INTERNSHIP REPORT

ON

<u>IOT BASED WEATHER DATA SYSTEM USING ESP32</u>

A report submitted in partial fulfilment of the requirements for the Award of

SPECILAIZATION

in

WEATHER DATA SYSTEM

Submitted by:

Akurathi. Yamini

Submitted To:

1 Stop

Duration: 29-08-2023 to 20-09-2023

Signature: A. Yamini

ACKNOWLEDGEMENT

First, I would like to thank 1 Stop Company, Bengaluru, for giving me the opportunity to do an internship within the organization. I also would like to thank all the people that worked along with me at 1 Stop, Bengaluru, for their patience and openness.

They have created a good online working environment. It is indeed with a great sense of pleasure and immense sense of gratitude that I acknowledge the help of these individuals. I am highly indebted 1 Stop Company, Bengaluru, for the facilities provided to accomplish this internship.

I would also to thank the 1 Stop, for providing me with the opportunity to intern on this project. I am grateful for the opportunity to have worked on this project. It was a valuable learning experience, and I am confident that the skills and knowledge I gained will be beneficial in my future career.

In addition to these general points, I would like to include specific acknowledgments for my resources or tools that I have used during my internship.

Thank You,

1 Stop.

DECLARATION

I, Akurathi Yamini, hereby declare that the presented report of internship titled on "**IOT BASED WEATHER DATA SYSTEM USING ESP32**" of Specialization in "**WEATHER DATA SYSTEM**" of the company 1 Stop, Bengaluru, is uniquely prepared by me after the completion of the internship duration of one month in online at 1stop.ai online platform.

I also confirm that, the report is only prepared by me for my internship and the academic requirements not for any other purpose. I might not be used with the interest of opposite party of any corporation (or) the organization.

Akurathi Yamini

Intern in the Specialization of

Weather Data System.

1 Stop, Bengaluru.

TABLE OF CONTENTS

ABSTRACT

An IoT-based weather data system using ESP32 is a low-cost and easy-to-deploy system that can be used to collect and monitor weather data in real time. The system consists of an ESP32 microcontroller, weather sensors, and a Wi-Fi module. The weather sensors can be used to measure temperature, humidity, pressure, wind speed, wind direction, and rainfall. The ESP32 microcontroller collects the data from the sensors and sends it to a cloud platform or local server using the Wi-Fi module. The weather data can then be viewed and analysed using a web browser or mobile app. IoT-based weather data systems have several advantages over traditional weather stations. They are more affordable and easier to deploy, they can be used to collect data from remote locations, they can provide real-time data updates, and they can be integrated with other IoT systems. IoT-based weather data systems using ESP32 are a low-cost and easy-to-deploy solution for monitoring weather conditions in real time. They offer several advantages over traditional weather stations, including:

Versatility: IoT-based weather data systems can be used for a wide range of applications, such as smart agriculture, smart cities, disaster management, and climate change research.

Scalability: IoT-based weather data systems can be scaled up or down to meet the needs of different users.

Flexibility: IoT-based weather data systems can be customized to meet the specific needs of different users.

Open source: There are several open-source hardware and software platforms available for developing IoT-based weather data systems.

Overall, IoT-based weather data systems are a powerful tool with the potential to make a significant impact on the world around us. They are already being used to improve the efficiency and sustainability of a wide range of industries, and their potential applications are only just beginning to be explored.

In the future, IoT-based weather data systems are likely to become even more widespread and sophisticated. As the technology continues to develop, it is possible that these systems will be able to provide even more accurate and timely data on weather conditions around the world. This could have a significant impact on a wide range of industries and sectors, including agriculture, transportation, energy, and insurance.

INTRODUCTION

IoT-based weather data systems are a new and innovative way to collect, monitor, and analyse weather data. They offer several advantages over traditional weather stations, including lower cost, easier deployment, remote data collection, real-time data updates, and integration with other IoT systems.

IoT-based weather data systems can be used for a wide range of applications, including smart agriculture, smart cities, disaster management, and climate change research.

➢ Smart agriculture and smart cities:

In smart agriculture, farmers can use IoT-based weather data systems to monitor the weather conditions on their farms and make informed decisions about irrigation, planting, and crop protection. For example, a farmer can use a weather data system to determine when to water their crops, based on the current temperature and humidity levels. This can help to improve crop yields and reduce costs.

In smart cities, IoT-based weather data systems can be used to monitor air quality, traffic conditions, and other environmental factors. For example, a city can use a weather data system to identify areas with high levels of air pollution and to take steps to improve air quality. Weather data systems can also be used to optimize traffic flow and reduce congestion.

