**Academic Report Cover Page**

Sed et lacus quis enim mattis nonummy

Urna Semper

Instructor’s Name

21 April 2023

**Overview of the Project**

The Overview of the Project is about designing a mobile-based temperature monitoring and control system for server rooms. Currently, the existing system is based on IoT and involves a mobile app acting as a medium between an IoT device and a smartphone. The IoT device provides information on the temperature inside the server room through various sensors that measure all parameters of temperature that can be measured. The proposed system continuously sends data to the cloud to monitor data from anywhere, and it also has features to provide notifications to users through the application dynamically.

The current system is limited in its ability to regulate temperature in server rooms effectively, and there is a need for a more comprehensive system that can manage the ideal temperature of a server room to prevent critical room conditions. The aim of this project is to provide an easy approach to monitoring and controlling the temperature of server rooms.

The proposed system will address the limitations of the current system by improving the monitoring and regulation of temperature in server rooms. The system will use sensors to measure temperature and humidity levels, and the data collected will be used to regulate the temperature in the server room using an Arduino actuation module. The system will also send notifications via the mobile app to alert users when temperature values fall below or rise above the set threshold values.

Overall, the goal of this project is to provide a comprehensive mobile-based temperature monitoring and control system for server rooms that will ensure proper accountability and transparency for temperature levels in server rooms, resulting in a more efficient and effective system for managing the ideal temperature of a server room.Section 1 Requirements Gathering

Section 1 Requirements Gathering

1.1 **Sampling Techniques**

Sampling is the process of selecting a subset of individuals or objects from a larger population with the aim of drawing conclusions about the population. The process of picking a selection of individuals or items from a larger population in order to make conclusions about the population is known as sampling. In other words, a sample of data from a bigger dataset or population is used to represent the entire population.

Because each individual in the population has an equal chance of being chosen, we used the simple random method.

1.2 Target Population and Sampling Size

Population refers to the entire group of individuals, objects, or events that share a common characteristic or feature of interest to the researcher.

Size generally refers to the number of individuals, objects, or events in a sample or population. The size of a sample  is an important consideration in research design and analysis, as it can affect the precision, accuracy, and generalizability of the results.

The target population for this study is the workers at DICTS at Makerere University in Kampala, which consists of 11 individuals. The sample size required for this study can be calculated using the following formula:

Sample Size = N / (1 + N\*e^2)

Where N is the population size and e is the margin of error.

In this case, N = 11 and e = 5%

1.3 Data Collection Methods and Instruments

The interviews were conducted at the DICTS data center, and participants were selected based on their availability and expertise in server room management.

During the interviews, the research team asked open-ended questions and allowed participants to share their thoughts and experiences in their own words. This method allowed them to gather rich qualitative data that complemented their quantitative survey data.

The technical team's insights were particularly valuable as they have hands-on experience with managing server rooms and could provide a more detailed understanding of the challenges and limitations of existing temperature monitoring systems. Additionally, the personal and in-depth nature of the interviews created a more engaging and trusting relationship between the participants and researchers, which can lead to more honest and detailed responses.

The information gathered through the interviews was used to further understand the specific challenges and pain points of managing a server room temperature monitoring system and informed the development of a mobile-based solution that would address these issues. Overall, the use of interviews provided valuable insights and a deeper understanding of the technical team's needs and experiences, which ultimately led to a more effective solution.

In order to collect information from a particular participant group, namely servers and DICTS employees, we used questionnaires. The use of both closed-ended and open-ended questions allowed the questionnaires to be specifically crafted to the requirements and experiences of each group.

To contact participants who were informed and experienced in their respective disciplines, the study team circulated the questionnaires through networks and organizations for professionals. The open-ended questions allowed participants to provide more thorough and individual responses while the closed-ended questions produced quantitative data that could be easily examined.

The research team was able to gather a wide variety of data from different participant groups using these methodologies, which gave them a thorough knowledge of the study questions they were examining. When combined

**Section 2: Data Analysis and Findings**

**2.1 Methods Used to Analyze Data:**

Data collected was analyzed using qualitative methods of analyzing data which included the following.

Narrative analysis: This involves analyzing the structure and content of stories or narratives shared in interviews or surveys to identify themes and patterns in the ways that people construct meaning and understand their experiences.

Content analysis: This method involves analyzing text data to identify specific categories or codes. For example, as researchers,we used content analysis to code responses to survey questions into categories such as "positive," "negative," or "neutral."

System Architecture

The development and implementation of the prototype require both hardware and software requirements. It means that the system needs specific hardware components and other software resources. of a server room temperature monitoring system consists of various hardware and software components that interact with each other to ensure the system functions as required. The architecture involves sensors for monitoring the temperature, an Arduino board for processing the data, a Wi-Fi module for communication, and a web application for user interaction.

3.1 Hardware Requirements

The hardware tools consist of choosing the proper hardware for developing prototypes, a computer, a mobile device, and other materials.

