IoT based Weather Monitoring System

COEN 6711-AA Summer 2024

Department of Electrical and Computer Engineering

Concordia University

Siddhi Dilip Harischandrakar
Electrical and Computer Engineering
Concordia University
Montreal, Canada
Student ID: 40270218
hsiddhi47@gmail.com

Atharva Mahajan
Electrical and Computer Engineering
Concordia University
Montreal, Canada
Student ID:40270127
atharvam09@gmail.com

Athul Thomas
Electrical and Computer Engineering
Concordia University
Montreal, Canada
Student ID:40269294
thomasathul03@gmail.com

***Abstract*—In this project, we designed and implemented an IoT based weather monitoring system. It is designed to present real-time monitoring of environmental parameters such as temperature, humidity, air quality, and other weather-related conditions. Multiple sensors like BMP180, MQ135,DHT11 and MCP3008 ADC is used to find the current weather conditions. An ARM-based microprocessor is used to interface with these sensors, facilitating data collection and processing. The collected data is transmitted wirelessly using communication protocols like I2C, SPI and WiFi to a cloud platform for storage and analysis. ThingSpeak is the cloud platform for storing sensor data, allowing for continuous monitoring and data visualization.Some of the major key contributions of this project are focused on acquiring, and transmitting in real-time the data obtained in real-time about weather. Our report is our own work and each member of the group has contributed to the project's development.**

***Index Terms*—internet of things, monitoring weather, environmental sensors, Arm Cortex A53, Thing-speak, real-time data visualization.**

I. Introduction

The Internet of Things is a combination of digital objects that comprises of sensors, software, and electronic parts that have the capacity to connect with one another along with end-users.It is advancing so fast because the different kinds of information, communication technologies and the internet converge. In sectors like agriculture and urban planning understanding weather patterns and predicting weather patterns is a difficult task. The main limitation of traditional methods is that they struggle with timely and accurate data collection. The presence of IoT technology has transformed this filed by enabling development of wireless sensor network which continuously gathers and transmits data from various locations.

Current weather monitoring systems may be limited with respect to realtime data acquisition, data accuracy and accesibility. Our project focuses on these limitations by developing a IoT based system which is capable of capturing, proceesing and transmitting real time weather data. It also aims to over come the limitations of traditional methods.

The main objectives of this project are to design and implement an efficient IoT based weather monitoring system that provides real time monitoring, transmission and visualization of temperature, humidity, air quality and atmospheric pressure.

II. Literature Review

The literature review for this project consist of four different reasearch papers. Firstly, Mohapatra and Subudhi (2022) describe a budget-friendly IoT-based weather monitoring system in an IEEE Consumer Electronics Magazine. The underlying message of their work is their ability to measure the parameters of the environment in real-time through the IoT-based hardware and ARM-based microprocessor. ThingSpeak is the WiFi data transport and storage platform which provides the continuous collection and display of data for diverse applications like, for example, agriculture and urban planning [1]. Secondly, Shah and Mishra (2016) tackled a city-environment monitoring project that was founded on the concept of Internet of Things (IoT) was presented by the 2016 International Conference on Internet of Things and Applications (IOTA). The way to do their approach is by inclusion of various sensors into the network that will be used to monitor the most important environmental conditions in the city, for example pollution and CO2 levels, showing the development of IoT technology and its role in the city's sustainability and resilience [2].

Furthermore, Sindhwani.(2022) suggest a ThingSpeak environmental monitoring based on IoT which is very encapsulated in their research paper that was published in the Seventh International Conference on Parallel, Distributed, and Grid Computing (PDGC).Cloud platforms for storage and analysis like the integration of cloud platforms and data storage technology, for example, are helpful to deploy, avail remote access, and manage at a larger scope of environmental monitoring apps [3]. Finally, Girija.(2018) have written an article about an IoT-based weather monitoring system in the International Journal of Engineering Research that applies the technique of IoT for cloud computing in real-time data acquisition and visualization. The advantages of this setup are primarily the speed and accuracy of information which is now possible due

to the improvement of sensor technology and data analytics. Topics in this paper shows how modern technology affected the environment and its analysis. [4].

Overall, this series of papers collectively play a role in boosting the field of IoT-based weather monitoring systems, demonstrating the importance of different methods, techniques, and applications to the improvement of environmental monitoring and management practices.

