that lights up when the sensor is turned on and a digital output LED.

Basically, the resistance of the collector board varies accordingly to the amount of water on its surface.

When the board is:

* **Wet:** the resistance increases, and the output voltage decreases
* **Dry:** the resistance is lower, and the output voltage is higher

Price: EGP100.00



And this is the list of the components we used

|  |  |
| --- | --- |
| Component | Price |
| Raspberry PI | Provided by the UNI |
| Rain Detector Sensor | 100 EGP |
| DHT22 (Precision Humidity & Temperature Sensor) | 115/40 EGP |
| Jumper Wires | 30 EGP |
| Buzzer | 5 EGP |
| Breadboard | 50 EGP |
| 10k Ohm Resistors | 60 EGP |

# Softwares used:

1. Thonny IDE: which is [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) for [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) that is designed for beginners. It supports different ways of stepping through the code, step-by-step expression evaluation, detailed visualization of the call stack and a mode for explaining the concepts of references and heap.(Programming Language: Python)

2. Thingspeak (**‘Collects’** the data from the sensors, **‘Analyse and Visualize’** the data and **‘Acts’** by triggering a reaction)

Diagram

Description automatically generated

# State Machine Model:

**Function specifications:**

This IoT based Project aims to show the current Humidity, Temperature and Rain detection parameters on the Internet server using Raspberry Pi, which makes it a Raspberry Pi Weather Station.

**Firstly** DHT22 sensor senses the Humidity & Temperature Data and Fc37 sensor which detects rain.

**Secondly** Raspberry Pi reads the DHT22 sensor module’s output by using single wire protocol and rain detection sensor and extracts both sensors values into a suitable number in percentage (humidity), Celsius scale (temperature), High or Low detection of rain (rain detection).

**Thirdly,** these values are sent to ThingSpeak server by using inbuilt Wi-Fi of Raspberry Pi 4 and also an email is sent to the receiver’s email acknowledging the data values from both sensors.

**Finally** ThingSpeak analyses the data and shows it in a Graph form and an email is sent which displays the data numerically.

**Protocols applications:**

After getting the data from the sensors, create a single loop function to capture the data at regular bases, saving them on a local file. Now we use a Protocol application to send the data over the Iot platform.

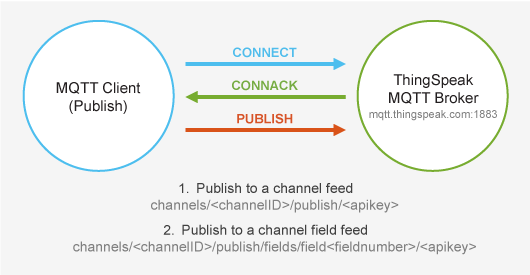
We will use ThingSpeak.com. “ThingSpeak is an open source Internet of Things (IoT) application to store and retrieve data from things, using REST and MQTT APIs. ThingSpeak enables the creation of sensor logging applications, location tracking applications, and a social network of things with status updates."

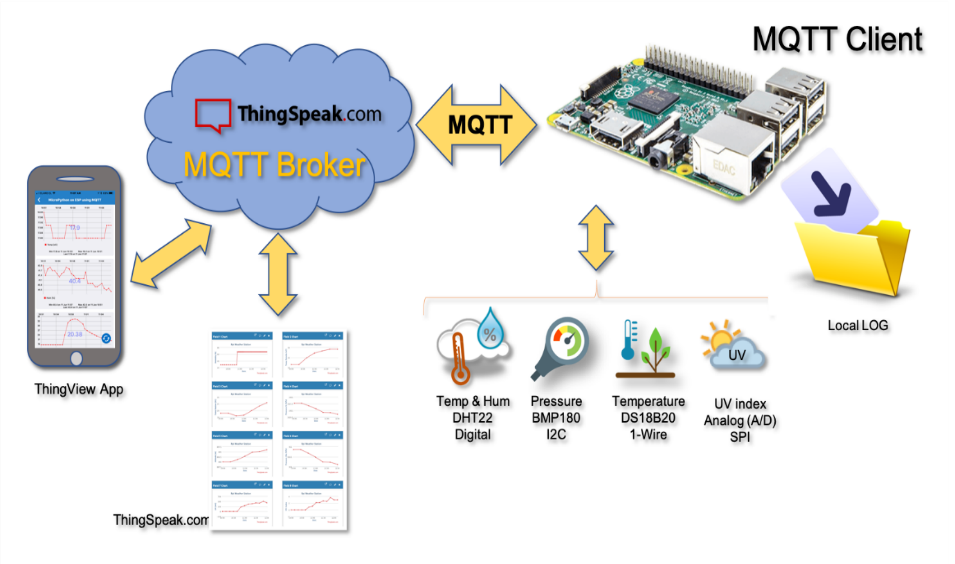
you must have an account at ThinkSpeak.com. Next, follow the instructions to create a Channel and take note of your Channel ID and Write API Key.