The suggested system needs automated and remote cooling, temperature monitoring, and a notification mechanism. The prototype therefore needs the ESP-WROOM32 microcontroller, a DHT11 temperature sensor, electrically programmable relays, fans that act as air conditioners, a 12V solar battery, a mobile phone, resistors, jumper wires, a breadboard, and a rooter.

A laptop computer is used to power and program the ESP- WROOM32 board during the system development. In order to share the internet connection (Wi-Fi) with the ESP WROOM board, a mobile device and rooter utilized as hotspots. As a result, ESP32 transmits temperature sensory data to the cloud to enable data processing and visualization. Additionally, the web based mobile applications and ESP web applications makes cooling devices and remote data viewing possible. Following are the main components for prototyping the project.

(i) The ESP-WROOM32 microcontroller

The ESP32 chip, also referred to as the Microcontroller ESP-WROOM32, is capable of functioning as a full standalone system. According to Foltynek et al. (2019), the chip integrates Wi-Fi, Bluetooth, and additional communication interfaces like SPI and I2C/UART. This microcontroller has certain advantages over others, including the size of the chip, the quantity of pins, and the incorporated Wi-Fi.

The ESP32 DOIT DEVKIT V1 board and the 30 pin versions are the ones utilized to construct the prototype on a breadboard.

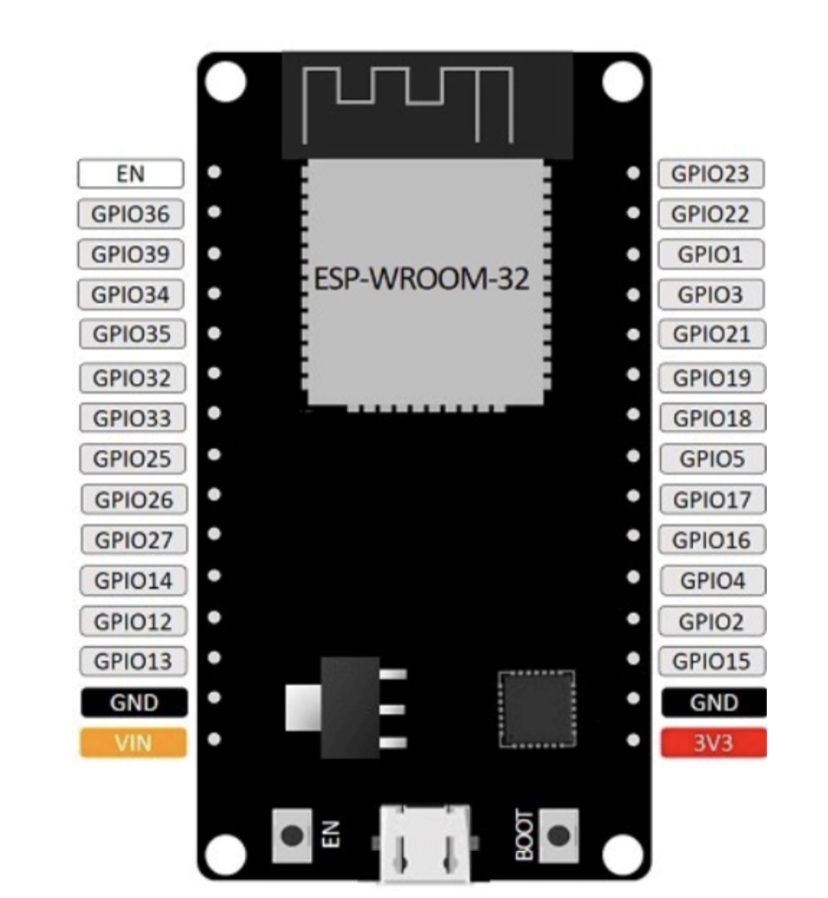


Figure 1: ESP32 DOIT DEVKIT V1 Board version with 30 GPIOs Pinouts

(ii) Digital Temperature and Humidity Sensor

There are numerous temperature sensors that are compatible with development boards like Arduino, ESP32, ESP8266, and others (Santos, 2016). Six commonly used temperature sensors, including the DHT11, DHT22, LM35, DS18B20, BME280, and BMP180, (Santos, 2019).

The best temperature sensor for this project, in terms of characteristics and accuracy, is the DHT22. This sensor can measure temperatures between -40 and 80 degrees Celsius and provides a humidity reading between 0 and 100% with an accuracy of 2 to 5%. (Alvan et al.)

One DHT22 sensor was utilized to obtain the precise temperature data that was then saved to the cloud based on these criteria, the size of the data center room, and the DHT22 sensor. (Lady-ada, 2020).

(iii) Electrical Relays

To control one electrical circuit, relays open and close connections in a different circuit. Relays are programmable switches that cause disruption in the circuit so that users can interact with it and control it from a distance. To open and close circuits, they use electromechanical or electrical devices.