III. Methodology

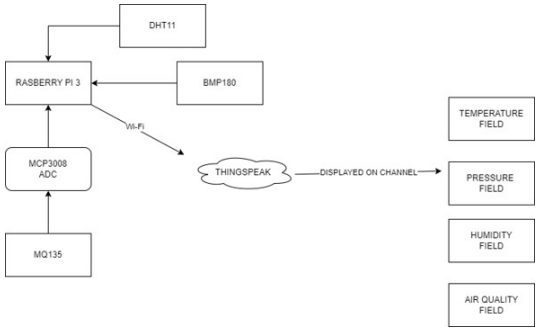

Fig. 1. Block Diagram

The weather monitoring system is designed to collect 4 weather related parameters using 3 different sensors. The figure below shows a brief design of our monitoring system. Our target parameters for the project were temperature, humidity, air quality and pressure. the basic outline of our project is that three different sensors are connected to a microprocessor. The data collection is initiated and is sent over a cloud platform using wifi. Analysis on weather conditions is done on the cloud. The cloud helps visualize the results making it easier for analysis.

IV. Model Description

A. Components of Model

Components used for IoT based Weather Monitoring systems are Raspberry Pi 3 model B, DHT11 temperature and humidity sensor, BMP180 pressure sensor, MQ135 gas sensor and MCP3008 ADC.

B. Detailed explanation of each component with code snippets

- Raspberry pi 3 Model B- The microprocessor used for our project is a Raspberry pi 3 model b. It operates on a arm cortex a53 mcu. The raspberry pi has 40 GPIO pins and it supports i2c and spi communication protocols. It also has inbuilt wifi eliminating the need of a wifi module.
- Dht11-temperature and humidity sensor- This sensor is used to find temperature and humidity of its surrounding environment. It is a digital sensor module. The difference between a sensor and sensor module of dht11 is the addition of a pull up resistor in the module. The pull up resistor ranging between 330 ohms to 10k ohms is important due to its bidirectional data line. Adafruit dht11 sensor library is openly available across all programming languages. The library is a popular choice for all developers due its high accuracy.

Fig. 2. DHT11

```
#DHT11
DHT_SENSOR = Adafruit_DHT.DHT11
DHT_PIN = 4
```

Fig. 3. DHT11 Code Snippet

```
#DHT11
humidity, temperature = Adafruit_DHT.read(DHT_SENSOR, DHT_PIN)
print("Temperature is ",temperature, "C") #Temperature is in degree Celcius
print("Humidity is ",humidity, "%")
```

Fig. 4. DHT11 Code Snippet

- Bmp180- it is an atmospheric pressure sensor which can also detect surface temperature and altitude. The interfacing of the sensors was done using i2c communication protocol. In our case the rasp pi is the master and bmp180 is the slave. The communication took place over two pins that is SDA which carried the data and SCL which carried the clock. There are multiple libraries available for the sensor across all programming languages. Adafruit Bmp180 library is the most easy and reliable to implement [6].

Fig. 5. BMP180

- I2c communication- The master (raspberry pi) puts SDA line low while SCL is high, signaling the start of data transfer. The raspberry pi sends the 7-bit address of the BMP180 followed by a read/write bit- 0xEE for write and 0xEF for read. The BMP180 responds by puting SDA line low during the ninth clock pulse. Data is exchanged between the raspberry pi and BMP180 carries out by two operations- Write Operation: Raspberry pi sends data to the BMP180. -Read Operation: The BMP180 sends data

to raspberry pi. Each byte of data is followed by an ACK bit. The raspberry pi puts the SDA line high while SCL is high, signaling the end of data transfer.

```
#BMP180
pressure = bmpsensor.readBmp180()
print("Pressure is ",pressure) # Pressure in Pa
```

Fig. 6. BMP180 Code Snippet

Fig. 7. MQ135

- Mq135- it is a gas sensor that can detect various types of gases such as nitrogen oxygen alcohol carbon dioxide etc. It is an analog sensor hence it needs an ADC to transfer data to a gpio pin of the rasp pi. It has a wide detection range and has fast response with high accuracy. Adafruit MQ135 python library is an open-source library.

```
def read_gas_concentration():
    # Read the analog value
    adc_value = chan.value
    voltage = chan.voltage

    gas_concentration = (voltage / 3.3) * 100.0

    return gas_concentration

while True:
    #MQ135
    gas_concentration = read_gas_concentration()
    print("Gas Concentration: {:.2f}%".format(gas_concentration))
```

