

# 9238-MANGAYARKARASI COLLEGE OF ENGINEERING

(Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai)

MANGAYARKARASI NAGAR, PARAVAI, MADURAI -625 402

Website: <http://mce-madurai.ac.in>

E-Mail: : mangai.enggcoll@gmail.com

---

## IOT BASED AIR QUALITY MONITORING

### ***PROJECT MEMBERS***

**T.ABINAYA. - 923821106003**

**R.MAHALAKSHMI. - 923821106024**

**D.MANGALA JOTHI. - 923821106026**

**J.NOURIN RIFFANA. - 923821106031**

**M.VARSHINI. - 923821106054**

### ***PROJECT GUIDE***

**Mr.R.M.SENTHILKUMAR – AP/ECE**

# INTRODUCTION

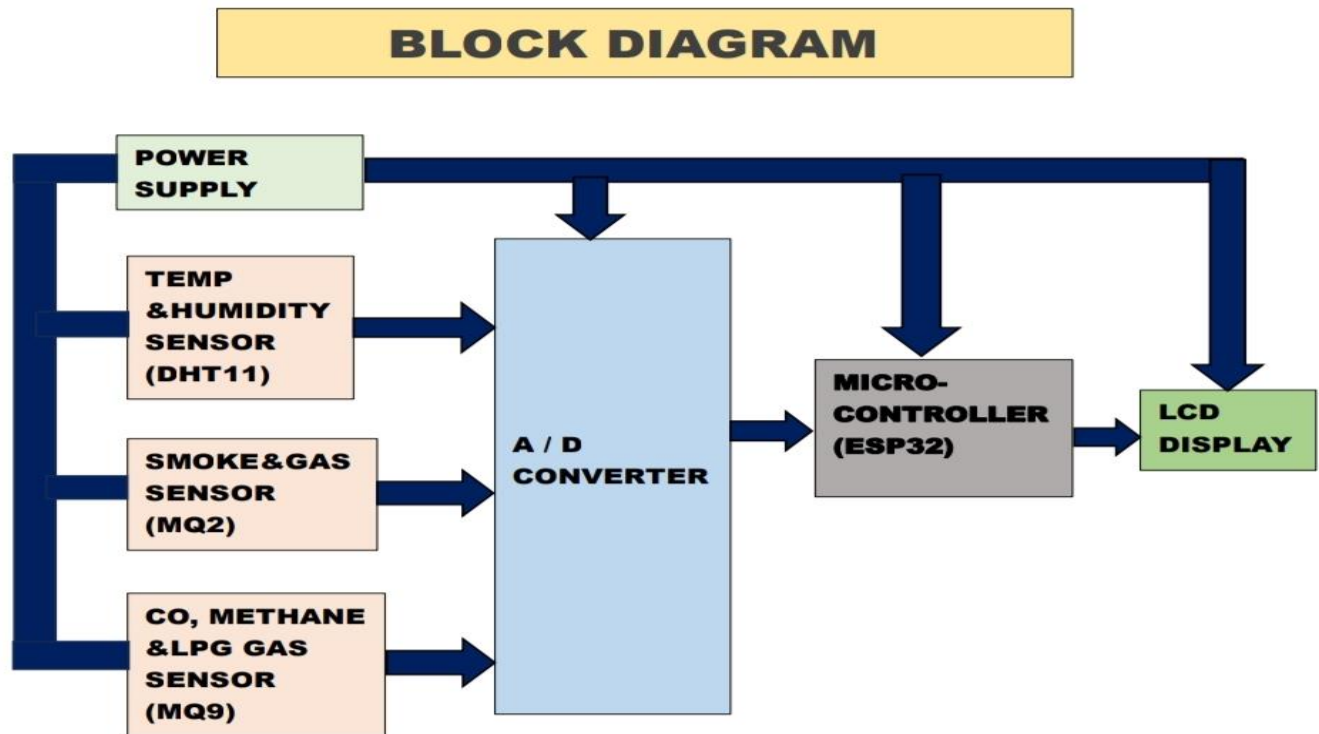


- An IOT-based air pollution monitoring system is an ideal solution that can provide real-time data and insights about the air quality in a particular area.
- An IOT based air pollution monitoring system consists of several hardware and software components that work together to collect and process data.