MQTT is a publish/subscribe architecture that is developed primarily to connect bandwidth and power-constrained devices over wireless networks. It is a simple and lightweight protocol that runs over TCP/IP sockets or WebSockets. MQTT over WebSockets can be secured with SSL. The publish/subscribe architecture enables messages to be pushed to the client devices without the device needing to continuously poll the server.

The MQTT broker is the central point of communication, and it is in charge of dispatching all messages between the senders and the rightful receivers. A client is any device that connects to the broker and can publish or subscribe to topics to access the information. A topic contains routing information for the broker. Each client that wants to send messages publishes them to a certain topic, and each client that wants to receive messages subscribes to a certain topic. The broker delivers all messages with the matching topic to the appropriate clients.

ThingSpeak has an MQTT broker at the URL mqtt.thingspeak.com and port 1883. The ThingSpeak broker supports both MQTT publish and MQTT subscribe. In our case, we will use the MQTT Publish.

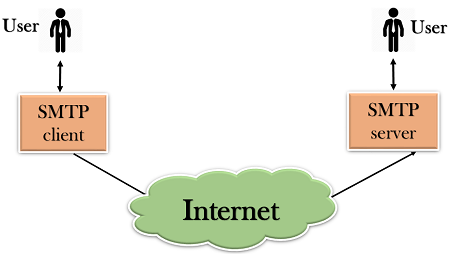
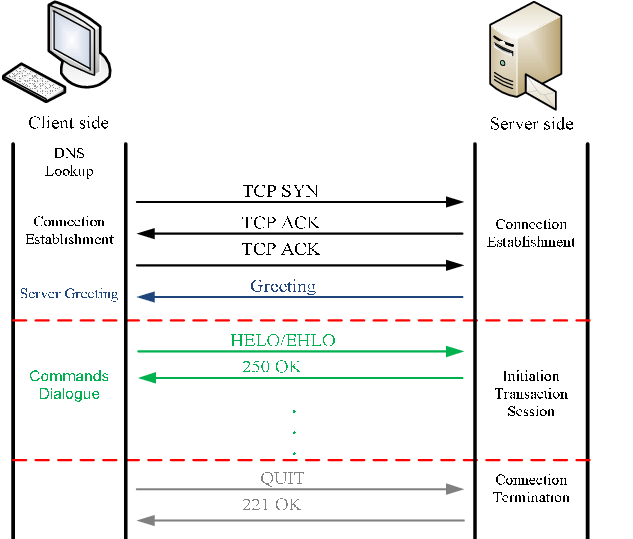




Parallel to this, we added another communication protocol which is SMTP. SMTP stands for Simple Mail Transfer Protocol which uses TCP connection, and it’s an application used by mail servers to send, receive, and/or relay outgoing mail between email senders and receivers.

An SMTP email server will have an address (or addresses) that can be set by the mail client or application that you are using and is generally formatted as smtp.serveraddress.com. For example, the SMTP server Gmail uses is smtp.gmail.com

When you send an email, with SMTP host Gmail or AOL, the SMTP server processes your email, decides which server to send the message to, and relays the message to that server. The recipient’s inbox service provider, such as Gmail or AOL, then downloads the message and places it in the recipient’s inbox.



**The code we followed for both sensors:**

**For the rain sensor:**

import urllib.request

import requests

import threading

import json

import random

import RPi.GPIO as GPIO

import time

import smtplib

from time import sleep

from gpiozero import Buzzer, InputDevice

server =smtplib.SMTP\_SSL("smtp.gmail.com",465)

server.login("mariam.m.maher.18@gmail.com","mariammagdy18")

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BOARD)

GPIO.setup(18,GPIO.IN)

buzz = Buzzer(13)

no\_rain = InputDevice(18)

rain = True

def buzz\_now(iterations):

for x in range(iterations):

buzz.on()

sleep(0.1)

buzz.off()

sleep(0.1)

def thingspeak\_post():

threading.Timer(15,thingspeak\_post).start()

URl='https://api.thingspeak.com/update?api\_key='

KEY='SRV8DX5KJDI268CH'

HEADER='&field3={}'.format(rain)