Fig. 8. MQ135 Code Snippet

- MCP3008 ADC- It is an 8 channel 10 bit analog to digital converter. It is mainly used to convert analog data signal to digital data signal. It supports spi communication protocol. The digital output code calculation is based on the formula [7]

$$Digital\ Output\ Code = \frac{1024 \times V_{IN}}{V_{REF}}$$

Where:

V_{IN} = analog input voltage

V_{REF} = analog input voltage

Fig. 9. Formula used by MCP3008 ADC

- The MCP3008 samples the analog signal by acquiring it on an internal sample/hold capacitor for 1.5 clock cycles, starting from the first rising edge of the serial clock (CLK) after the Chip Select (CS) pin is pulled low. Once sampled, the SAR mechanism converts this signal to a digital output by comparing the input voltage with a reference voltage (VREF) in a binary search manner, sequentially determining each bit. This involves charging the sample/hold capacitor to the input voltage and discharging it through binary-weighted capacitors to produce a digital value proportional to the input voltage. The resulting 10-bit digital code is then shifted out serially through the DOUT pin on each falling edge of the clock signal [7].

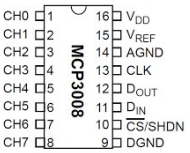

Fig. 10. MCP3008 ADC

```
# Set up SPI
spi = busio.SPI(clock=board.SCK, MISO=board.MISO, MOSI=board.MOSI)
cs = digitalio.DigitalInOut(board.D5)  # Chip select on GPIO5 (Pin 29)
mcp = MCP3008(spi, cs)
```

Fig. 11. SPI Code Snippet

To initiate communication, the CS pin is pulled low; if it was initially low at power-up, it must be pulled high and then low again. Communication begins with a start bit followed by configuration bits sent via the DIN pin, which include Start Bit: First clock cycle with CS low and DIN high. SGL/DIFF Bit: Indicates single-ended or differential conversion. Channel Selection Bits (D0, D1, D2,...): Select the input channel (in our case channel 0). Sampling of the analog input starts on the fourth rising clock edge after the start bit and ends on the fifth falling clock edge. The conversion result is then shifted out bit by bit through the DOUT pin on each falling clock edge, providing a 10-bit digital representation of the analog signal.

V. Experimental Setups

A. Experimental Setups in Software

Software setup- Thingspeak is an Iot Cloud platform used for visualization and analysis of real time data. The steps to create a new channel and get graphs are as follows.

- Go to the website and click on sign in at the top right corner, Sign in to the MathWorks account.
- Once logged in navigate to channels and click on new channel. Fill in channel details which include name, description and field labels. And save the channel.
- navigate to API keys where you'll find read write keys of your channel. Using the write key you will send request to thingspeak to send data.
- on successful integration, start collecting sensor data. Add a response statement in the code to update the channel fields in real time.

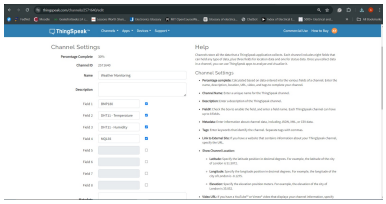

Fig. 12. Setting up a ThinkSpeak Channel

```
#Thingspeak Credentials
channel_id = 2571640
write_key="2LPV4J63J5Q19J3N"

#Channel Ids
channel = thingspeak.Channel(id=channel_id, api_key = write_key)
```

Fig. 13. Setting up a ThinkSpeak Credentials

- By default, line graphs will be created with timestamps and date. If you wish to visualize the data using different tools, then you need to navigate to MATLAB analysis and create a MATLAB analysis.

```
#Thingspeak channel update
reponse = channel.update({
    'field1' : pressure,
    'field2' : temperature,
    'field3' : humidity,
    'field4' : gas_concentration
    })
```

