A

PROJECT REPORT

ON

“Air Monitoring System Using ESP32”

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE

TE MINI PROJECT

BACHELOR OF ENGINEERING

In

ELECTRONICS AND TELECOMMUNICATION ENGINEERING

By

Akshada Gade **Exam. No:T190433028**

Swapnali Tayade **Exam. No:T190433135**

Under the Guidance of

Prof. S. S. salave

Sinhgad Institutes

Submitted to

DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION ENGINEERING

STES’S SINHGAD ACADEMY OF ENGINEERING, PUNE-411048

2022-2023

CERTIFICATE

This is to certify that the project report entitled

"Air Monitoring System Using ESP32"

Submitted By

1. Akshada Gade	T190433028
2. Swapnali Tayade	T190433135

Is a bonafide work carried out by them under the guidance of Prof.S.S.Salave and it is approved for the partial fulfilment of the requirement of **Savitribai Phule Pune University** for the Mini Project in the Third Year of Electronics and Telecommunication Engineering.

This project report has not been earlier submitted to any other institute or University for the award of any degree or diploma.

Prof.-S.S.Salave	**Prof.**	**Dr. K. P. Patil**
Project Guide	**H.O.D**	**Principal, SAOE, Pune**

Place: Pune

Date: **External Examiner**

ACKNOWLEDGEMENT

It is great pleasure for me to acknowledgment the assistance and contribution of number of individuals who helped me in developing “Air monitoring system using ESP32”project.A project is defined as a piece of work that needs skill , efforts and careful planning but during the course of project we found that it not only sharpened our logical skill but also taught us the value joint effort and hard work.

A successful project is the result of good team work , which contains not only the people who put efforts but also who guide them

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I am highly indebted to **Prof S S Salve** for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.
I would like to express my special gratitude and thanks to industry persons for giving me such attention and time.
My thanks and appreciations also go to my colleague in developing the project and people who have willingly helped me out with their abilities.

Project Group Members:

Akshada Gade Exam. No: T190433028

Swapnali Tayade Exam. No: T190433135

Abstract

This paper is written to represent a web-server based real-time weather monitoring system implemented using the ESP32 module. The system is designed to measure the weather conditions like temperature, pressure, humidity etc , in its vicinity. This system continuously provides real-time data (weather conditions) over limited or controlled areas, for example, agriculture fields, homes or a particular room, industries etc. Although we have satellites for weather monitoring but to perform research over a particular area, the data observed should be highly accurate which is not possible with satellites. To implement an IoT network we need to use a webserver to continuously store and display the real-time data which can be accessed globally. 'Thing Speak' is a web service where one can upload and access (globally) with internet-connected devices. ESP32 will be acting as an interface medium between the various sensors and the web-server. ESP32 module is also responsible for collecting, processing and then communicating the data to the server. The real-time data stored on the web server can be further used for research and analysis of weather conditions of a particular place.

Table of Contents

Chapter 1
Introduction

1.Introduction

The evolving generation of wireless technology has made human life a lot easier. Where everything is online and automatic we can easily monitor multiple things virtually from anywhere in the world. The WSN (wireless sensor networks) and Internet of things or IoT play an important role in implementing and accessing these wireless technologies. Smart homes, smart cities and smart weather monitoring systems are examples of such technologies where things are quite simpler or easier.

In this web-server based weather monitoring system, the weather data (from the surrounding environment or of a particular location) like temperature, humidity, pressure etc. is measured with the help of some sensors and then the collected data will be stored on a server after being processed by a microcontroller. Our daily activities are inseparable from weather conditions and various environmental factors. The real-time data collected can be used in research and analysis and the results can be helpful in human life and for improving environmental conditions as well.

The Internet of Things is a system made up of multiple inter-related computing devices. The main factor **'things'** in IoT is designated to a component that is capable of communicating data over a network (IoT), the thing could be animals, a digital machine, a sensor, a human being etc. Each component of the Internet of Things network is given an individual or a distinct identity and the ability to communicate data or information over a wireless network that is too without the intervention of a human or a computer [8].

An interface medium capable of collecting, controlling, and communicating data among transmitter and recipient electronic equipment or servers is required to build the IoT network[9].

The ESP32 microcontroller series was developed by Espressif Systems. This module (the ESP32) includes a 2.4GHz Wi-Fi chip, memory, a 32-bit Tensilica microcontroller, an antenna, peripheral interfacing abilities, power management modules, and more. This ESP32 module is excellent for the Internet of things because of all of its technological and infrastructural aspects [10].

DHT11 and BMP180 sensors are used to collect the data from their surrounding environment and then communicate the data to the ESP32 module over a particular protocol [11].

The application of this weather monitoring system can also play an important role in the field of agriculture[12] to increase productivity, research application, and reducing manpower (by reducing the need to manually monitor the field status). Sometimes in a particular agricultural zone that is hazardous for a human beings, it is quite difficult to manually (offline) monitor the environment or weather conditions. In such cases, this web server based or online weather monitoring system can be of great importance.

1.1Motivation:

Air pollution is a growing concern globally, and it has serious implications for public health and the environment. Air monitoring systems can provide valuable information about the quality of the air we breathe, and can help us take steps to improve it. The use of an ESP32 microcontroller and the Arduino IDE provides a cost-effective and efficient way to build an air monitoring system.

Here are some reasons why an air monitoring system using ESP32 and Arduino IDE is a great idea:

1. Real-time monitoring: The system can monitor air quality in real-time and provide immediate alerts when pollution levels exceed safe limits.

2. Low cost: ESP32 is an affordable and widely available microcontroller, and the Arduino IDE is an open-source platform that is free to use. This makes the system accessible to a wide range of people and organizations.

3. Ease of use: The Arduino IDE provides a user-friendly interface that simplifies programming and makes it easy for beginners to get started. The ESP32 is also easy to use and has a wide range of built-in functionalities.

4. Wireless connectivity: The ESP32 comes with built-in Wi-Fi and Bluetooth capabilities, making it easy to connect to the internet and other devices. This enables the air monitoring system to send data to a remote server or other devices for analysis.