NEW\_URL = URl+KEY+HEADER

print(NEW\_URL)

data=urllib.request.urlopen(NEW\_URL)

print(data)

while True:

if not no\_rain.is\_active:

print("It's raining - Take an umbrella")

buzz\_now(5)

rain = 1

server.sendmail("mariam.m.maher.18@gmail.com","ahmedelarabi935@gmail.com","It's raining")

time.sleep(2)

thingspeak\_post()

server.quit()

# insert your other code or functions here

# e.g. tweet, SMS, email, take a photo etc.

elif no\_rain.is\_active:

rain = 0

print("The weather's clear")

thingspeak\_post()

sleep(1)

**As for the DHT22:**

import time

import board

import adafruit\_dht

import psutil

import urllib.request

import requests

import threading

import json

import random

import smtplib

server =smtplib.SMTP\_SSL("smtp.gmail.com",465)

server.login("mariam.m.maher.18@gmail.com","mariammagdy18")

# We first check if a libgpiod process is running. If yes, we kill it!

for proc in psutil.process\_iter():

if proc.name() == 'libgpiod\_pulsein' or proc.name() == 'libgpiod\_pulsei':

proc.kill()

sensor = adafruit\_dht.DHT22(board.D4)

def thingspeak\_post():

threading.Timer(15,thingspeak\_post).start()

URl='https://api.thingspeak.com/update?api\_key='

KEY='SRV8DX5KJDI268CH'

HEADER='&field1={}&field2={}'.format(temp, humidity)

NEW\_URL = URl+KEY+HEADER

print(NEW\_URL)

data=urllib.request.urlopen(NEW\_URL)

print(data)

while True:

try:

temp = sensor.temperature

humidity = sensor.humidity

print("Temperature: {}\*C Humidity: {}% ".format(temp, humidity))

thingspeak\_post()

server.sendmail("mariam.m.maher.18@gmail.com","ahmedelarabi935@gmail.com",str(temp)+"\*C (Temperature) "+str(humidity)+"% (Humidity)")

server.quit()

except RuntimeError as error:

print(error.args[0])

time.sleep(2.0)

continue

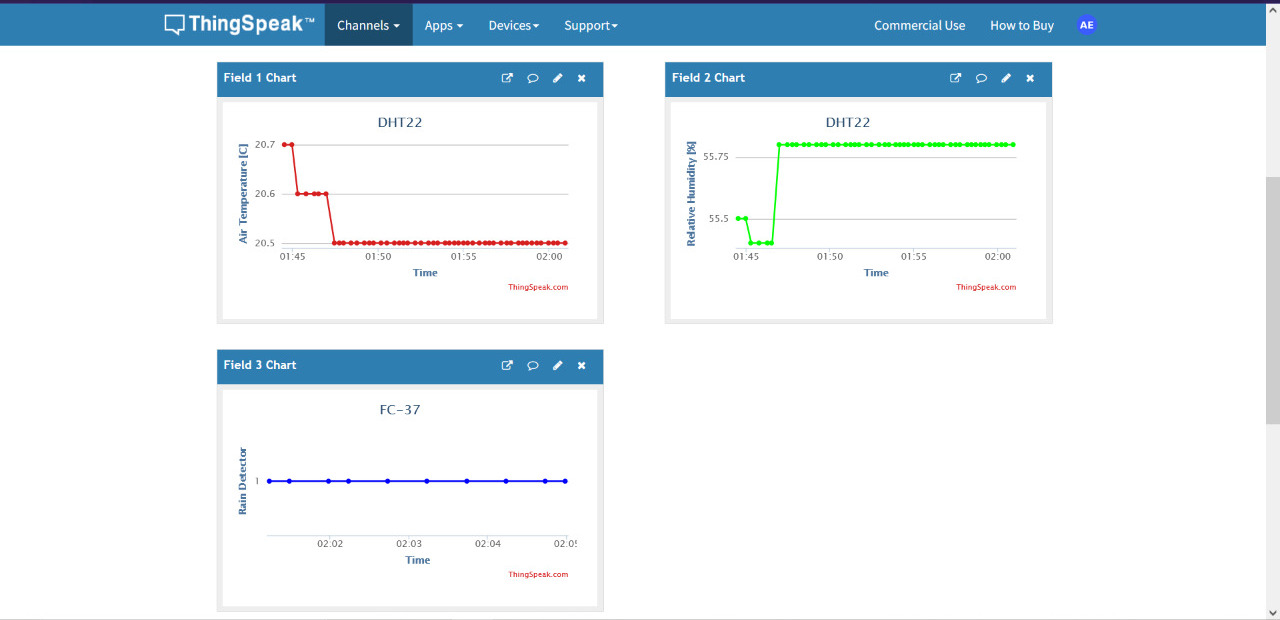
except Exception as error:

sensor.exit()