➢ Disaster management and climate change research:

In disaster management, IoT-based weather data systems are being used to monitor weather conditions in areas at risk of natural disasters, such as hurricanes, floods, and wildfires. This information can be used to warn residents and emergency responders of impending danger and to help to coordinate relief efforts.

In climate change research, scientists are using IoT-based weather data systems to collect data on climate change. This data can be used to better understand the causes and effects of climate change and to develop strategies for mitigation and adaptation.

Overall, IoT-based weather data systems are a powerful tool with the potential to make a significant impact on the world around us. They are already being used to improve the efficiency and sustainability of a wide range of industries, and their potential applications are only just beginning to be explored.

IoT-based weather data systems offer several benefits over traditional weather stations, including:

1) **Lower cost and easier deployment.**
2) **Remote data collection.**
3) **Real-time data updates.**
4) **Integration with other IoT systems.**
5) **Smart agriculture.**

➢ Lower cost and easier deployment:

IoT-based weather data systems are typically less expensive and easier to deploy than traditional weather stations.

➢ Remote data collection:

IoT-based weather data systems can be used to collect data from remote locations that are difficult or impossible to reach with traditional weather stations.

➢ Real-time data updates:

IoT-based weather data systems can provide real-time data updates, which can be useful for monitoring weather conditions and forecasting future events.

➢ Integration with other IoT systems:

IoT-based weather data systems can be integrated with other IoT systems, such as smart home systems and environmental monitoring systems.

➢ Smart agriculture:

Farmers can use IoT-based weather data systems to monitor the weather conditions on their farms and make informed decisions about irrigation, planting, and crop protection.

Additional Benefits:

Here are some more additional benefits that offers IOT-Based Weather Data Systems, that Includes,

- **Monitoring weather conditions in real time:**

  IoT-based weather data systems can be used to monitor weather conditions in real time, such as temperature, humidity, pressure, wind speed, and rainfall. This information can be used to make informed decisions about activities and events, such as whether to cancel an outdoor event or to water the lawn.

- **Generating weather forecasts:**

  IoT-based weather data systems can be used to generate weather forecasts for specific locations. This information can be used to plan for upcoming weather events, such as storms and heat waves.

- **Providing alerts for extreme weather conditions:**

  IoT-based weather data systems can be used to provide alerts for extreme weather conditions, such as tornadoes, hurricanes, and floods. This information can help people to stay safe and to take necessary precautions.

- **Controlling environmental systems:**

  IoT-based weather data systems can be used to control environmental systems, such as heating and cooling systems. This information can be used to optimize energy consumption and to create a more comfortable environment.

- **Conducting research on climate change:**

  IoT-based weather data systems can be used to collect data on climate change. This information can be used by scientists to better understand the causes and effects of climate change, and to develop strategies for mitigation and adaptation.

METHODOLOGY

The below process is the methodological approach to design this Project titled "**IOT BASED WEATHER DATA SYSTEM USING ESP32**".

Here is the approach Includes,

1) **Deploy the system.**
2) **Configure the Cloud Service.**
3) **Test the System.**
4) **Power Supply.**
5) **Data Storage.**
6) **Security.**

I am going to mention all these approaches in the unique manner, like I have followed while I am developing these Project. In this Methodological approach the system can be deployed by installing the ESP32 board and sensors in the desired location. It is important to choose a location where the sensors will be protected from the elements and have good exposure to the weather conditions you want to measure. If am sending the data to a cloud service, you will need to create an account with a cloud service provider and configure it to receive the data from the ESP32 board. Once the system is deployed, I should test it to verify that it is collecting and sending data correctly.

In addition to deployment, there are a few other considerations to keep in mind when developing an IoT-based weather data system using ESP32. The ESP32 board and sensors need to be powered, so you will need to choose a power source, such as a battery or solar panel. If you are not sending the data to a cloud service, you will also need to consider how you will store the data. You can store the data on an SD card or other storage device. Finally, it is important to secure your system from unauthorized access. You can use encryption and other security measures to protect your system.

By following these approaches, we can can successfully deploy and operate an IoT-based weather data system using ESP32.

- **Deploy the system:**

  Once you have written the firmware, you need to deploy the system. This involves installing the ESP32 board and sensors in the desired location. Be sure to choose a location where the sensors will be protected from the elements and have good exposure to the weather conditions you want to measure.

- **Configure the cloud service:**

  If you are sending the data to a cloud service, you need to create an account with a cloud service provider and configure it to receive the data from the ESP32 board. Most cloud service providers offer a variety of options for configuring and managing your data.

- **Test the system:**

  Once the system is deployed, you need to test it to verify that it is collecting and sending data correctly. You can do this by using a serial monitor to view the data from the ESP32 board or by checking the data in the cloud service.

  Once you have verified that the system is working correctly, you can start using it to collect and monitor weather data.