Figure 14: Single

3.4.2 Software requirements

Software needed for the proposed system includes Fritzing for creating the temperature monitoring and control system, Arduino IDE with the necessary libraries for programming the Microcontroller ESP-WROOM32, and cloud platforms (software and APIs) to facilitate data storage.

HTML scripts for relay remote control and PHP scripts for email and SMS notifications can be edited in Sublime Text. ThingSpeak is a cloud platform used to store sensory data, and a website hosting relays is created using the ESP webserver library. ThingSpeak mobile app (Thing View Free) is used for visualizing temperature data in real-time.

3.5 System overview

The proposed system design consists of four parts: a temperature monitoring system, data processing, data visualization, and remote control of relays. Sensor data is sent and stored in the cloud, data processing is done by Microcontroller ESP-WROOM32, and data visualization is done by a mobile application. Remote control is done through the mobile application.

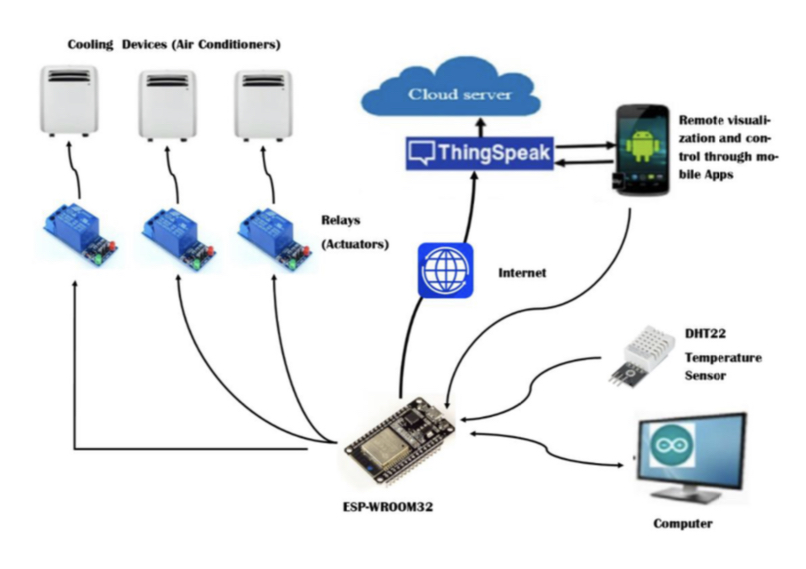


Figure 2: Prototype system architecture

The prototype has an ESP-WROOM32 board as a center board, a DHT22 sensor, three 12V-Fans and three relays, a 12V battery, cloud storage, and a mobile phone. To control the cooling system, the cell phone is used to remotely turn on and off the relays.

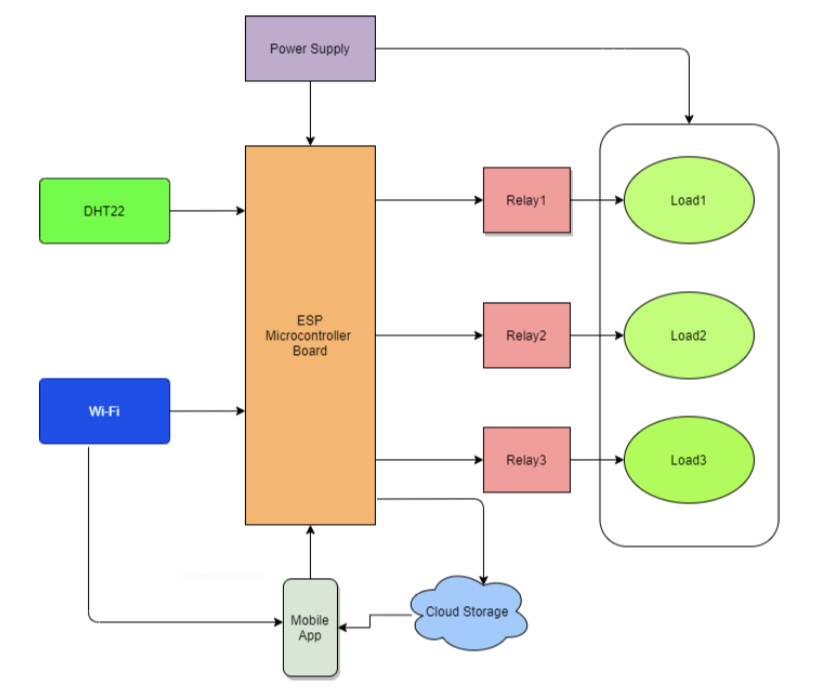


Figure 3: Block diagram of the system

3.5.1System Assembling

Once built, every component of the system must function in accordance with Fig. 2. A temperature monitoring system starts the operation, and the cooling and notification system responds in accordance with the detected data. Every time, the sensed data is transferred to the cloud for analysis and visualization.