5. Customizability: The open-source nature of the Arduino IDE allows for customization of the air monitoring system to meet specific needs and requirements.

Overall, an air monitoring system using ESP32 and Arduino IDE is an effective and affordable solution for monitoring air quality and improving public health.

1.2 Problem Statement

Indoor air pollution and poor urban air quality are listed as twoof the world's worst toxic pollution problems. The quality of the air we breathe is estimated to become nonbreathable in afew hundred years. Right awareness should be given to thepeople around the world to rethink their decisions that affectthe air and pollute its composition making it harmful to us.There exists no universal system or device that can workefficiently in all sorts of environments from homes, roads,nuclear facilities to hospitals. The main problem arises due topoor monitoring of air pollution due to miscellaneousinteractions, limited protocol standardization, security of datastorage and complex identification systems to access data. Thewhole air monitoring process is under scrutiny in the case ofurban areas. Hence to overcome all these problems we areproposing the Air monitoring system

1.3 Objectives of the proposed work

We have considered and studied various existing systems that have been developed to solve this issue. With the system we have proposed, we have tried to reduce the overall cost by using better alternatives, giving us almost accurate readings. Kumar S et al [4] have proposed a system in which Raspberry pi is playing the major node controlling role. The sensors that they have used in their system are used for measuring particulate matter, temperature, humidity, pressure, CO, and CO2. Since Raspberry pi does not have analog-to-digital converter (ADC) pins on board, additional ADC is required to be connected. Here, they have interfaced it with Arduino Uno that is providing an effective ADC. Since they have used Raspberry pi 2 model B, it does not have an in-built WiFi adapter hence an external Wi-Fi adapter is used. Further, using the MQTT protocol, they have displayed the outputs on the IBM Watson IoT Platform. Raspberry pi 3B has an in-built Wi-Fi adapter which, if used, would comparatively lead to less circuitry. This is implemented by Gupta, Harsh et al [5] in their system. Another approach for measuring air quality is using Arduino Uno as the main unit and this is proposed by Kennedy Okokpujie et al [1] in their paper. Since Arduino does not have an in-built Wi-Fi module, an external Wi-Fi module had to be interfaced with it. This module was NodeMCU ESP8266. NodeMCU ESP8266 is a specially targeted development board for IoT-based applications and hence it can also be used as a major node controlling in a system. This is presented by Kumar A et al [6] in their paper. NodeMCU ESP8266 has just one ADC pin onboard, hence if more are required, an additional ADC chip has to be connected. One more approach for this is by using an ESP32 microcontroller. ESP32 has dual processing coreswhich makes it faster as compared to others, an in-built Wi-Fi module, and enough ADC pins. Asra Noorain F et al [7] have presented this in their paper. They have sent the sensor's values to the cloud using the Blynk Platform. As compared to the other sensors, particulate matter sensors are quite expensive. Jayaratne R et al [8], in their paper, have mentioned six low-cost PM2.5 sensors and have assessed their suitability for various applications. This helps in making an inexpensive system

Chapter 2

2.1 Literature Review

1. "An Arduino-Based Air Pollution Monitoring System" by R. A. F. Arof et al. (2017)
This study presents an Arduino-based air pollution monitoring system that uses sensors to measure air quality parameters such as carbon monoxide, nitrogen dioxide, and temperature. The system is designed to be portable and affordable, and the authors report successful results in laboratory and field tests.

2. "Development of a Low-cost Air Pollution Monitoring System Using Arduino and Raspberry Pi" by R. B. Katpally et al. (2018)
This paper describes the development of a low-cost air pollution monitoring system that uses Arduino and Raspberry Pi microcontrollers. The authors report successful measurements of particulate matter and carbon monoxide levels using the system, and they suggest that it could be useful in air quality monitoring in developing countries.

3. "IoT-based Air Pollution Monitoring System using Arduino" by A. M. Bhadke et al. (2020)
This study presents an IoT-based air pollution monitoring system that uses Arduino microcontrollers and sensors to measure air quality parameters such as particulate matter, temperature, and humidity. The authors report successful results in laboratory tests and suggest that the system could be useful in environmental monitoring and management.

4. "Real-time air quality monitoring system based on Arduino and cloud computing" by Y. Zhang et al. (2018)
This paper describes the development of a real-time air quality monitoring system based on Arduino microcontrollers and cloud computing. The authors report successful measurements of air quality parameters such as particulate matter, carbon monoxide, and ozone levels, and they suggest that the system could be useful in air quality monitoring and management.

Overall, these studies demonstrate the feasibility and effectiveness of using ESP32 and Arduino IDE for air monitoring systems. The low cost, ease of use, and customizability of these microcontrollers make them a popular choice for developing air monitoring systems.

2.2Problem Statement

The problem statement of an air monitoring system using ESP32 can vary depending on the specific application and context. However, in general, the problem that such a system aims to address is the need for accurate and reliable monitoring of air quality and environmental conditions in real-time.

Air pollution and other environmental factors can have a significant impact on human health, and there is a growing need to monitor and analyze these factors in various settings. The traditional methods of air quality monitoring are often expensive and time-consuming, and they may not provide real-time data that can be used for immediate action.