Fig. 14. Setting up a ThinkSpeak Channel Update

B. Experimental Setups in Hardware

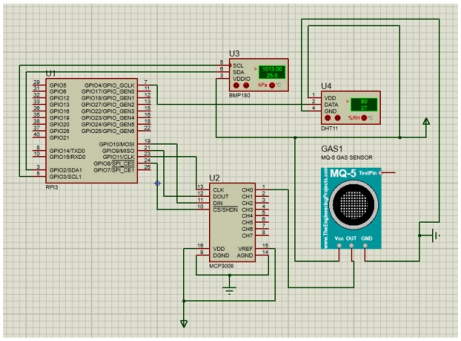

Fig. 15. Schematic Diagram

The figure above shows the schematic of our system. The microprocessor used is a raspberry pi 3 model b. Dht11 temperature and humidity sensor is connected using GPIO4. As it is a digital sensor the data is transmitted and received using the same data signal. BMP180 pressure sensor is connected to the I2c bus of the raspberry pi 3. SDA and SCL pins that is GPIO2 and GPIO3 respectively are used for I2c communication.

MQ135 is an analog sensor hence it is connected to channel 0 (pin1) of the ADC (MCP3008). The ADC supports spi communication protocol hence it is connected to the raspberry pi using 4 pins. Din pine is connected to GPIO10(MOSI) and Dout is connected to GPIO9(MISO). Clock is connected to GPIO11, and slave select is connected to GPIO8. The ADC is connected to a power supply of 3.3v. Vdd and Vref are connected to the power supply while AGND and DGND are connected to ground. All the sensors are connected to a 5v power supply.

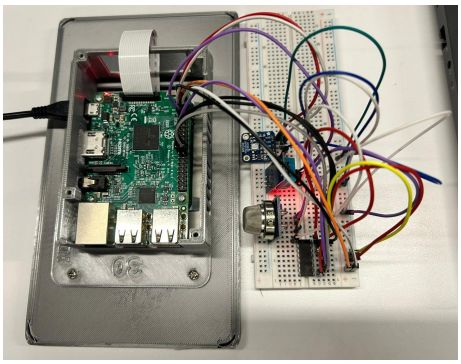

Fig. 16. Hardware Connections

C. Tools and platforms used

- Thonny- The python IDE used on Raspbian OS.

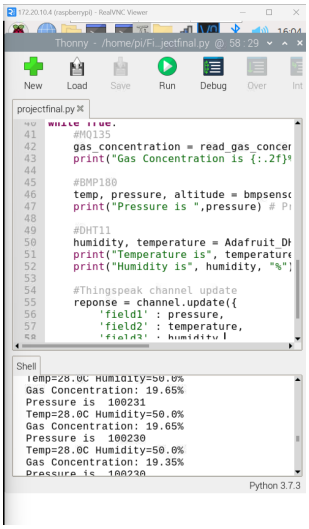

Fig. 17. Outputs on Thony

- Proteus 8 professional- software used for the simulation of the system.
- Libraries- the libraries include all sensor libraries along with thingspeak library and other python pre-requisites.
- Setting up the environment on Raspbian OS-

The I2c and spi communication is disabled by default on the raspberry pi. To enable it, we need to follow these steps:- Using the command sudo rasp-config, we can open the configuration menu of the raspberry pi. Navigate to interfacing options and enable I2c and SPI communication protocols

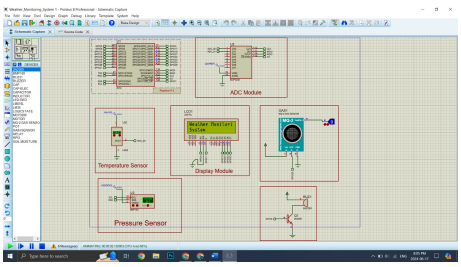

Fig. 18. Simulation Circuit Diagram

individually. To check if the I2c connection has been enabled use the command sudo i2cdetect -y 1. You should get an output 77.

- Connecting to Real VNC server-viewer

Open raspberry pi configuration menu and navigate to interfacing options. Select VNC and enable it. Use ifconfig command to find the internal ip address of the raspberry pi. Now download real vnc server-viewer on your laptop. Once the software is successfully installed, type the internal ip address of your raspberry pi in the search option and click enter. A prompt will pop up where you need to put in your username and password of the raspberry pi. Once logged in the raspberry pi screen will be mirrored on your laptop and you can control everything from you're your laptop.