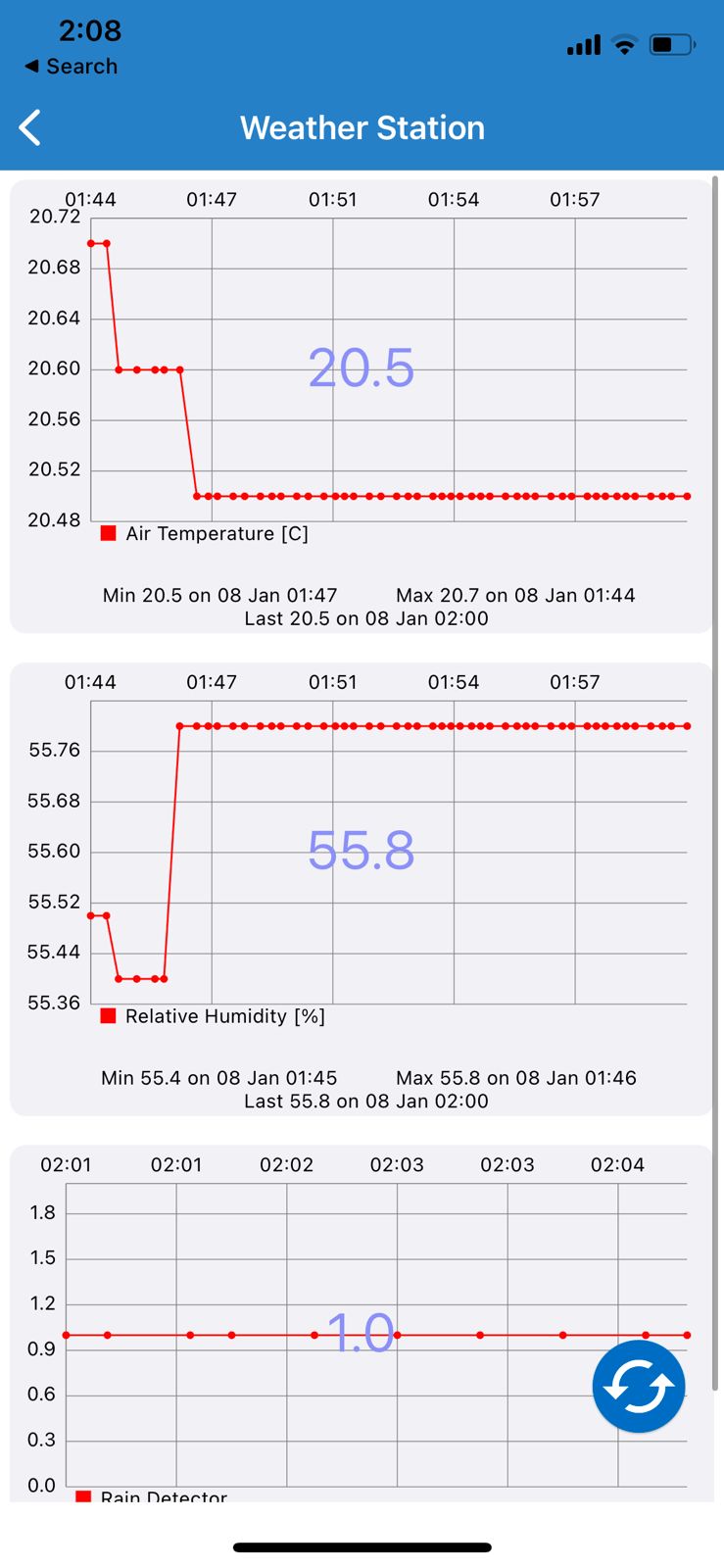
raise error

time.sleep(2.0)