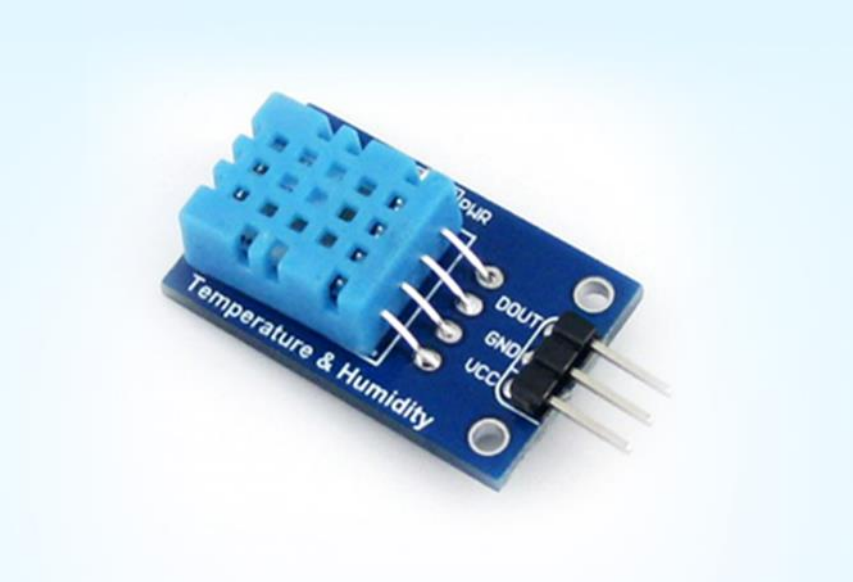
Therefore, the problem statement for an air monitoring system using ESP32 is to provide an affordable and easy-to-use solution that can accurately measure air quality and environmental

conditions in real-time. The system should be able to measure parameters such as temperature, humidity, barometric pressure, and other air pollutants such as carbon monoxide, ozone, and particulate matter. It should be designed to provide continuous monitoring of air quality data and enable users to access this data remotely using a web-based interface.

The air monitoring system should also be user-friendly and require minimal setup and maintenance. It should be able to operate in a wide range of environments and be suitable for use in both indoor and outdoor settings. Overall, the goal of the air monitoring system is to provide accurate and reliable data that can be used for informed decision-making and action to improve air quality and environmental conditions.

2.3 Hardware and software components

1) DHT11 sensor



- The DHT11 sensor is a popular digital temperature and humidity sensor that can be used with microcontrollers such as Arduino and ESP32. The sensor is relatively inexpensive and easy to use, making it a popular choice for hobbyist projects and small-scale applications.

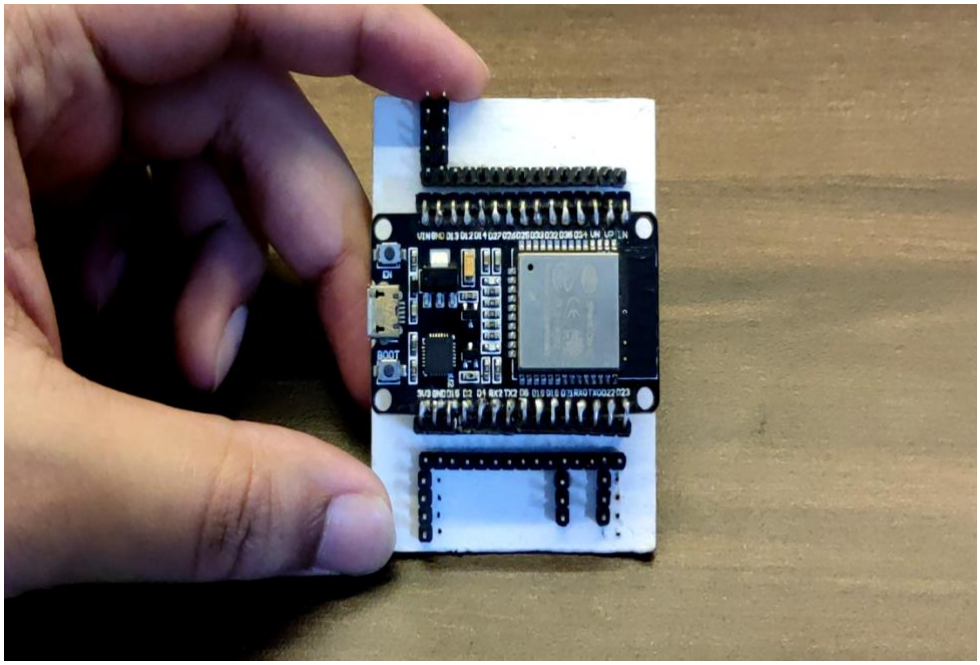
- The DHT11 sensor uses a single-wire communication protocol to transmit data to the microcontroller. The sensor can measure temperature in the range of 0°C to 50°C with an accuracy of ±2°C, and relative humidity in the range of 20% to 90% with an accuracy of ±5%. The sensor is not as accurate as some other sensors on the market, but it is sufficient for many applications.

- To use the DHT11 sensor with a microcontroller, you need to connect the sensor to the microcontroller and use a library to read the data from the sensor. The library handles the communication protocol and provides functions to read the temperature and humidity data from the sensor. The DHT11 sensor is relatively easy to use with popular microcontroller platforms such as Arduino and ESP32, and there are many tutorials and examples available online to help you get started.

- DHT11 sensor (or temperature and humidity sensor) is a sensor module used to measure humidity and temperature from its surrounding. This sensor module is responsible for monitoring the ambient temperature and humidity of a given area. An NTC (negative temperature co-efficient) temperature sensor and a resistive type humidity sensor make up the sensor. An 8-bit microcontroller is also included. The microcontroller performs ADC (analogue to digital conversion) and outputs a digital signal via the single wire protocol [13].

- Some of the technical specifications of the DHT11 sensor are:
- DHT11 sensors can also be used to create a wired sensor system with up to 20 meters of cable.
- To measure temperature and humidity, two DHT modules (DHT11 and DHT22) are available on the market. Both modules serve the same purpose, but they have different specifications. The DHT22 sensor, for example, has a wider range of temperature and humidity sensitivity. However, DHT22 is more expensive than DHT11. As a result, you can choose to use any of the modules as per your needs.

2) ESP32



- The ESP32 is a popular low-cost, low-power system-on-chip (SoC) microcontroller developed by Espressif Systems. It is designed for Internet of Things (IoT) applications and provides a variety of features, including built-in Wi-Fi and Bluetooth connectivity, low power consumption, and a dual-core processor.

- The ESP32 microcontroller features a dual-core Xtensa LX6 processor with clock speeds of up to 240 MHz. It also includes 520 KB SRAM, 448 KB ROM, and 4 MB flash memory for

program and data storage. The microcontroller provides a variety of communication interfaces, including SPI, I2C, UART, CAN, and Ethernet.