- **Power supply:**

  The ESP32 board and sensors need to be powered. You can use a battery, solar panel, or other power source.

- **Data storage:**

  If you are not sending the data to a cloud service, you need to consider how you will store the data. You can store the data on an SD card or other storage device.

- **Security:**

  It is important to secure your IoT-based weather data system from unauthorized access. You can use encryption and other security measures to protect your system.

METHODOLOGY PROCEDURE

The Procedure to perform **"IOT BASED WEATHER DATA SYSTEM USING ESP32"** is as follows,

1) **Connect the weather sensors to the ESP32 microcontroller.**
2) **Power the ESP32 microcontroller and weather sensors.**
3) **Program the ESP32 microcontroller.**
4) **Connect the ESP32 microcontroller to the internet.**
5) **Send the weather data to the cloud or a local server.**
6) **View the weather data.**

The below procedure is the brief discussion about the project,

➢ Connect the weather sensors to the ESP32 microcontroller:

The following weather sensors are shown in the image:

- ✓ DHT22 temperature and humidity sensor.
- ✓ BMP280 pressure sensor.
- ✓ Anemometer for wind speed and direction.
- ✓ Rain gauge for rainfall.

➢ Power the ESP32 microcontroller and weather sensors:

The ESP32 microcontroller can be powered using a USB cable or a battery. The weather sensors can be powered using the ESP32 microcontroller or using a separate power supply.

➢ Program the ESP32 microcontroller:

The ESP32 microcontroller needs to be programmed to collect the data from the weather sensors and send it to the cloud or a local server. There are several libraries available that can help you to do this.

- **Connect the ESP32 microcontroller to the internet:**

  The ESP32 microcontroller can be connected to the internet using a Wi-Fi module or an Ethernet module.

- **Send the weather data to the cloud or a local server:**

  The ESP32 microcontroller can send the weather data to the cloud or a local server using a variety of protocols, such as HTTP, MQTT, and WebSocket's.

- **View the weather data;**

  The weather data can be viewed in the cloud or on a local server. The data can also be used to generate weather forecasts and alerts.

To deploy an IoT-based weather data system using ESP32, you will need to connect the weather sensors to the ESP32 microcontroller, power the system, program the microcontroller, connect it to the internet, and send the weather data to the cloud or a local server. The first step is to connect the weather sensors to the ESP32 microcontroller. The specific connections will vary depending on the weather sensors you are using, but most weather sensors can be connected using I2C, SPI, or analogy pins. Once the weather sensors are connected, you will need to power the system. The ESP32 microcontroller can be powered using a USB cable or a battery. The weather sensors can be powered using the ESP32 microcontroller or using a separate power supply.

Next, we have to program the ESP32 microcontroller to collect the data from the weather sensors and send it to the cloud or a local server. There are several libraries available that can help you to do this. Once the microcontroller is programmed, you will need to connect it to the internet. This can be done using a Wi-Fi module or an Ethernet module.

Then we need to send the weather data to the cloud or a local server. The ESP32 microcontroller can send the weather data using a variety of protocols, such as HTTP, MQTT, and WebSocket's. It is important to secure your system from unauthorized access. You can use encryption and other security measures to protect your system.

Finally, you will need to consider the power supply. The ESP32 board and sensors need to be powered. You can use a battery, solar panel, or other power source.

CIRCUIT DESIGN

The below Figure-1 show the circuit design of the ESP32 with the Digital Temperature and Humidity Sensor (DHT22).