VI. Results

A. Experimental Results and graphical representation

Fig. 19. Results on ThinkSpeak

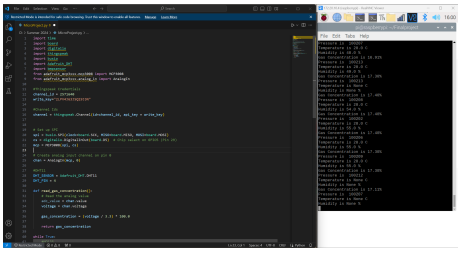

Fig. 20. Results on Raspberry Pi Screen

B. Statistical analysis

1) Cloud Formation and Rain Prediction:

- High Humidity + Low Pressure + Moderate Temperature: Likely to indicate cloudy conditions with potential rain.

1) Clear and Calm Weather:

- Low Humidity + High Pressure + Moderate Temperature: Typically suggests clear skies and calm weather.

1) Thunderstorm Potential:

- Low Pressure + High Humidity + High Temperature: This could indicate the potential for thunderstorms or severe weather.

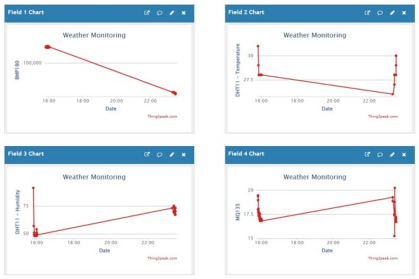

Fig. 21. Results on ThinkSpeak from 4PM-11PM

Pressure	Temperature	Humidity	Air Quality
100207	28	48	16.910048
100212	28	54	17.007706
100209	28	51	17.007706
100212	28	50	17.007706
100207	28	49	17.007706
100213	28	49	16.81239

Fig. 22. Sensor data recorded at 4pm

Pressure	Temperature	Humidity	Air Quality
99617	26	73	18.866255
99618	27	70	17.399863
99614	25	69	17.49752
99610	26	75	20.233463
99609	28	74	18.768597
99604	28	67	16.714733

Fig. 23. Sensor data recorded at 11pm

The standard atmospheric pressure at the campus is 100,325 Pa. The pressure drops at night to 99,000 Pa. The temperature remains moderate around 25-28°C. There is an increase in humidity at night compared to the day. The CO2 levels range from 10% to 16% in an indoor environment setting. Based on this data, we can infer that at 4 PM, the weather indication is clear and calm, as the experiment was conducted on campus. At 11 PM, the combination of high humidity, low pressure, and moderate temperature suggests that the weather was cloudy with the potential for rain.

VII. Comparison with Existing Solutions

Comparison with existing solutions and our project is appended in appendix B, please refer to appendix B for it. We could not add table here because it was printing horizontal so we decided to put it in appendix.

VIII. ARM Processor Description

Raspberry Pi 3 model B is third generation raspberry pi with a dimension of 85x56x17 mm. The processor used is 1.2GHz Quad-Core ARM Cortex-A53 with Broad-com BCM2387 chip-set. It has wireless LAN and Bluetooth connectivity; 802.11 b/g/n Wireless LAN and Bluetooth 4.1 (Bluetooth Classic and LE) respectively. GPU is Dual Core VideoCore IV® Multimedia Co-Processor which provides Open GL ES 2.0, hardware-accelerated OpenVG, and 1080p30 H.264 high-profile decode. GPU is capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs with texture filtering and DMA infrastructure. This model has memory of 1GB. The operating system boots from Micro SD card, running a version of the Linux operating system or Windows 10 IoT [5].

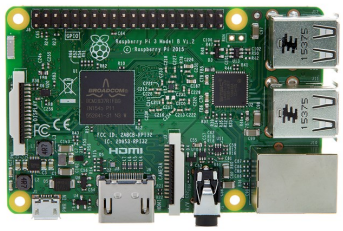

Fig. 24. Raspberry Pi 3 Model B

Raspberry Pi 3 has multiple connectors. Each connector has a specific output like ethernet with 10/100 BaseT ethernet socket, HDMI (rev 1.3 & 1.4 Composite RCA (PAL and NTSC) for video output, audio output with 3.5 mm audio output jack and HDMI USB 4 x USB 2.0 Connector. 40 GPIO pins with 2.54 mm (100 mil) expansion header: 2x20 strip Providing 27 GPIO pins as well as +3.3 V, +5 V and GND supply lines, 15-pin MIPI Camera serial interface, display serial interface 15-way flat flex cable connector with two data lanes and a clock lane and Push/pull Micro SDIO memory card slot [5].