- The ESP32 microcontroller is well-suited for IoT applications due to its built-in Wi-Fi and Bluetooth connectivity. It supports 802.11 b/g/n Wi-Fi and Bluetooth 4.2, making it easy to connect to wireless networks and other devices. The microcontroller also supports a variety of security features, including SSL/TLS, WPA/WPA2, and AES encryption.

- The ESP32 microcontroller is widely used in a variety of applications, including home automation, smart agriculture, industrial automation, and wearable devices. It is also popular among hobbyists and makers due to its low cost and ease of use. The ESP32 is supported by the Arduino IDE and a variety of other development environments, making it easy to get started with programming and development.

3) BMP180

- The BMP180 is a digital barometric pressure sensor that can also measure temperature. It is manufactured by Bosch Sensortec and is a popular choice for weather monitoring, altitude sensing, and other applications that require accurate pressure and temperature readings.

- The BMP180 sensor is a very small and low-power device that communicates with microcontrollers over an I2C interface. It has a measuring range of 300 to 1100 hPa (hectopascals) with an accuracy of ±0.1 hPa, and a temperature range of -40°C to +85°C with an accuracy of ±2°C.

- The BMP180 sensor operates by measuring the difference in air pressure between the sensor and the surrounding environment. The sensor uses this pressure difference to calculate the altitude and also provides the temperature readings. The sensor is very accurate and can measure changes in altitude as small as 0.17 meters.

- To use the BMP180 sensor with a microcontroller, you need to connect the sensor to the microcontroller and use a library to read the data from the sensor. The library handles the communication protocol and provides functions to read the pressure and temperature data from the sensor. The BMP180 sensor is relatively easy to use with popular microcontroller platforms such as Arduino and ESP32, and there are many tutorials and examples available online to help you get started.

4) Thing speak web service

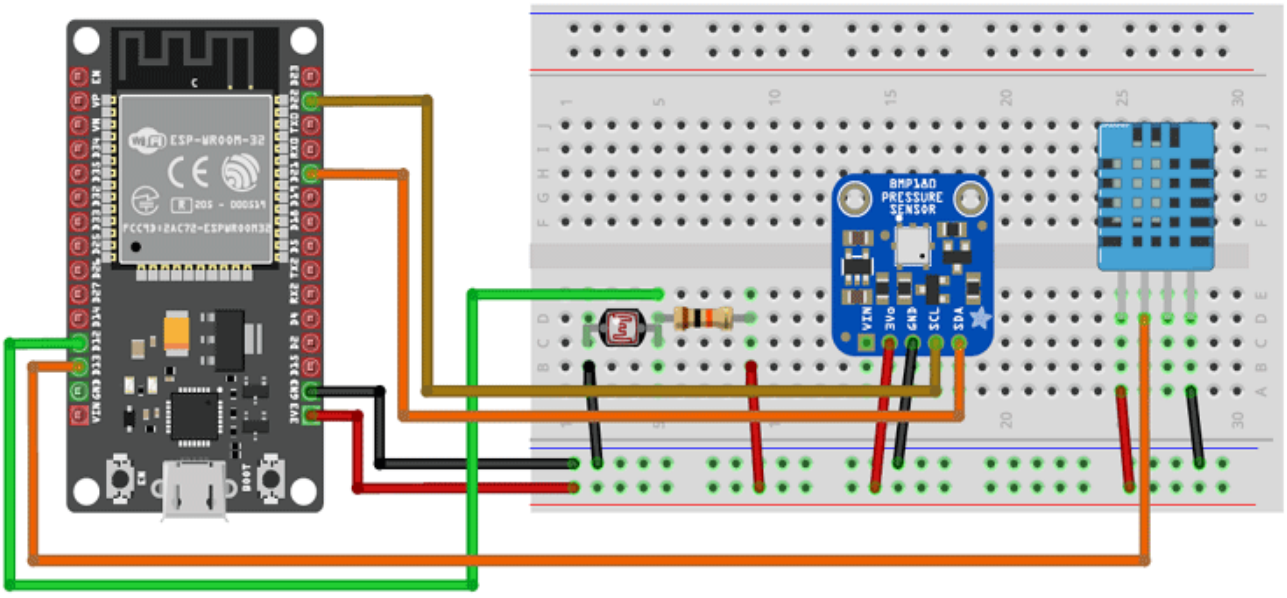
- A web server is a place where one can store data online and can access that data at any time and from anywhere in the world [16].
- A real-time data is created with the help of a web-server. There are various web services available to store real-time data for research and analysis like AWS (Amazon Web Service), Azure, Firebase etc.
- We are using the Thing Speak web service provided by Math Works which allows us to send sensor readings/data to the cloud. The Thing Speak is an open source data platform for the Internet of Things (IoT) applications. We can also visualize and act on or access the data (calculate the data) sent to the Thing Speak server from ESP32. Two different types of channels are available to store data on the Thing Speak server namely 'Public Channel' and 'Private Channel' and one can use either of the available channels to store and display data [17].
- Thing Speak is frequently used for IoT prototyping and proof-of-concept systems that require analytics[18].

5)Software used

- We need to give instructions to our ESP32 module so that it can interface, read data from DHT11 and BMP280 sensors and then finally publish the collected data to the ThingSpeak server.
- **Arduino IDE** is an integrated development environment used to write, compile and debug the program for the ESP32 module[19].

Chapter 3
Project Description

3.1 Circuit diagram



1. ESP32 Microcontroller: This is the heart of the system and serves as the main controller. It receives data from both the BMP180 and DHT11 sensors and transmits this data over Wi-Fi to a server or cloud service.

2. BMP180 Sensor: This sensor is responsible for measuring barometric pressure and temperature. It is connected to the ESP32 via the I2C interface.

3. DHT11 Sensor: This sensor measures temperature and humidity levels. It is connected to the ESP32 via a digital pin.

4. Voltage Regulator: The voltage regulator ensures a steady voltage supply to the sensors and the ESP32.

5. Power Supply: A stable power supply is required to run the system. This can be provided by a battery or an AC/DC adapter.