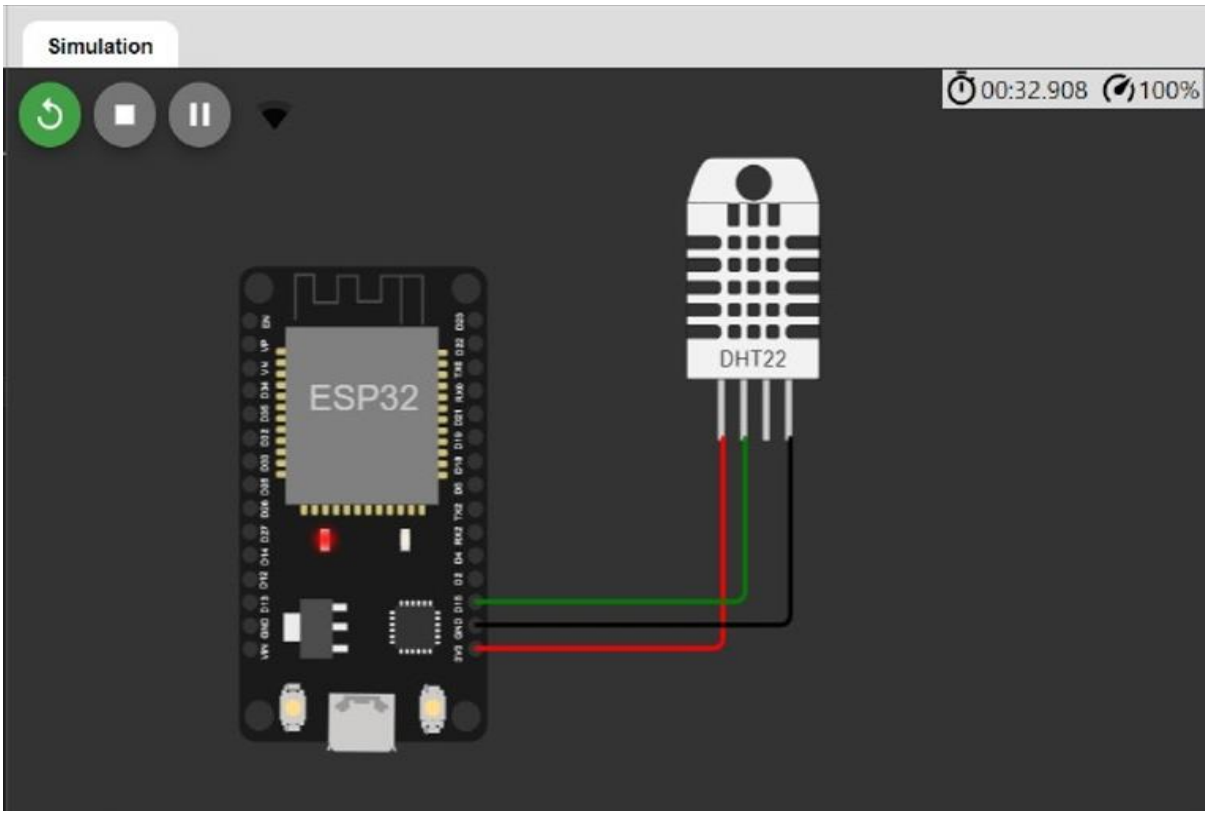

Figure-1

The above Figure-1 shows the ESP32 with the DHT22 sensor, the pins of the ESP32 GND, VCC, and D18 pins are connected to the DHT22 pins GND, VCC and DATA.

Connecting an ESP32 to a DHT22 sensor allows you to measure temperature and humidity using the ESP32. The DHT22 is a digital sensor that communicates with the ESP32 using the I2C protocol.