## PROJECT OBJECTIVES

- ❖ *Air quality monitoring refers to the collection and measurement of ambient air pollution samples.*
- ❖ *The data from these samples is compared to clean air standards and historical information regarding air quality levels, along with data reflecting its health and environmental impacts, to determine the state of the air.*
- ❖ *It allow the measurement, operation and predictive analysis of the evolution of air pollution in different areas (urban areas, industrial areas, special nature conservation areas, etc.).*
- ❖ The primary focus of air pollution regulation in industrialized countries has been on protecting ambient, or outdoor, air quality.
- ❖ This involves the control of a small number of specific “criteria” pollutants known to contribute to urban smog and chronic public health problems.
- ❖ To plan a comprehensive programme for the prevention, control or abatement of air pollution and to secure the execution thereof. To advise the State Government on any matter concerning the prevention, control or abatement of air pollution.

# IOT DEVICE SETUP



## ✓ *Microcontroller(ESP32)*

- *ESP32 is widely used SoC microcontroller developed by Espressif Systems. It is low cost and highly versatile microcontroller used for various applications which include wireless communication, IoT (Internet of things) devices, home automation, robotics, embedded systems etc.*
- *Features of the ESP32 include the following: Processors: CPU: Xtensa dual-core (or single-core) 32-bit LX6 microprocessor, operating at 160 or 240 MHz and performing at up to 600 DMIPS.*

### ✓ *MQ2 sensor*

- *The MQ-2 is a smoke and combustible gas sensor from Winsen. It can detect flammable gas in a range of 300 – 10000ppm. It's most common use is domestic gas leakage alarms and detectors with a high sensitivity to propane and smoke.*
- *It can detect flammable gas in a range of 300 – 10000ppm.*

### ✓ *MQ9 sensor*

- *The Grove – Gas Sensor(MQ9) module is useful for gas leakage detection (in home and industry). It is suitable for detecting LPG, CO, CH<sub>4</sub>. Due to its high sensitivity and fast response time, measurements can be taken as soon as possible. The sensitivity of the sensor can be adjusted by using the potentiometer.*
- *The sensor can measure concentrations of CO from 10 to 1,000 ppm and flammable gas from 100 to 10,000 ppm.*

### ✓ *DHT11 sensor*

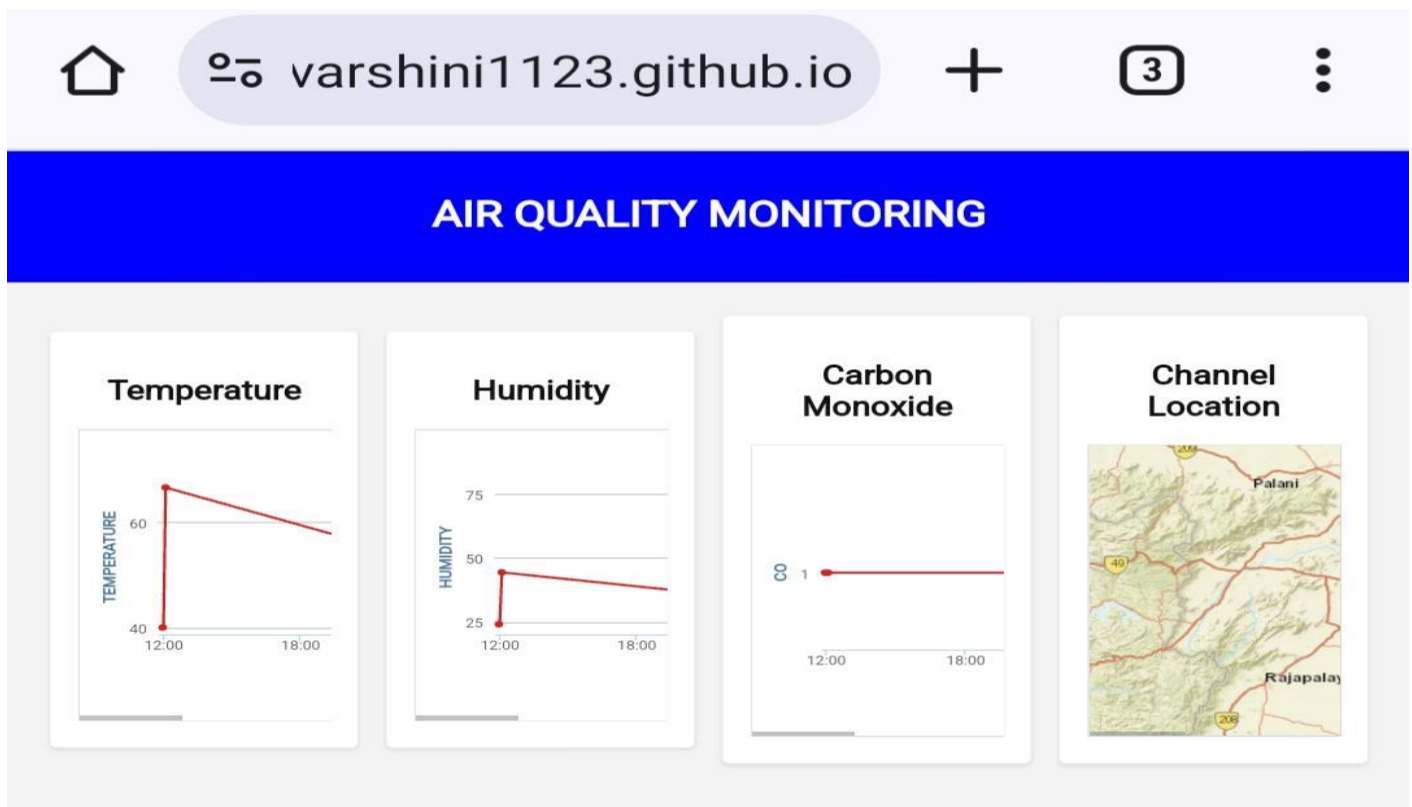
- *The DHT11 is a basic, ultra low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin (no analog input pins needed).*
- *It has a temperature measurement range of -40°C to +125°C with +-0.5°C precision and has a temperature range of 0°C to 50°C with +-2 degrees accuracy.*