Some of the key benefits are low cost, 10x faster processing, consistent board format and more connectivity. Raspberry Pi 3 model B is use in applications like IoT, robotics, media centre, web camera, low-), etc.c, security monitoring, Environmental sensing/monitoring (e.g. weather station), etc [5].

Raspberry Pi 3 Model B interfacing Application using DHT11:

- Hardware Setup:

Connect Vcc and GND pin of DHT11 to 3.3V pin and ground pin of raspberry pi 3 respectively.

Connect output data pin of DHT11 to GPIO pin 4 of raspberry pi 3.

- Software Setup:

Install Raspbian OS on raspberry pi 3.

Install necessary python libraries to read data from DHT11 sensor like Adafruit_DHT.

Write a python script for DHT11 and print the values of temperature and humidity.

IX. Limitations and Future work

Limitations:

- Data accuracy: Due to environmental interference or hardware malfunctions sensors can provide inaccurate data.
- Power Dependent: Most of the IoT devices are dependent on continuous power supply, which can be a challenge in remote areas.
- Connectivity Issues: For real-time data transmission reliable internet connectivity is essential.
- Data Security: IoT systems are vulnerable to cyber attacks, it compromises integrity and confidentiality of collected data.
- High Initial cost: Building and setting up an IoT facility requires a lot of capital due to cost of sensors, network architecture,etc.

Future Improvements:

- Integrate additional sensors for more comprehensive weather analysis. We can add sensors like light intensity sensor, rain gauge,Anemometer which measures wind speed, wind vane measure wind direction,etc.
- Enhance data analytics with advanced machine learning algorithms. Using machine learning algorithms require alt of data to train a simple model. There are two solutions to this.

  First one is to collect 2 to 3 months worth of data and use this data to train a LSTM machine learning model. Doing this the model will predict weather forecast for next 7 days.

  For second solution we need to calculate climate normal and climate average 5-30 years of sensor monitored data.Then use statistical tools to find average and normals so we can use it to forecast the climate.

X. Conclusion

The proposed IoT enabled weather monitoring system for monitoring temperature, humidity, atmospheric pressure and air quality has been successfully designed and implemented. Utilizing the ThingSpeak cloud platform for data transmission and visualization ensures that this information is readily accessible and comprehensible.The proposed IoT enabled weather monitoring system compares well with similar designs in [1]-[4]. Apart from sensing just temperature and humidity we added more sensing parameters like atmospheric pressure and air quality for better monitoring.

References

[1] D. Mohapatra and B. Subudhi, "Development of a Cost-Effective IoT-Based Weather Monitoring System," in IEEE Consumer Electronics Magazine, vol. 11, no. 5, pp. 81-86, 1 Sept. 2022, doi: 10.1109/MCE.2021.3136833.

[2] J. Shah and B. Mishra, "IoT enabled environmental monitoring system for smart cities," 2016 International Conference on Internet of Things and Applications (IOTA), Pune, India, 2016, pp. 383-388, doi: 10.1109/IOTA.2016.7562757.
[3] N. Sindhwani, R. Anand, R. Vashisth, S. Chauhan, V. Talukdar and D. Dhabliya, "Thingspeak-Based Environmental Monitoring System Using IoT," 2022 Seventh International Conference on Parallel, Distributed and Grid Computing (PDGC), Solan, Himachal Pradesh, India, 2022, pp. 675-680, doi: 10.1109/PDGC56933.2022.10053167.
[4] Girija C, Harshalatha H, Andreanna Grace Shires, Pushpalatha H P, 2018, Internet of Things (IOT) based Weather Monitoring System, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) NCESC – 2018 (Volume 6 – Issue 13),
[5] Raspberry Pi 3 Model B DATA SHEET
[6] BMP180 DATA SHEET
[7] MCP3008 ADC DATA SHEET