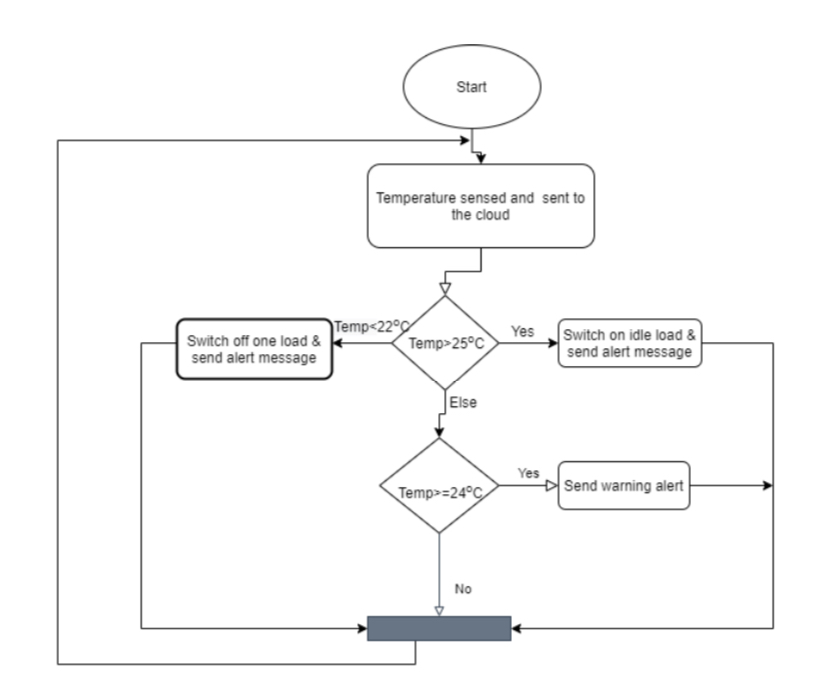


Figure 4: The flowchart diagram of the system

3.5.2 Temperature Monitoring System

The proposed Temperature Monitoring system was developed by using the DHT22 temperature and humidity sensor. The current detected temperature value is sent to the cloud database to be remotely visualized. The e-mail and SMS notifications from the cloud platform are regularly done pushed by the webserver script (Fig. 5).

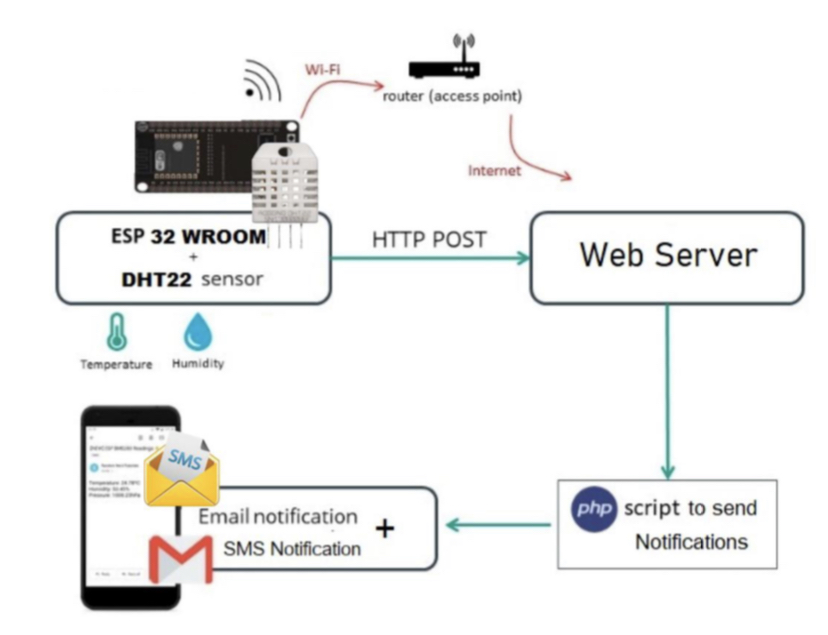


Figure 5: Temperature monitoring and notification system

The temperature sensed is also sent to the cloud ThingSpeak over the internet by the means of the cloud API (Fig. 6).

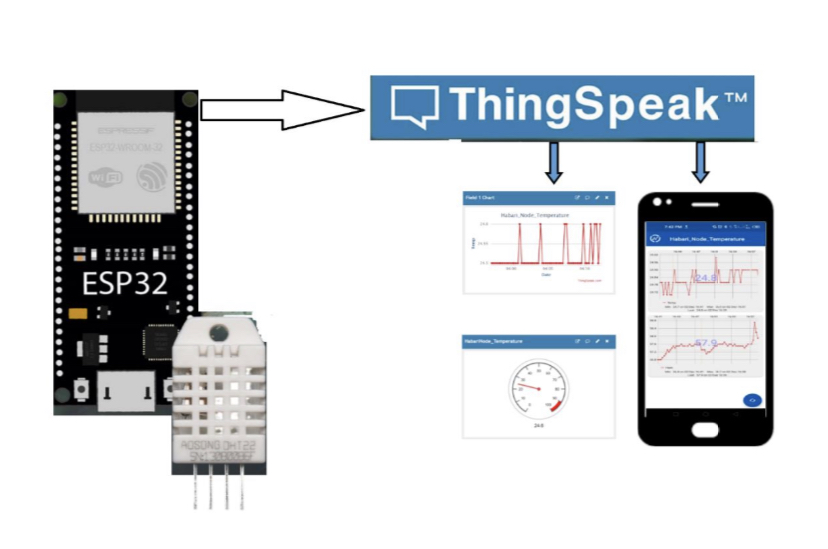


Figure 6: Temperature sensing and monitoring system

3.5.3 Automated cooling System

The auto-switching system for cooling devices operates through actuation that is either triggered by a device failure or a signal from the microcontroller. The schematic to flip the relays when the temperature deviates from the typical range can be seen here.