### ✓ *A/D converter*

- *An analog-to-digital converter (ADC) is used to convert an analog signal such as voltage to a digital form so that it can be read and processed by a microcontroller. Most microcontrollers nowadays have built-in ADC converters.*
- *The ADC range is the maximum and minimum ADC input (e.g., 0 to +3.3V).*

# PLATFORM DEVELOPMENT

Our website is a user-friendly platform designed For air quality Enthusiasts and data aficionados. It provides Real-time access to essential Quality information, such As temperature and humidity. The site offers an Intuitive and Visually appealing interface, displaying data in an easily Digestible format Users can stay informed about air quality conditions in Their Preferred locations, be it environment, gardens, or other outdoor Settings.





The website extracts data from ThingSpeak, ensuring Accuracy and reliability. This data is presented with clean, organized design elements, allowing users To track Environmental trends and make informed decisions. Our website is a Valuable resource for both casual observers and Serious environmentalists, About air quality offering a seamless and Experience for exploring and Understanding the world around us.

YOU CAN ACCESS OUR WEBSITE USING THE BELOW URL

<https://Varshini1123.github.io>

## CODE IMPLEMENTATION

Wokwi is an embedded systems and IOT simulator supporting ESP32, Arduino, and the Raspberry Pi Pico. Your code never leaves your computer –Wokwi runs the simulation inside VS Code, using the firmware binaries from Your project.Wokwi is a platform that allows you to simulate And test your code for Microcontroller-based projects, including Those written in Python for Microcontrollers like the ESP32.

The provided python program is designed for an ESP8266 based to Monitor temperature and humidity using a DHT11 sensor and send that Data To ThingSpeak, a cloud – based IOT platform .

Import machine

Import time

Import network

Import dht

Import urequests

# DHT22 data pin connected to GPIO5 (D1)

Dht\_pin = machine.Pin(23)

# IR sensor pin

Ir\_pin = machine.Pin(15)

# WiFi credentials

Ssid = "Wokwi-GUEST"

Password = ""

# ThingSpeak API key

Api\_key = "R4PC4HW83LV9O3KE"

# ThingSpeak server

Ts\_server = "api.thingspeak.com"

Channel\_number = 2319907



```

# Connect to WiFi

Def connect_wifi():

Sta_if = network.WLAN(network.STA_IF)

If not sta_if.isconnected():

Print("Connecting to WiFi...")

Sta_if.active(True)

Sta_if.connect(ssid, password)

While not sta_if.isconnected():

Pass

Print("Connected to WiFi")

# Function to read and return temperature, humidity, and IR data

Def read_sensors():

Dht22 = dht.DHT22(dht_pin) # Create the DHT22 object

Dht22.measure()

Temp, hum = dht22.temperature(), dht22.humidity()

Ir_data = ir_pin.value() # Read IR data

Return temp, hum, ir_data