AppendixA

```
import numpy as np
import time
import board
import digitalio
import thingspeak
import busio
import Adafruit_DHT
import bmpsensor
from adafruit_mcp3xxx.mcp3008 import MCP3008
from adafruit_mcp3xxx.analog_in import AnalogIn
#Thingspeak Credentials
channel_id = 2571640
write_key="2LPV4J63J5Q19J3N"
#Channel Ids
channel = thingspeak.Channel(id=channel_id, api_key = write_key)
# Set up SPI
spi = busio.SPI(clock=board.SCK, MISO=board.MISO, MOSI=board.MOSI)
cs = digitalio.DigitalInOut(board.D5)
# Chip select on GPIO5 (Pin 29)
mcp = MCP3008(spi, cs)
# Create analog input channel on pin 0
chan = AnalogIn(mcp, 0)
#DHT11
DHT_SENSOR = Adafruit_DHT.DHT11
DHT_PIN = 4
def read_gas_concentration():
    # Read the analog value
    adc_value = chan.value
    voltage = chan.voltage
    gas_concentration = (voltage / 3.3) * 100.0
    return gas_concentration
while True:
    #MQ135
    gas_concentration = read_gas_concentration()
    print("Gas Concentration: {:.2f}%".format(gas_concentration))
    #BMP180
    pressure = bmpsensor.readBmp180()
    print("Pressure is ",pressure) # Pressure in Pa
    #DHT11
    humidity, temperature = Adafruit_DHT.read(DHT_SENSOR, DHT_PIN)
    print("Temperature is ",temperature, "C") #Temperature is in degree Celcius
    print("Humidity is ",humidity, "%")
    #Thingspeak channel update
    reponse = channel.update({
        'field1' : pressure,
        'field2' : temperature,
        'field3' : humidity,
        'field4' : gas_concentration
        })
    time.sleep(0.5)
```

Appendix B

Aspect	D.Mohapatra and B.Subudhi (2022)	J. Shah and B. Mishra (2016)	N. Sindhwani et al. (2022)	Girija C et al. (2018)	Our Project (2024)
Title	Development of a Cost-Effective IoT-Based Weather Monitoring System	IoT enabled environmental monitoring system for smart cities	ThingSpeak-Based Environmental Monitoring System Using IoT	Internet of Things (IoT) based Weather Monitoring System	IoT Based Weather Monitoring System
Focus	Cost-effective IoT weather monitoring	Environmental monitoring in smart cities	Environmental monitoring using ThingSpeak	General IoT-based weather monitoring	Real-time monitoring of environmental parameters
Methodologies	Sensors for weather parameters, cloud storage	Deployment in urban settings for monitoring	Sensor data transmission to ThingSpeak	Using IoT to gather and analyse weather data	ARM-based microprocessor, ThingSpeak for data visualization
Hardware	Various sensors for temperature, humidity, pressure	Sensors for temperature, humidity, air quality	Sensors for temperature, humidity, CO2 concentration	Not specified	Raspberry Pi 3, MQ135, BMP180, DHT11, MCP3008
Software	Cloud storage and analysis	Cloud-based data management	ThingSpeak for data aggregation and visualization	Not specified	Python, Thonny IDE, ThingSpeak
Results	Effective real-time monitoring with low-cost implementation	Improved environmental monitoring in smart cities	Effective use of ThingSpeak for real-time monitoring	General observations without specific data points	Successful real-time monitoring with detailed analysis
Temperature Range	15.6°C to 12.7°C (around 6:45 PM)	29°C (inside campus)	34.5°C to 42.2°C	Not specified	Moderate ,25-28^{o}C
Humidity Range	Highest around 5:00 AM to 6:00 AM	65 gm/mm³ (inside campus)	21.1% to 30.2%	Not specified	Increase in humidity at night
Air Quality (CO2/Carbon Monoxide)	Not specified	578 ppm (carbon monoxide)	1182 ppm to 1355 ppm (CO2)	Not specified	CO2 level at 10%-16% in indoor environment
Pressure Range	Not specified	Not specified	Not specified	Not specified	Standard atmospheric pressure at campus 100325, drop at night to 99000

Comparison with Existing solutions