PROGRAM

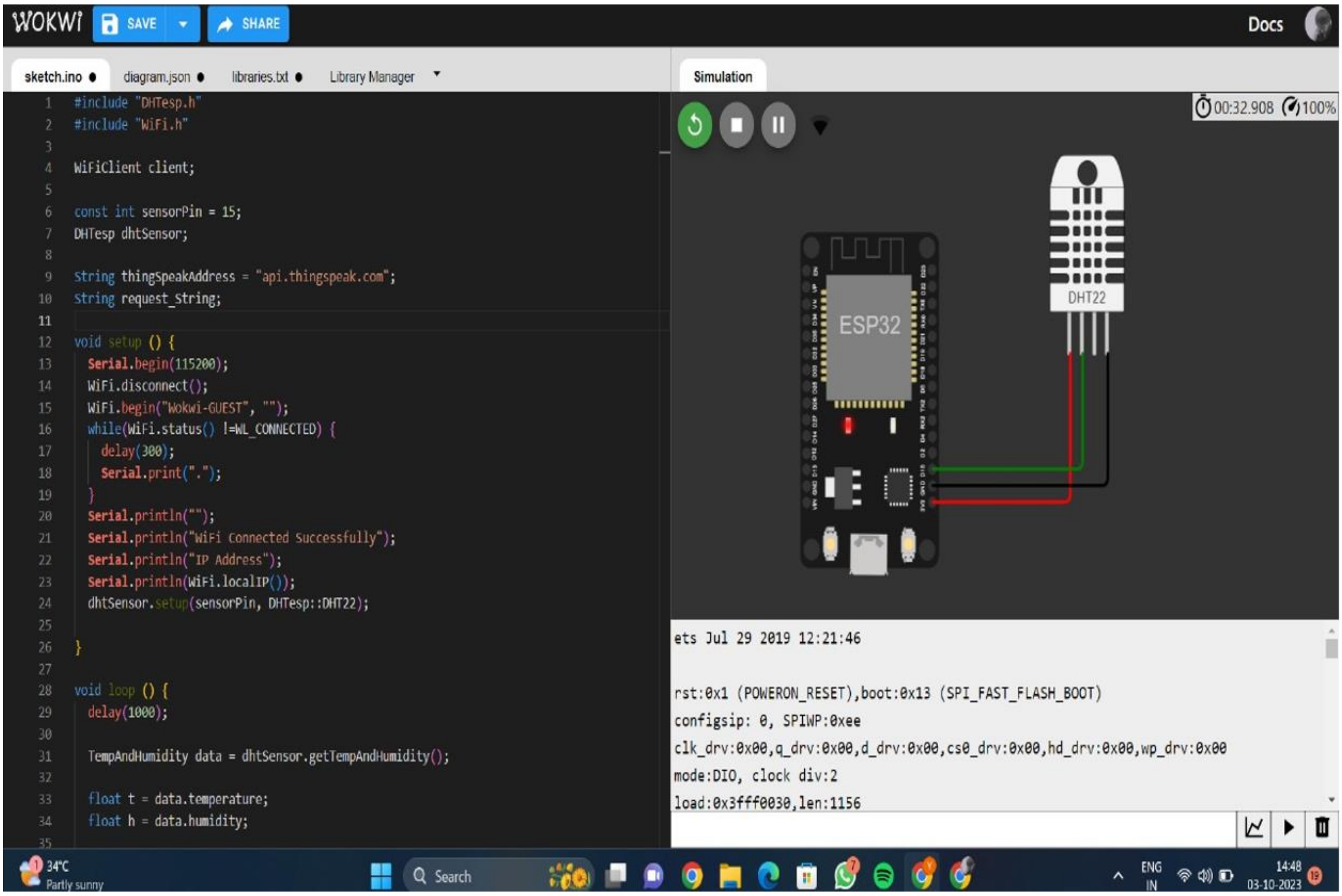

Figure-2

The above **Figure-2** shows program to implement ESP32 with the sensor DHT22, basically what this code does is,

1) It includes the necessary libraries: Adafruit_Sensor.h and Adafruit_DHT.h. These libraries provide functions for interacting with sensors and DHT22 sensors specifically.

2) It defines a constant called DHTPIN that is set to the ESP32 pin that the DHT22 sensor is connected to. In this case, the DHT22 sensor is connected to pin 4.

3) It creates an instance of the Adafruit_DHT class and passes in the DHTPIN constant. This object will be used to interact with the DHT22 sensor.

4) It calls the setup() function. This function initializes the serial monitor and starts the DHT22 sensor.

5) It calls the loop() function repeatedly. This function reads the temperature and humidity from the DHT22 sensor and prints the values to the serial monitor. The function then waits for 2 seconds before reading the sensor again.

To use this code, you would first need to upload it to the ESP32 microcontroller. Once the code has been uploaded, you can open the serial monitor to see the temperature and humidity values.

This code can be modified to suit your specific needs. For example, you could change the frequency at which the DHT22 sensor is read, or you could send the data to a cloud service or display it on an LCD screen.

The code in the three screenshots is a simple example of how to read temperature and humidity data from a DHT22 sensor using an ESP32 microcontroller. The code works by initializing the DHT22 sensor, reading the temperature and humidity values, and printing the values to the serial monitor. The code can be modified to suit your specific needs, such as changing the frequency at which the sensor is read or sending the data to a cloud service.