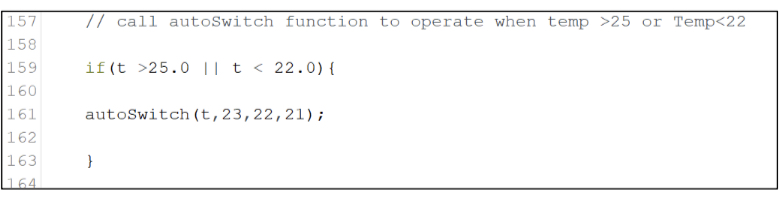


Figure 7: displays the scenario of the backup and auto-switch system.

The function autoSwitch() only operates when the temperature t is above 25 ̊C or below 22 ̊C to automatically switch ON or OFF the cooling devices controlled by the relays connected to ESP pin 21, 22, and 23.

Security mechanism activates standby backup unit when unit malfunctions, alerting data center management.

3.5.4 Remote Control of Cooling Devices

Remote control cooling systems use a mobile application to manually turn on or off relays to help the automated system decide if a cooling device can or cannot function. To access an ESP32 web-server from outside its local network, a static IP address and port forwarding or tunneling must be configured.

(i) Setting ESP Static IP Address

The rooter must be configured to always assign the same IP address when requested to give the ESP board a static IP address. The MAC address of the ESP board can be obtained from the Arduino sketch, and a static IP address is chosen from the pool of IP addresses that can be given to clients connected to the rooter.

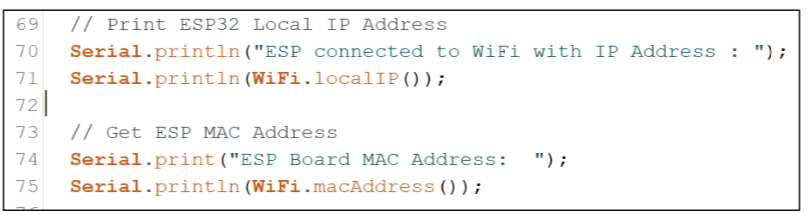


Figure 8: Arduino sketch to get ESP board IP and MAC Addresses

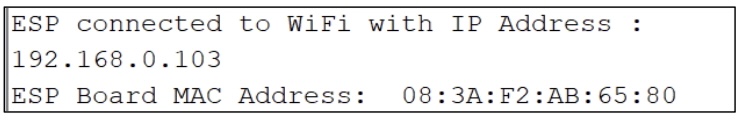


Figure 9: Output of Arduino sketch to get IP and MAC Addresses

The IP address 192.168.0.103 and MAC address 08:3A:F2:AB:65:80 are then used to configure the IP address of the rooter as shown in Figure 10.

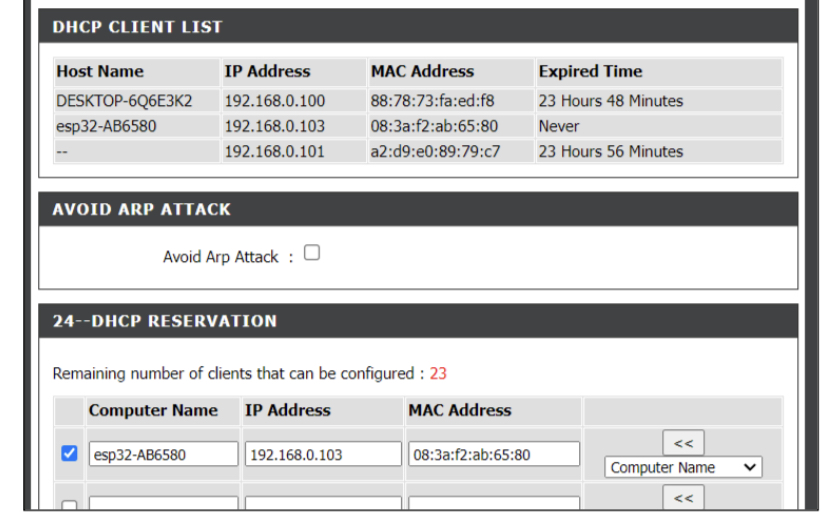


Figure 10: Static IP address reservation for ESP Board on D-Link Rooter

When a reserved IP address for an ESP board is assigned, it will never expire or be assigned to another device. The ESP32 board will then continue to get 192.168.0.103 from the same rooter as its IP address.