```

```

# Send data to ThingSpeak

Def send_to_thingspeak(temp, hum, ir_data):

Base_url =

http:///{}/update?api\_key={}&field1={}&field2={}&field3={}.format\(

Ts\_server, api\_key, temp, hum, ir\_data\)

Response = urequests.get\(base\_url\)

If response.status\_code == 200:

Print\("Data sent to ThingSpeak successfully"\)

Else:

Print\("Failed to send data to ThingSpeak"\)

# Main program

Def main\(\):

Connect\_wifi\(\)

While True:

Temp, hum, ir\_data = read\_sensors\(\)

Print\("Temperature: {}°C, Humidity: {}%, IR Data: {}".format\(temp,
Hum, ir\_data\)\)

```

```
Send_to_thingspeak(temp, hum, ir_data)
```

```
Time.sleep(300) # Send data every 5 minutes
```

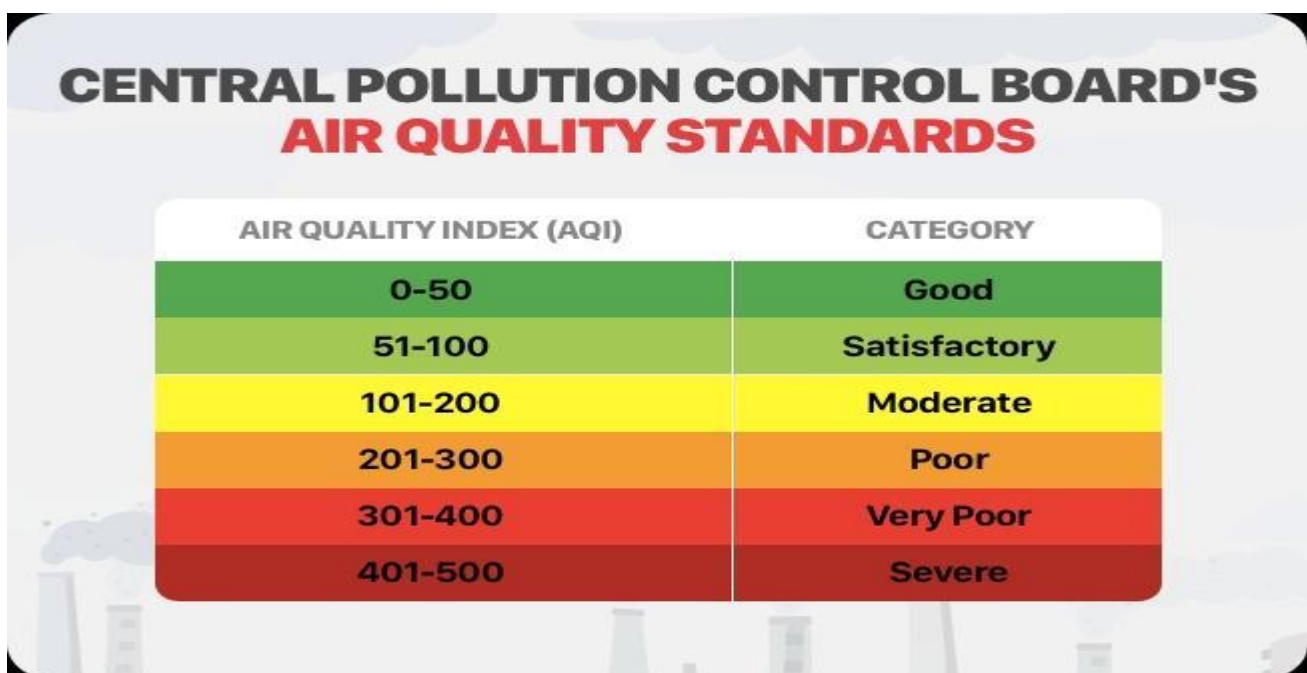
```
If __name__ == '__main__':
```

```
Main()
```

The Above Program Send the data to the Thingspeak server. The Data Get From the Microcontroller ESP32. Humidity and Temperature Can be Get Using the DHT 22 Sensor. The Data will Send for Every 5 Minutes The Program Access The ThingSpeak server Via the API key.

## AIR QUALITY INDEX MEASURING

An air quality index is an indicator developed by government agencies to communicate to the public how polluted the air currently is or how polluted it is forecast to become. As air pollution levels rise, so does the AQI, along with the associated public health risk.