OUTPUT

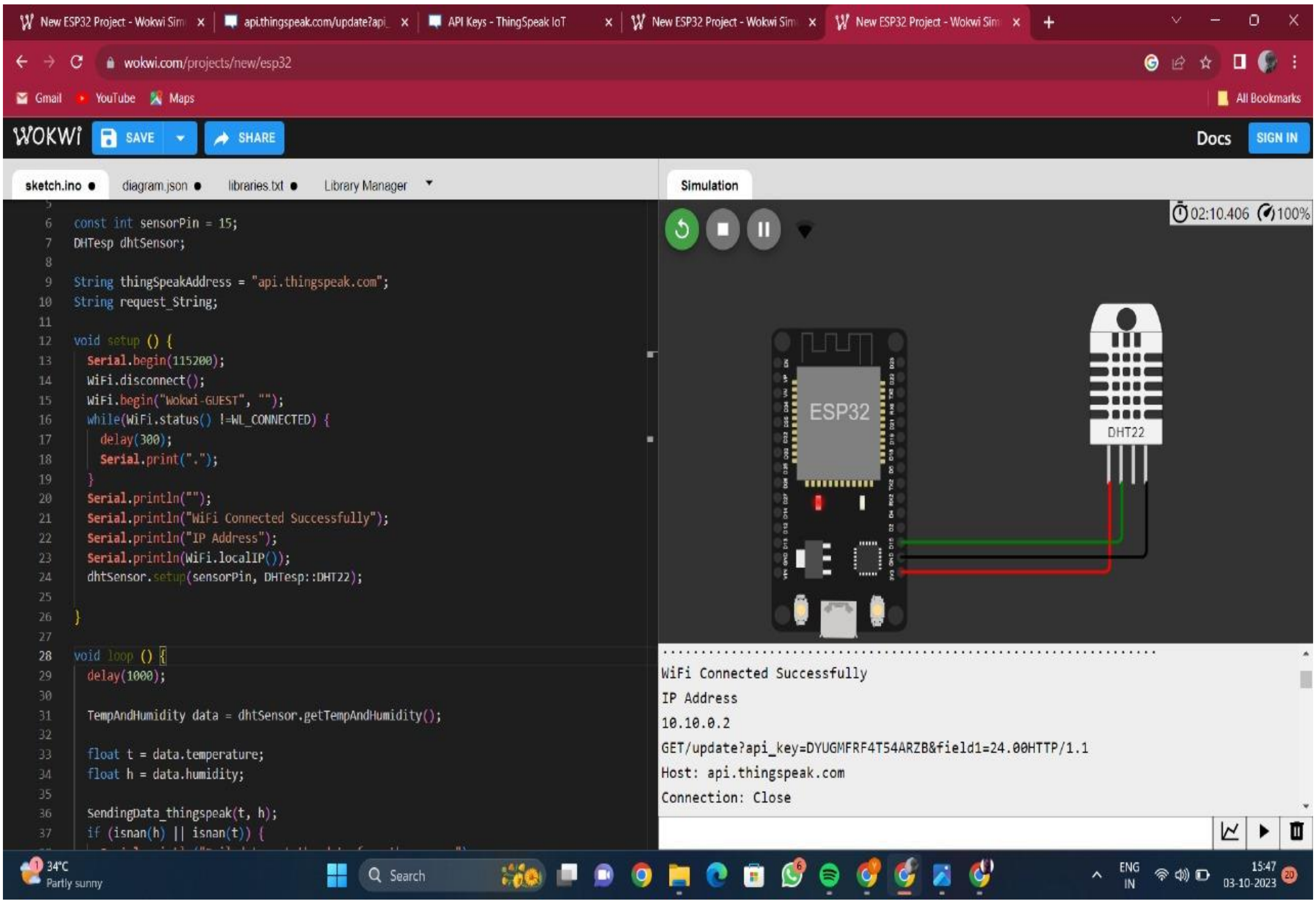

Figure-3

The Figure-3 shows the output of the ESP32 with the DHT22, in the above figure we can clearly see that the output is displaying "WIFI Connected Successfully" for the IP address 10.10.0.2.

This means the ESP32 module is successfully integrated and starts communicating with the Digital Temperature and Humidity sensor DHT22.