(ii) Port Tunneling

The prototype used port tunneling to access the ESP web server from outside of the LAN, as it is more secure than port forwarding. It was created using a free version of Ngrok, which uses a secure tunnel to expose local networked services to the public internet.

Port tunneling requires a Tunnel Auth-token from the Ngrok platform, a static IP Address, and a port through which the ESP's website is accessible.

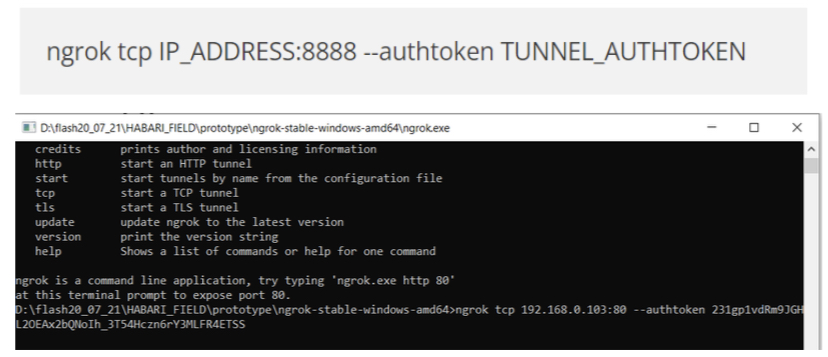


Figure 11: Port tunneling configuration in Ngrok window

When the instruction of the Figure 12, was run, it generated a URL that can be used to reach the ESP web server from any device that is connected to the internet.

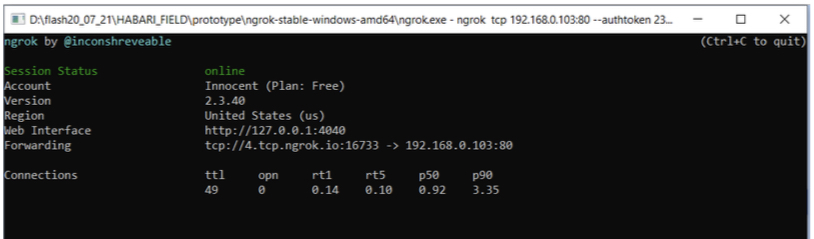


Figure 12: Port tunneling trough Ngrok

In Fig. 13, http://4.tcp.ngrok.io:16733 is a public URL that helps access the ESP webserver from any device connected to the internet regarding Figure 14.