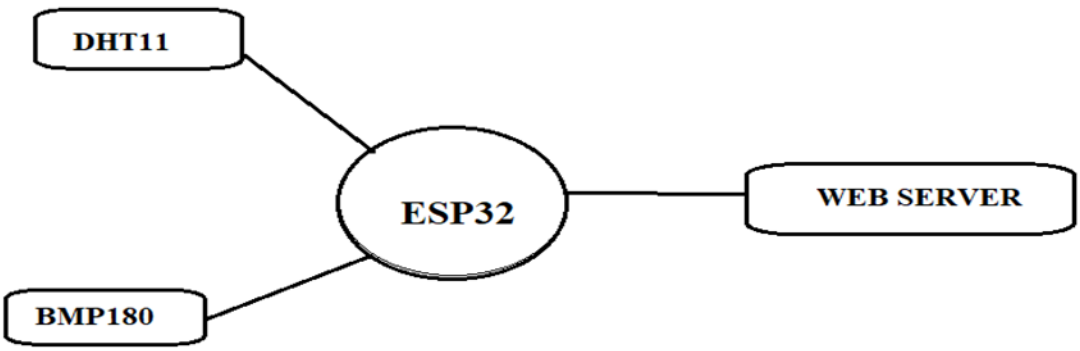
6. Wi-Fi Antenna: The Wi-Fi antenna is used to connect the ESP32 to the internet.

7. OLED Display (Optional): An OLED display can be added to the circuit to display sensor readings in real-time.

In the program code, the ESP32 reads data from both sensors using their respective libraries. The data is then transmitted over Wi-Fi to a server or cloud service for further analysis and storage. The optional OLED display can also be updated with the current sensor readings.

Overall, this circuit diagram shows a simple but effective air monitoring system that can be used in various applications such as smart homes, environmental monitoring, and agriculture.

3.2 Block Diagram



1. ESP32 Microcontroller: The ESP32 serves as the central processing unit (CPU) of the system, responsible for collecting data from the BMP180 and DHT11 sensors, processing the data, and transmitting it over Wi-Fi to a server or cloud service.

2. BMP180 Sensor: The BMP180 sensor measures the barometric pressure and temperature of the environment. It is connected to the ESP32 via the I2C interface.

3. DHT11 Sensor: The DHT11 sensor measures the temperature and humidity of the environment. It is connected to the ESP32 via a digital pin.

4. Wi-Fi Module: The Wi-Fi module enables the ESP32 to connect to the internet, allowing the system to transmit data to a server or cloud service.

5. Power Supply: The power supply provides a stable voltage to the system components. It can be a battery or an AC/DC adapter.

6. Voltage Regulator: The voltage regulator regulates the voltage supplied to the sensors and the ESP32, ensuring a stable power supply.

7. Analog-to-Digital Converter (ADC): The ADC converts analog signals from the sensors to digital signals that the ESP32 can process.

8. Real-time Clock (RTC): The RTC provides an accurate time reference for the system, enabling it to log data accurately.

9. Memory: The memory stores the data collected by the system, which can be retrieved later for analysis.

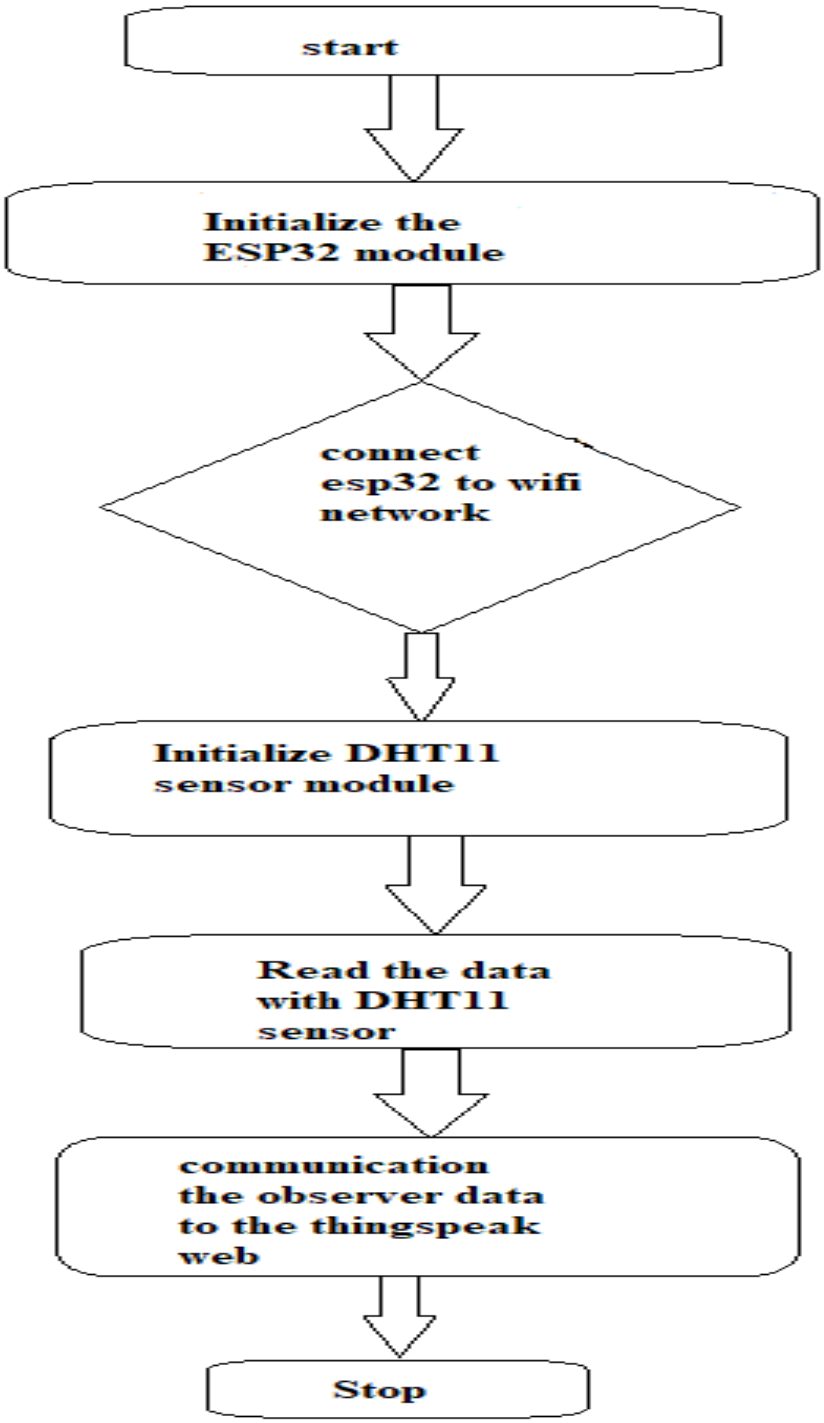
10. Server/Cloud Service: The server or cloud service receives the data transmitted by the ESP32 and stores it in a database for further analysis.