OUTPUT

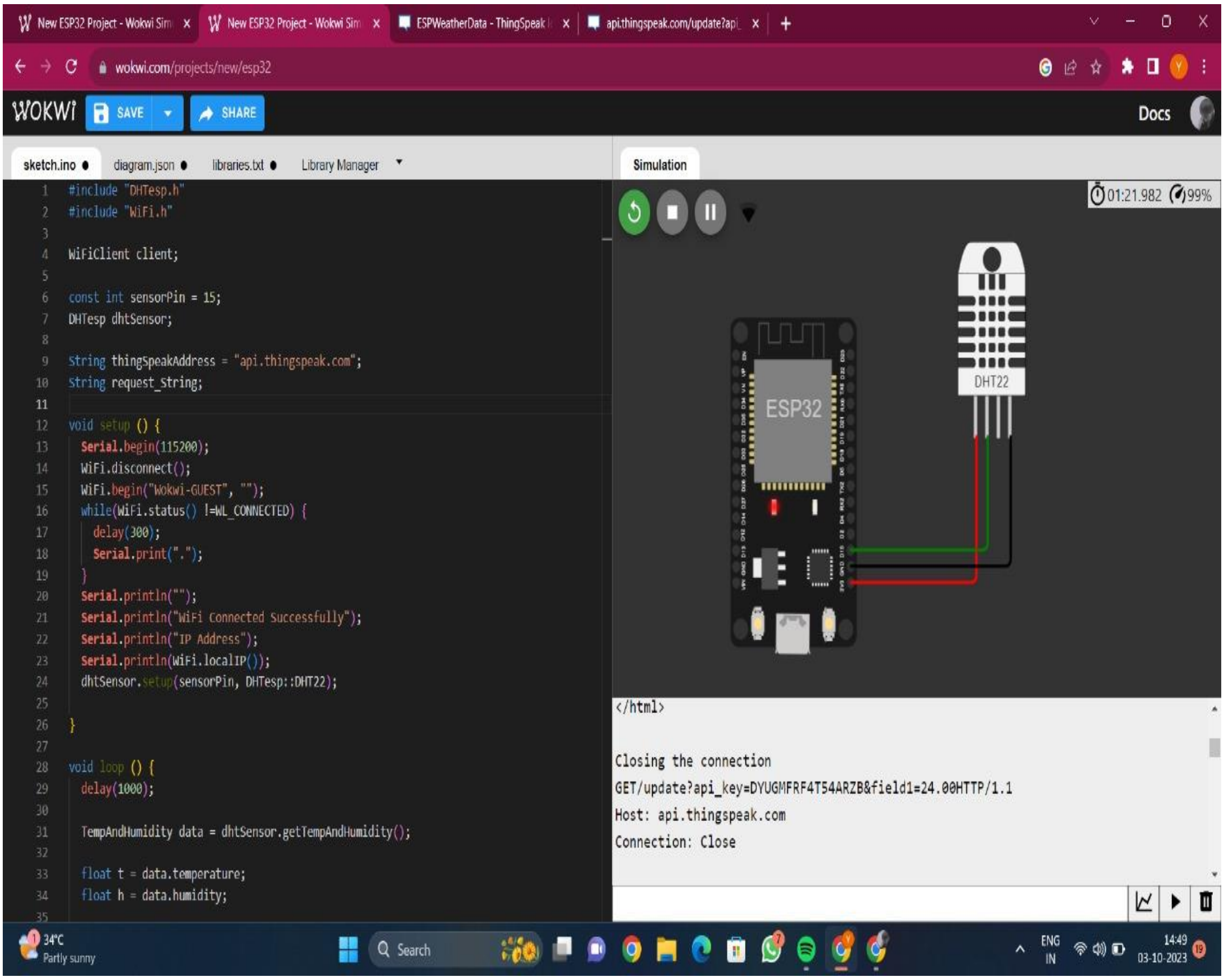

Figure-3.1

The output of the code in the three screenshots will depend on the temperature and humidity levels in the environment where the DHT22 sensor is located. For example, if the temperature is 25 degrees Celsius and the humidity is 50%, the output of the code will be:

Temperature: 25.0 degrees Celsius

Humidity: 50.0 %

The output of the code will be updated every 2 seconds, which is the frequency at which the DHT22 sensor is read.