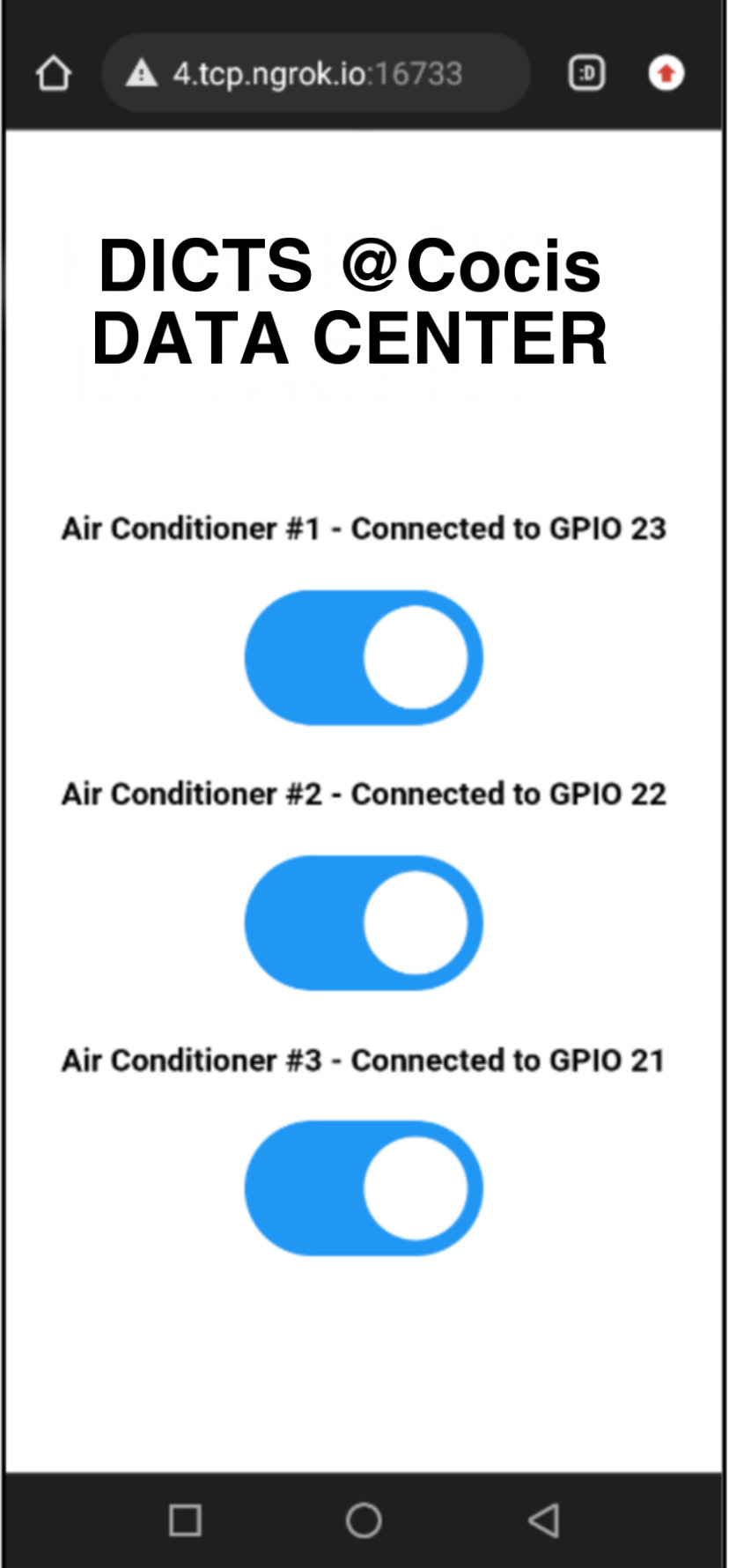


Figure 14: ESP accessed through Ngrok URL

3.5 System Implementation

The ESP-WROOM32 microcontroller board is used to program the temperature monitoring system and activate/deactivate relays, and can communicate with the cloud for data storage.

3.5.1 Connecting DHT22 to ESP Board

The DHT22 sensor must be connected to an ESP board pin to take input data, as it needs to operate between 3.3 V and 5 V.



Figure 15: Digital humidity and temperature sensor 22 pinouts

3.5.2 Connecting Relays to ESP Board

The relay module has three sockets: common (COM), normally closed (NC), and ordinarily open (NO). The COM connector is connected to the current that needs to be controlled, while the NC configuration is used when the relay should always be closed. The NO configuration breaks the circuit unless the ESP32 sends a signal to close it.

NO configuration was implemented to enable the current to flow, making testing easier.



Figure 16: Pin configuration of a single channel relay module

The COM socket is connected to the main power (Solar battery) and the NO is connected to the load (VCC of the fan). All relays are connected to the ESP board using GND, VCC, and SIG.

3.6.3 Implementation of the Overall System

Figure 17 shows the overall system design formed by an ESP32 board, 3 relays, and a DHT22 temperature sensor.

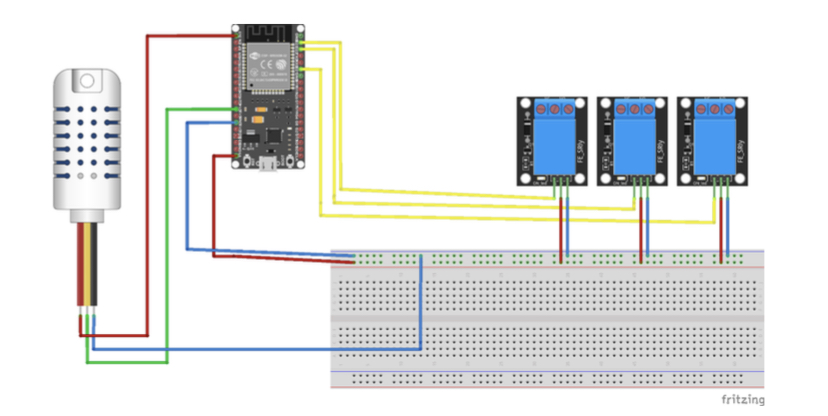


Figure 17: Prototype circuit diagram

The prototype was put into practice on both a PCB and a breadboard to test its usefulness in DC and Alternative Current systems, but its implementation on the PCB increased confidence in its usefulness.

Figure 18 Prototype implemented on PCB

4.2 Temperature Monitoring System

The temperature monitoring system is composed of the ESP32 board, the DHT22 temperature sensor (Figure 15), and the ThingSpeak cloud storage.