11. Web-based Interface: The web-based interface allows users to access the air quality data collected by the system from anywhere in the world.

Overall, the block diagram of an air monitoring system using ESP32, BMP180, and DHT11 shows a simple but effective system for monitoring air quality and environmental conditions. The system can be expanded with additional sensors or features to provide more comprehensive monitoring of air quality and environmental conditions. The data collected by the system can be used to improve air quality and take necessary measures to protect the environment and human health.

3.3 Process flow

The process flow of our weather monitoring system is shown below in *Chart 1*. The process starts with the initialization of ESP32 module which is acting as an interface medium between the sensor modules and the web-server. The ESP32 wi-fi module is continuously searching for the network credentials as per the instruction provided. After connecting to the internet the sensor modules will be initialized and the real-time data [22] collected from the surrounding environment will be pushed to the ThingSpeak web server[23].



Code

```
#define BLYNK_PRINT Serial
#define BLYNK_TEMPLATE_ID "TMPL36py2yjZg"
#define BLYNK_TEMPLATE_NAME "Weather Station"
#define BLYNK_AUTH_TOKEN "zxpc0mH3eXTCOqJvqhn6GZo8d1qaUPbM"
#include <WiFi.h> // importing all the required libraries
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include "Arduino.h"
```

```
#include "DHT.h"
#include "BMP085.h"
#include <Wire.h>

float temperature; // parameters
float humidity;
float pressure;
float mbar;

BMP085 myBarometer; // initialise pressure sensor

char auth[] = "zxpc0mH3eXTCOqJvqhn6GZo8d1qaUPbM"; // replace this with your auth token
char ssid[] = "Redmi"; // replace this with your wifi name (SSID)
char pass[] = "swapnali7"; // replace this with your wifi password

#define DHTPIN 5 // dht sensor is connected to D5
#define DHTTYPE DHT11     // DHT 11

DHT dht(DHTPIN, DHTTYPE); // initialise dht sensor
BlynkTimer timer;

void sendSensor() // function to read sensor values and send them to Blynk
{
  humidity = dht.readHumidity();
  temperature = dht.readTemperature();
  if (isnan(humidity) || isnan(temperature))
  {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }

  pressure = myBarometer.bmp085GetPressure(myBarometer.bmp085ReadUP()); // read pressure value in pascals
  mbar = pressure / 100; // convert millibar to pascals

  Blynk.virtualWrite(V0, temperature); // send all the values to their respective virtual pins
  Blynk.virtualWrite(V1, humidity);
  Blynk.virtualWrite(V2, mbar);
}

void setup()
{
  Serial.begin(115200);
  myBarometer.init();// sensor address  0x77
  dht.begin();
  delay(1000);
  Blynk.begin(auth, ssid, pass);
  delay(1000);
  timer.setInterval(1000L, sendSensor); // sendSensor function will run every 1000 milliseconds
}

void loop()
{
```

```
  Blynk.run();
  timer.run();
}
```

Chapter 4
Results

The results observed from the implemented weather monitoring system are shown below. We observed four different environmental conditions which include temperature, humidity, pressure . The sensor data collected by ESP32 from DHT11 and BMP180 sensors is published to the ThingSpeak web server. On ThingSpeak we have created a channel that contains four fields to store four different environmental factors temperature, humidity, pressure .