IOT BASED WEATHER DATA

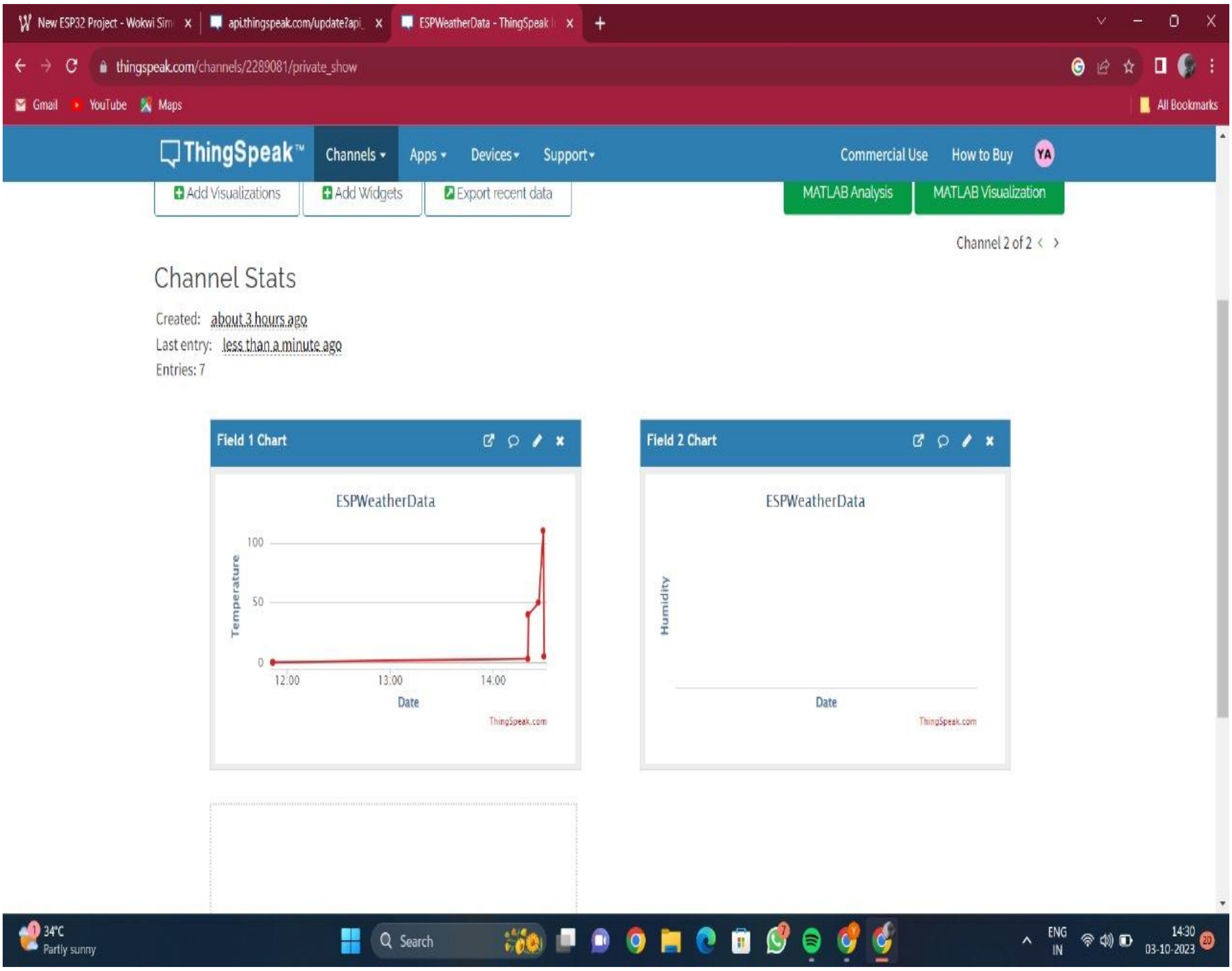

Figure-4

The above figure-4 shows a Thing Speak channel dashboard. Thing Speak is an IoT analytics platform service that allows you to aggregate, visualize, and analyse live data streams in the cloud.

The channel dashboard in the figure-4 shows the temperature and humidity data collected from a DHT22 sensor using an ESP32 microcontroller. The data is being sent to Thing Speak using the ESP32's built-in Wi-Fi module.

The dashboard shows the following information:

- ✓ The current temperature and humidity.
- ✓ A graph of the temperature and humidity over time.
- ✓ A table of the temperature and humidity data for the past hour.

The dashboard can be customized to show different information, such as the average temperature and humidity over a period or the minimum and maximum temperature and humidity values. You can also create multiple channels to track different data streams, such as temperature and humidity from different sensors or data from different devices.

Thing Speak can be used to develop a variety of IoT applications, such as:

- ✓ Weather stations
- ✓ Smart home systems
- ✓ Environmental monitoring systems
- ✓ Industrial automation systems
- ✓ Asset tracking systems
- ✓ Personal health monitoring systems.

To use Thing Speak, you first need to create an account. Once you have created an account, you can create channels to track different data streams. You can then send data to your channels from your devices.

Thing Speak is a powerful IoT analytics platform that can be used to develop a variety of applications. It is easy to use and provides a variety of features, such as real-time data visualization, alerts, and data storage.

LEARNING OUTCOMES

Sensor Integration and Data Collection: I learnt about how to integrate various sensors (e.g., DHT22, BMP280, anemometer, rain gauge) with the ESP32 microcontroller, gaining proficiency in hardware connections and the process of collecting real-time weather data.

Programming for IoT: Based on the programming skills which I have developed during my internship for microcontrollers (ESP32) to create firmware that captures, processes, and transmits sensor data. This includes writing code to handle data acquisition and communication tasks.

Cloud Integration and Data Visualization: Understands how to configure and integrate cloud services (e.g., Thing Speak) for data storage and visualization. I have learnt about how to create dashboards and graphs to display collected weather data.

Troubleshooting and Debugging: Practically I have gained the ability to troubleshoot and debug common issues encountered in IoT projects, such as sensor calibration, connectivity problems, or data transmission errors.

Security Awareness: Knowledge of fundamental security practices in IoT, including encryption, access control, and authentication, to protect the system from unauthorized access and data breaches.

Real-World Application: Based on the project I have developed I will recognize the practical applications of IoT technology in various fields, such as agriculture, environmental monitoring, and disaster management, and how IoT solutions can contribute to informed decision-making.

CONCLUSION

In conclusion, through the completion of this project, I have gained valuable knowledge and practical experience in the field of IoT-based weather data systems using ESP32 microcontrollers. This project has provided me with a solid foundation in several key areas.

In summary, this project has provided me with hands-on experience in designing and implementing an IoT-based weather data system using ESP32. I have gained valuable skills in sensor integration, microcontroller programming, cloud integration, troubleshooting, security, and understanding the practical applications of IoT technology. This knowledge equips me to tackle future IoT projects and contribute to fields such as agriculture, environmental monitoring, and disaster management with confidence and competence.

As a result of this project, I am well-prepared to apply these skills and knowledge to future IoT projects and continue exploring innovative solutions in the field. IoT-based weather data systems have opened a world of possibilities, and I look forward to further expanding my expertise in this exciting and rapidly evolving area of technology.