The infographic features a title 'CENTRAL POLLUTION CONTROL BOARD'S AIR QUALITY STANDARDS' in bold black and red text. Below the title is a table with two columns: 'AIR QUALITY INDEX (AQI)' and 'CATEGORY'. The table has six rows, each with a different background color corresponding to the AQI range and category. The background of the entire infographic shows a stylized city skyline with smokestacks.

AIR QUALITY INDEX (AQI)	CATEGORY
0-50	Good
51-100	Satisfactory
101-200	Moderate
201-300	Poor
301-400	Very Poor
401-500	Severe

## Current air quality measurement



## OUTPUT FROM WOKWI

We can adjust the Temperature and Humidity level Of the sensor Because this is the simulation process Not having a physical components The Following Library Files are used in WOKWI are,

- DHT22 sensor
- IR sensor
- DHT sensor library for ESPx
- WiFi
- ThingSpeak

Wokwi IoT - ThingSpeak IoT

wokwi.com/projects/305568836183130690

WOKWI SAVE SHARE esp32-micropython-ssd1306 by urish Docs SIGN UP

main.py diagram.json ssd1306.py Library Manager Simulation

```

14 # ThingSpeak API key
15 api_key = "R4PC4Hw83LV903KE"
16 # ThingSpeak server
17 ts_server = "api.thingspeak.com"
18 channel_number = 2319907
19
20 # Connect to WiFi
21 def connect_wifi():
22     sta_if = network.WLAN(network.STA_IF)
23     if not sta_if.isconnected():
24         print("Connecting to WiFi...")
25         sta_if.active(True)
26         sta_if.connect(ssid, password)
27         while not sta_if.isconnected():
28             pass
29         print("Connected to WiFi")
30
31 # Function to read and return temperature, humidity, and IR data
32 def read_sensors():
33     dht22 = dht.DHT22(dht_pin) # Create the DHT22 object
34     dht22.measure()
35     temp, hum = dht22.temperature(), dht22.humidity()
36     ir_data = ir_pin.value() # Read IR data
37     return temp, hum, ir_data
38
39 # Send data to ThingSpeak
40 def send_to_thingspeak(temp, hum, ir_data):
41     base_url = "http://{}update?api_key={}&field1={}&field2={}&field3={}".format(
42         ts_server, api_key, temp, hum, ir_data)
43     response = urequests.get(base_url)

```

Simulation

load:0x40078000,len:14876  
 ho 0 tail 12 room 4  
 load:0x40080400,len:3368  
 entry 0x400805cc  
 Connecting to WiFi...

Type here to search 31°C 22:15 26-10-2023

Wokwi IoT - ThingSpeak IoT

wokwi.com/projects/305568836183130690

WOKWI SAVE SHARE esp32-micropython-ssd1306 by urish Docs SIGN IN

main.py diagram.json ssd1306.py Library Manager Simulation

```

14 # ThingSpeak API key
15 api_key = "R4PC4Hw83LV903KE"
16 # ThingSpeak server
17 ts_server = "api.thingspeak.com"
18 channel_number = 2319907
19
20 # Connect to WiFi
21 def connect_wifi():
22     sta_if = network.WLAN(network.STA_IF)
23     if not sta_if.isconnected():
24         print("Connecting to WiFi...")
25         sta_if.active(True)
26         sta_if.connect(ssid, password)
27         while not sta_if.isconnected():
28             pass
29         print("Connected to WiFi")
30
31 # Function to read and return temperature, humidity, and IR data
32 def read_sensors():
33     dht22 = dht.DHT22(dht_pin) # Create the DHT22 object
34     dht22.measure()
35     temp, hum = dht22.temperature(), dht22.humidity()
36     ir_data = ir_pin.value() # Read IR data
37     return temp, hum, ir_data
38
39 # Send data to ThingSpeak
40 def send_to_thingspeak(temp, hum, ir_data):
41     base_url = "http://{}update?api_key={}&field1={}&field2={}&field3={}".format(
42         ts_server, api_key, temp, hum, ir_data)
43     response = urequests.get(base_url)

```

Simulation

01:58.494 99%

clk\_drv:0x00,q\_drv:0x00,d\_drv:0x00,cs0\_drv:0x00,hd\_drv:0x00,wp\_drv:0x00  
 mode:DIO, clock div:2  
 load:0x3fff0030,len:4728  
 load:0x40078000,len:14876  
 ho 0 tail 12 room 4  
 load:0x40080400,len:3368  
 entry 0x400805cc  
 Connecting to WiFi...  
 Connected to WiFi  
 Temperature: 24.0°C, Humidity: 40.0%, IR Data: 1  
 Data sent to ThingSpeak successfully

Type here to search 31°C 22:05 26-10-2023

# THINGSPEAK

ThingSpeak is our chosen server for your air quality Monitoring project.