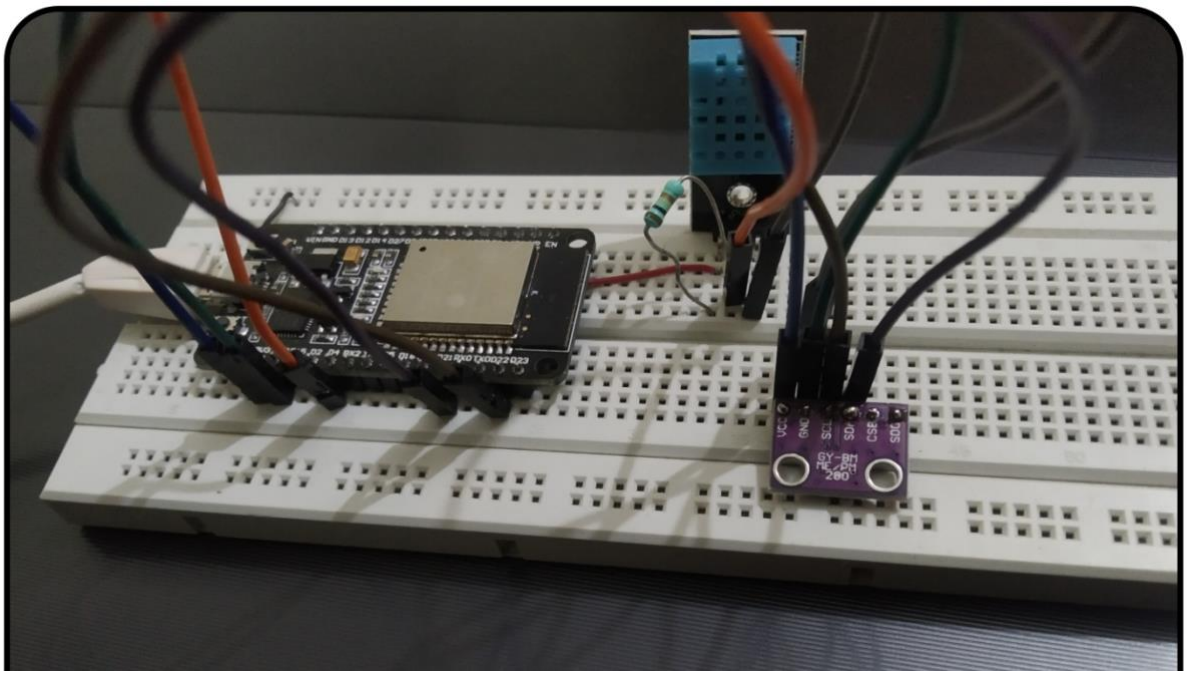


Fig 1. Hardware

we can see the ‘Field 1’ which is containing the temperature readings published or communicated from the ESP32 module and saved on the ThingSpeak server.
Similarly, fields 2, 3, and 4 are displaying the humidity, pressure and altitude respectively.

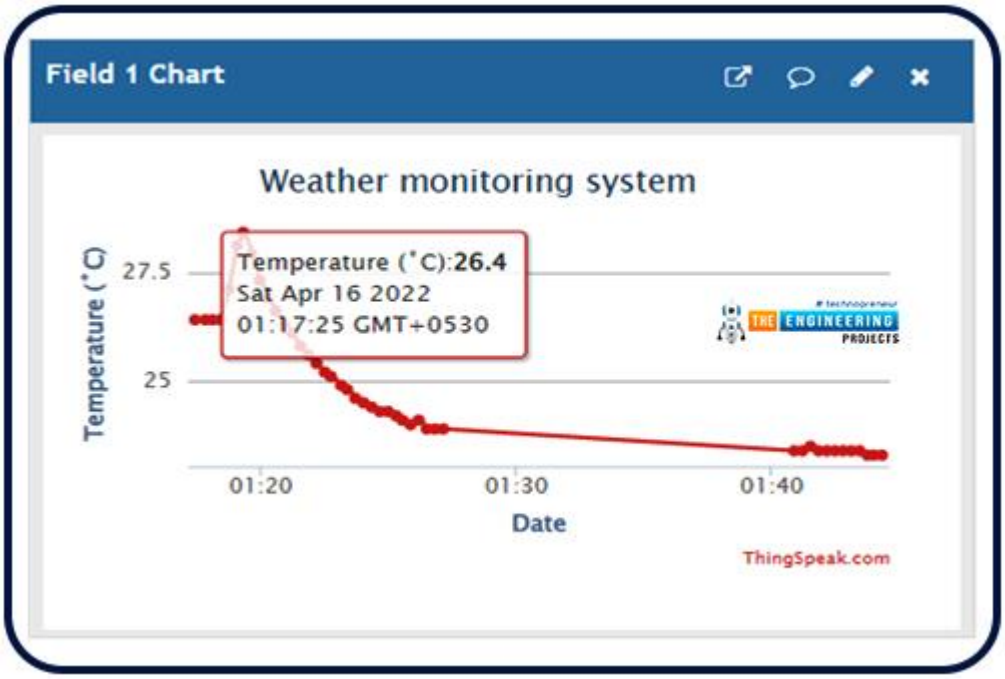


Fig. 2Temperature (°C)