**And these are the data after being sent successfully using our 2 communication protocols:**

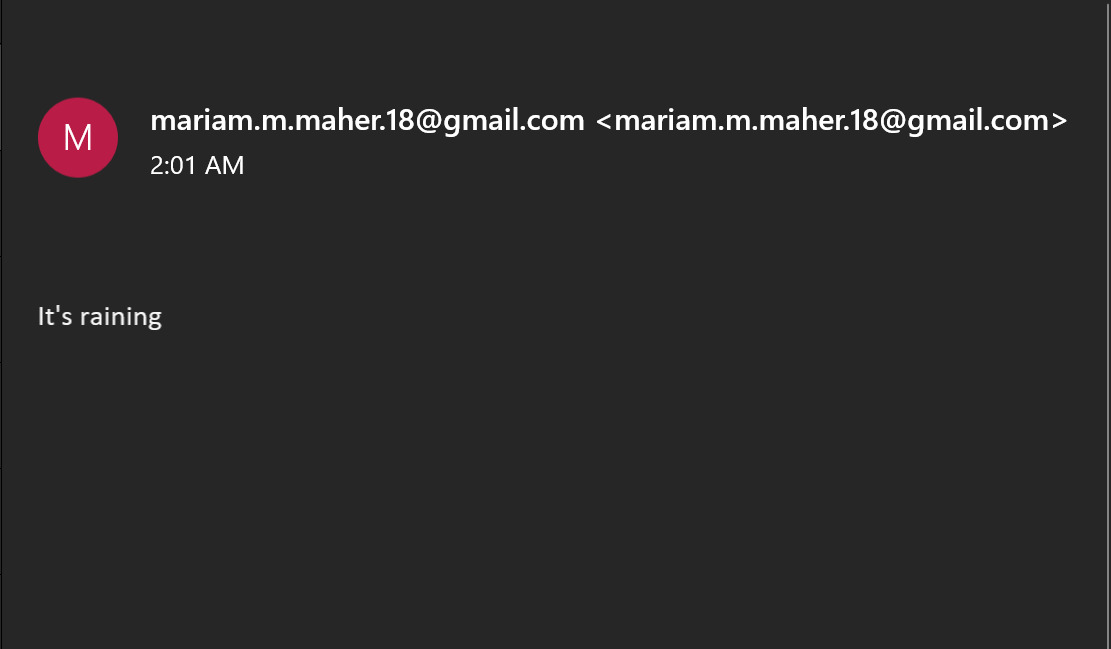
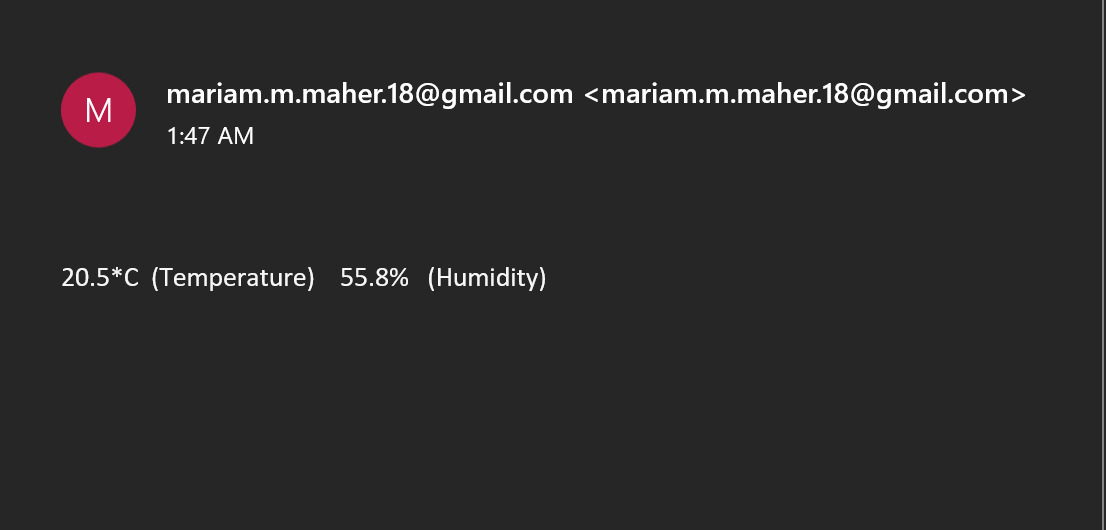


Data sent to our channel on the website



Accessing the channel on mobile App

And these are the acknowledgment emails for both sensors



**References**

Ansgar Meroth, Petre Sora, Sensornetzwerke in Theorie und Praxis, Embedded Systems-Projekte erfolgreich realisieren

<https://sendgrid.com/blog/what-is-an-smtp-server/>

<https://tutorials-raspberrypi.com/log-raspberry-pi-sensor-data-with-thingspeak-and-analyze-it/>

<https://raspi.tv/2017/make-a-rain-alert-system-with-raspberry-pi>

<https://github.com/mistylackie/solar-water-bot/blob/master/RainSensor.py>

<https://www.ibm.com/docs/en/ibm-mq/8.0?topic=telemetry-send-message-mqtt-client>

<https://kb.synology.com/en-global/SRM/tutorial/How_to_use_Gmail_SMTP_server_to_send_emails_for_SRM>