## 1.Set Up ThingSpeak Account:

If you haven't already, create an account on ThingSpeak

(<https://thingspeak.com/>).

## 2.Create a ThingSpeak Channel:

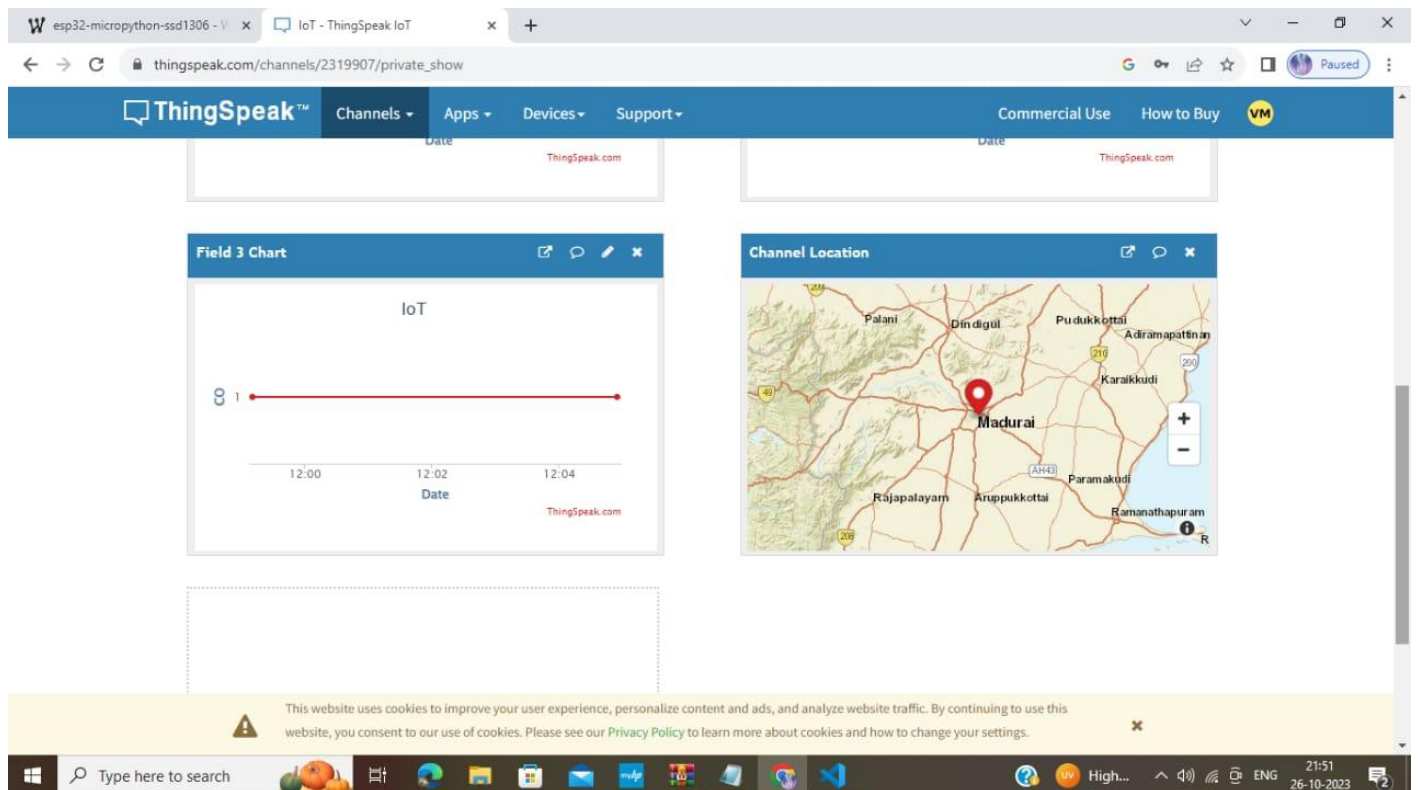
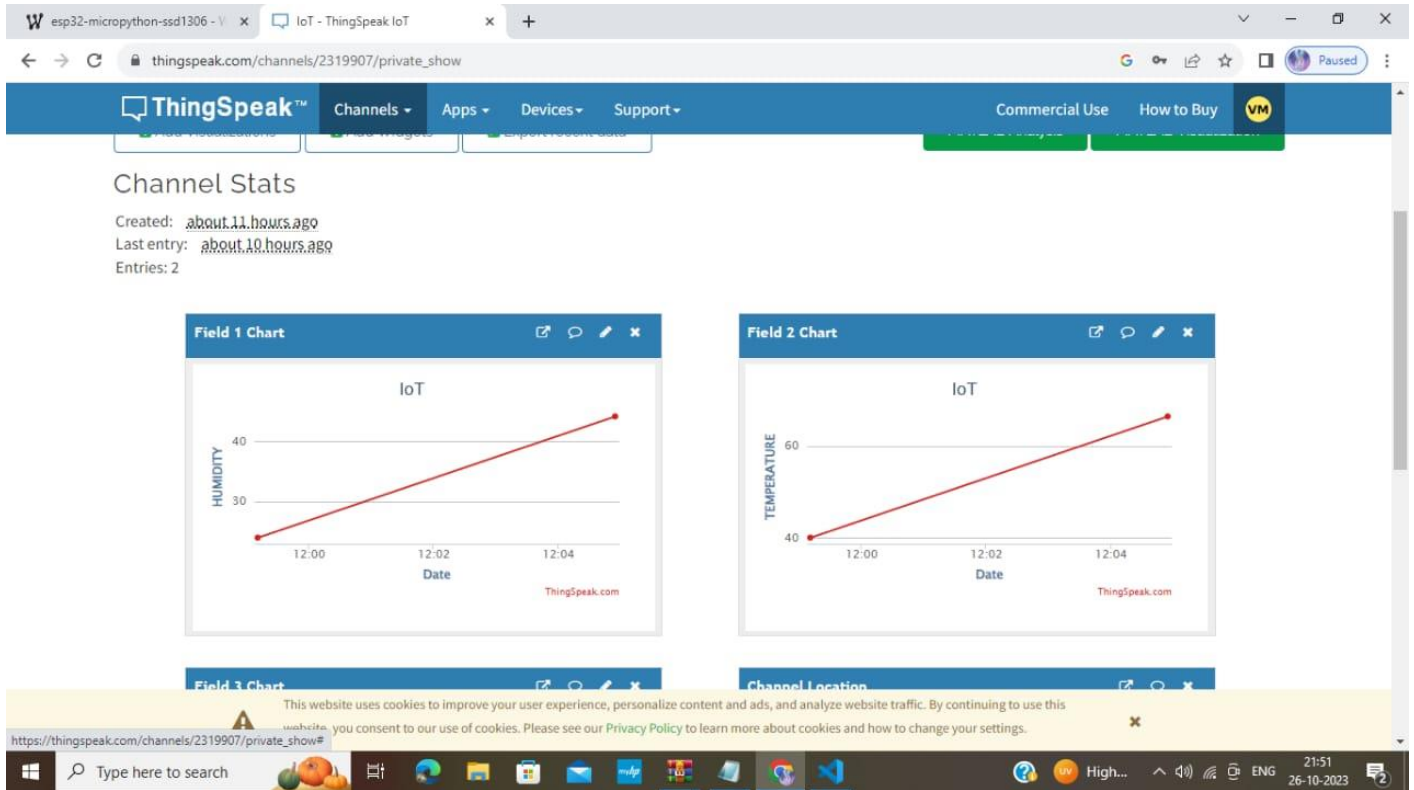
In your ThingSpeak account, create a new Channel. This channel will be Used to store the data From your air quality monitoring system.

## 3.Note Your API Key:

In the channel settings, you'll find an API Key. You will need This key To send data to your ThingSpeak channel

## 4.Integrate ThingSpeak in Your

In your Python code running on Wokwi, use the Urequests library to Send data to ThingSpeak. You can Construct a URL with your API Key and the Data you want To Python Code to send.