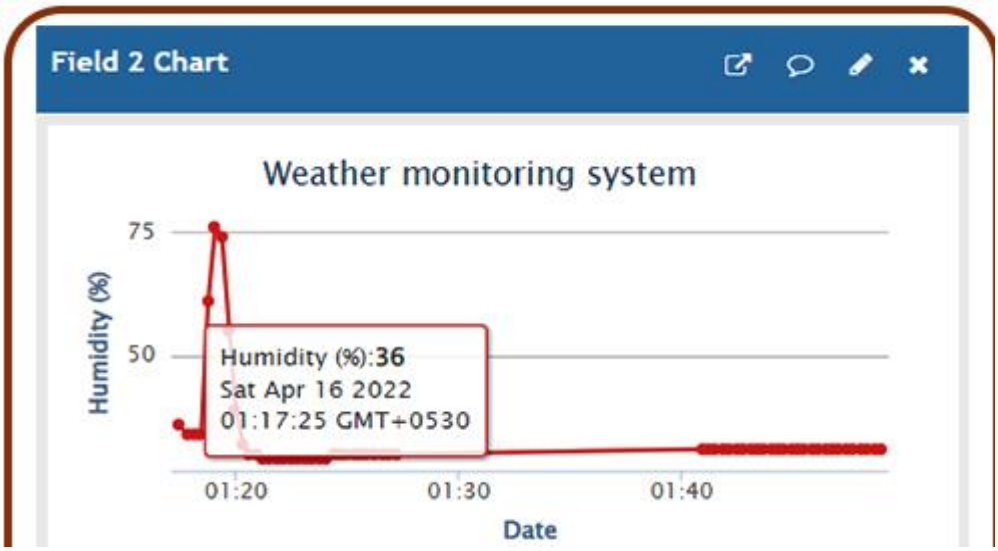


Fig. 3 Humidity

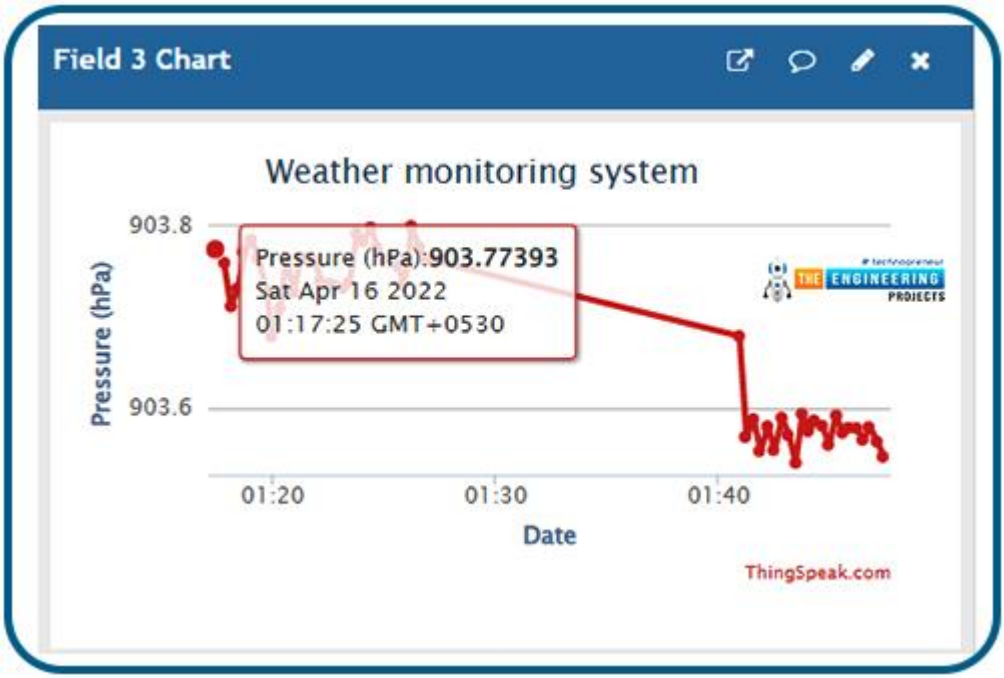


Fig. 4 Pressure (hPa)

Conclusion

We observed the weather conditions (that includes temperature, humidity, pressure and altitude) with our “Web-server based weather monitoring system using ESP32”. The observed real time data is stored on the ThingSpeak server which can be accessed globally. The different values of each (mentioned earlier) environmental factor at different intervals in time are also observed (given in table 5) and the observed result clearly shows the changes in the weather conditions for a full day cycle. Hence, we have successfully implemented and tested the web server based weather monitoring system with ESP32 and ThingSpeak web server.

Future Scope

The future scope of an air monitoring system using ESP32 is vast and can be expanded in many ways to improve its functionality and usability. Here are some potential future developments that could be made to the system:

1. Addition of Gas Sensors: Gas sensors can be added to the system to detect other air pollutants such as sulfur dioxide, nitrogen dioxide, and volatile organic compounds (VOCs). This would allow for more comprehensive monitoring of air quality.

2. Integration with AI and Machine Learning: By integrating the air monitoring system with AI and machine learning algorithms, the system can become more intelligent and capable of identifying patterns and predicting air quality trends. This would provide users with actionable insights for addressing environmental issues.

3. Mobile App Integration: A mobile app can be developed to provide users with real-time access to the air quality data and enable them to set alerts and notifications based on certain parameters.

4. Community Monitoring: The system can be expanded to allow for community-based monitoring of air quality. This would involve deploying multiple sensors in different locations, and the data collected can be used to identify air quality trends and potential sources of pollution.

5. Integration with Smart Home Devices: The air monitoring system can be integrated with smart home devices such as air purifiers and HVAC systems to automatically adjust settings based on the air quality data.

6. Real-time Forecasting: The system can be expanded to include real-time forecasting of air quality conditions based on weather and other factors. This would allow users to take preventative measures before air quality deteriorates.

Overall, the future scope of an air monitoring system using ESP32 is promising, and it has the potential to become an essential tool for improving air quality and environmental conditions in various settings.

References

- Maier, A., Sharp, A. and Vagapov, Y., November "Comparative analysis and practical implementation of the ESP32 microcontroller module for the internet of things" 2017 .
- Prisma Megantoro, Brahmantya Aji Pramudita, Vigneshwaran Pandi, "Real-time monitoring system for weather and air pollutant measurement with HTML-based UI application" June 2021.
- Foughali Karim, Karim Fathallah, Ali Frihida, "Monitoring System using web of things in precision agriculture" 2017.
- J.Shah and B. Mishra, "IoT enabled environmental monitoring system for smart cities," in 2016 International Conference on Internet of Things and Applications (IOTA), 2016.
- P.Susmitha , G. Sowmyabala, " Design and Implementation of Weather Monitoring and Controlling System" July 2014.
- S.Navulur, A. S. C. S. Sastry, M. N. Giri Prasad, "Agricultural management through wireless sensors and internet of things" .
- K.Siva, S. Ram, and A. N. P. S. Gupta, "IoT based Data Logger System for weather monitoring using Wireless sensor networks," 2016
- Ferdin Joe John Joseph, "IoT Based Weather Monitoring System for Effective Analytics", 2249 – 8958, Volume-8 Issue-4, April, 2019.
- A F Pauzi, MZ Hasan, "Development of IoT Based Weather Reporting System", 2020.
- ESP32-IoT Development Framework, https://www.espressif.com/en/products/sdks/esp-idf
- Carlo Guarnieri Calo Carducci, Antonello Monti, Markus Hans Schraven, Markus Schumacher, Dirk Mueller, "Enabling ESP32 based IoT Applications in Building Automation Systems", 2019.
- R.K. M. Math and N. V Dharwadkar, "IoT Based Low-cost Weather Station and Monitoring System for Precision Agriculture in India", 2018 (2nd International Conference on I-SMAC).
- N. Tianlong, "Application of Single Bus Sensor DHT11 in Temperature Humidity Measure and Control System [J]", 2010.
- "Create a Web Server With ESP32", https://www.theengineeringprojects.com/2021/11/create-a-web-server-with-esp32.html
- ThinSpeak, https://thingspeak.com/pages/learn_more
- T. Thilangam. J, T. S. Babu, and B. S. Reddy, "Weather monitoring system application using LabView", 2nd International Conference on I-SMAC, 2018.
- J.Weiss and R. Yu, "Wireless Sensor Networking for the Industrial Internet of Things".
- Gaurav Verma, Pranjal Mittal, Shaista Farheen, "Real Time Weather Prediction System Using IoT and Machine Learning.
- R.Shete, S. Agrawal, "IoT based Urban Climate Monitoring using Raspberry Pi", International Conference on Signal Processing and Communication, 2016.