## Implications for Public Health and Environment

IOT air quality monitoring has significant implications for public health And the environment. It empowers communities and policymakers to:

- **Protect Public Health:** By providing accurate, real-time data, IOT air Quality monitoring helps alert individuals and authorities to potential health Risks, allowing for timely interventions to reduce exposure to harmful Pollutants.
- **Support Environmental Policies:** The data generated by IoT systems can Inform the development of evidence-based policies aimed at reducing Pollution, mitigating climate change, and promoting sustainable urban Planning.
- **Raise Awareness:** Accessible air quality information increases public Awareness and encourages individuals to make informed choices, such as Adjusting their daily routines to reduce exposure to pollution.

# BENEFITS OF IOT AIR QUALITY MONITORING

- ❖ **Real-Time Monitoring:** IOT allows for continuous, real-time monitoring of Air quality, enabling timely responses to changing conditions and helping Authorities implement mitigation measures swiftly.
- ❖ **Data Accuracy:** IOT sensors provide highly accurate data, reducing the Margin of error associated with traditional monitoring methods. This accuracy Is crucial for making informed decisions and understanding the true impact of Air pollution.
- ❖ **Cost-Efficiency:** IOT technologies can be more cost-effective than Traditional air quality monitoring stations, making it feasible to deploy a wider Network of sensors, especially in areas where comprehensive monitoring was Previously challenging.
- ❖ **Customization:** IOT air quality monitoring systems can be tailored to Specific needs, such as urban areas, industrial zones, or indoor spaces. This Adaptability ensures that the data collected is relevant to the context.

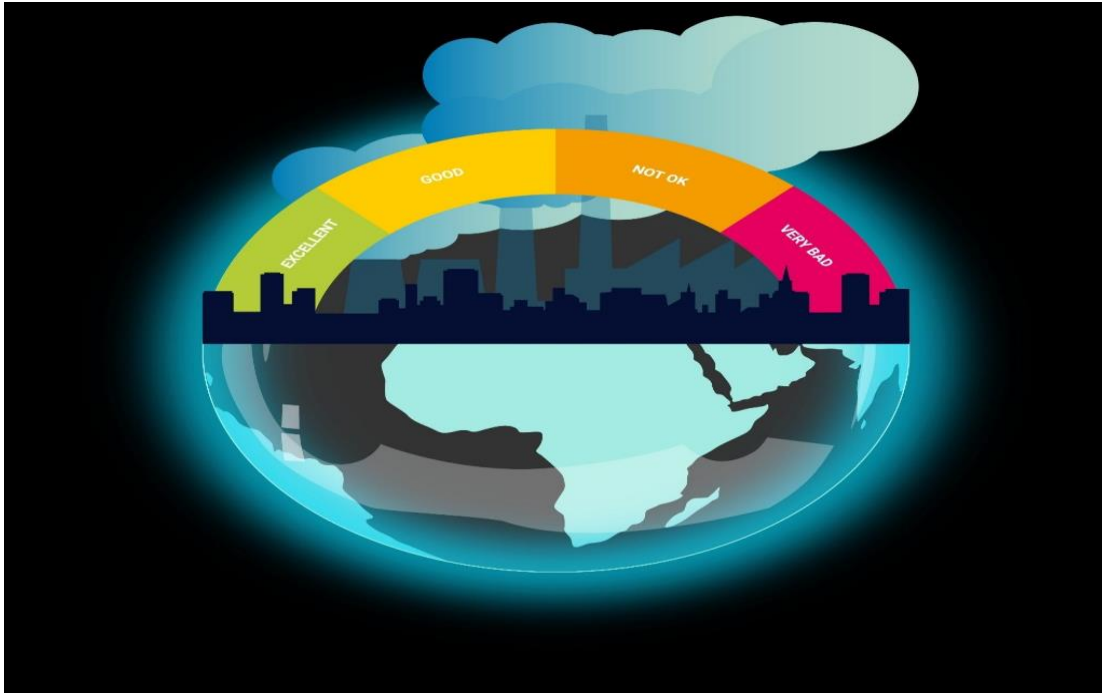
# PUBLIC AWARENESS ABOUT AIR QUALITY AND HEALTH IMPACTS

## Purposes of Real-time air quality data



We can gather this information through the use of air quality monitoring equipment. This equipment can measure the levels of various pollutants in the air and transmit that data in real-time to a central location or app. The data helps in alerting individuals and organizations to potential health hazards.

# CONCLUSION



In conclusion, an IoT-based air pollution monitoring system is a revolutionary solution that can provide accurate and real-time data about the air quality in a particular area. It can help identify the sources of pollution and take necessary measures to reduce it, protecting the environment and human health.