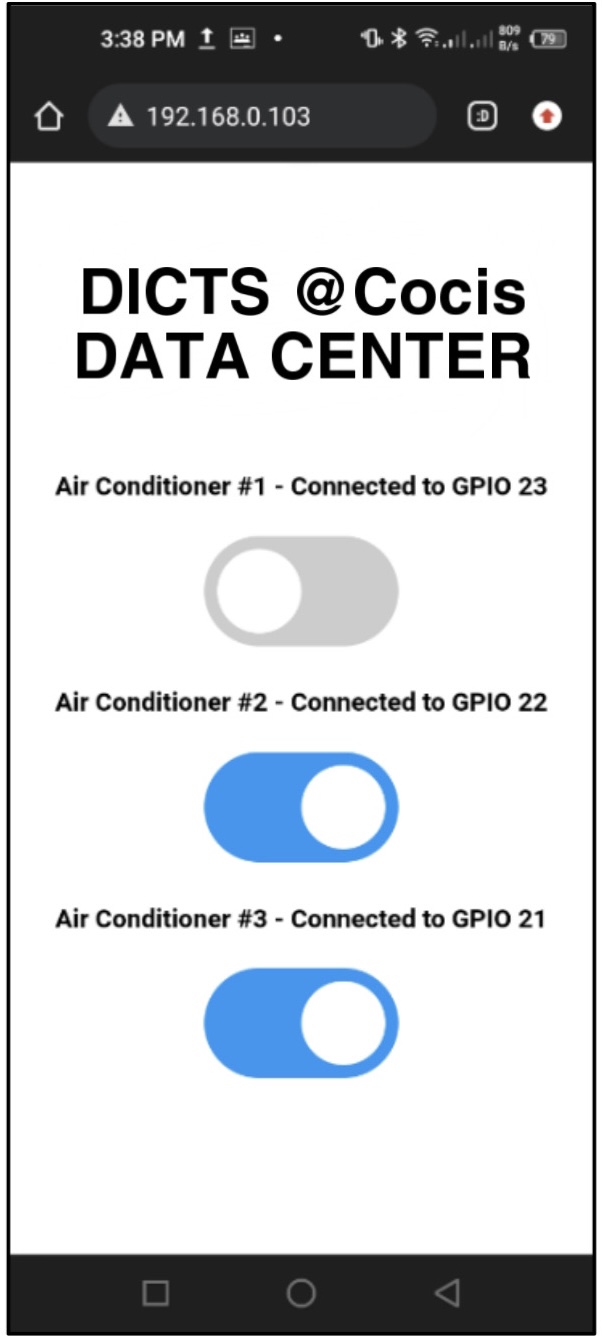
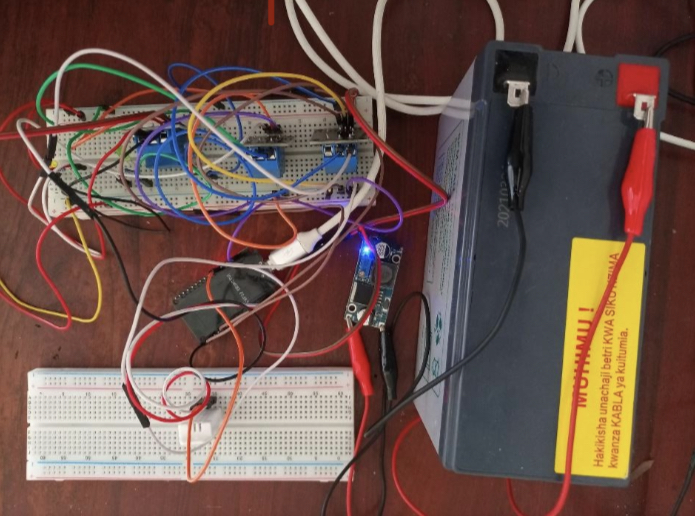
The temperature monitoring system was able to accurately detect the value of temperature in the environment and save it to the cloud (Fig. 19).

Figure ThingSpeak chart of the temperature monitored value

Therefore, a gadget was set in the cloud to help end-user to see the variation of temperature in real-time. However, the last monitored value of temperature is displayed along with the gadget for much accuracy as shown in Fig. 20

In contrast with the existing system to monitor temperature, the developed system was able to save data and to provide to the end-user an interface to visualize the current and recent value of temperature.

4.3 Notification System

A webserver was used to monitor the value and provide notifications when a weird value was found. The monitored value is sent by the ESP32 board to both the cloud and the website at the same time. If a threshold is surpassed (either below 22 ̊C or above 24 ̊C), an email is sent to the IT personnel.

The system has two kinds of notification; on one hand, there is an SMS notification sent to the mobile number of the Data center manager

On the other hand, there is a mail sent to the data center manager just to compliment the SMS notification to maximize the transmission of the information to the end-user of the system (Fig. ).

Figure 36: Mail notification

As indicated in Table 6, the new system powered by backups utilized would significantly contribute to saving the expenses used to run the current four cooling devices at the same time. Only three devices are used while one of them stands as a backup to intervene when it is needed. Therefore, it can reduce the cost spend on the data center facility to half since the idle cooling device would work rarely or when the data center wants to interchange it with a working one.

Data Processing and Visualization

4.4.1 ThingSpeak for Temperature Storage and Visualization

3,168 3,168

The temperature data from the data center was processed using an ESP 32 microcontroller board and a DHT22 sensor, as described in Appendix 2. As a result, whether through a web interface or on a mobile phone, the cloud provides an excellent interface for displaying the monitored value. The newest entry overlaps the plotted data on the mobile app interface, which displays all of the entries of the monitored value. In addition, Figure 37 shows the lowest and maximum values as well as the date when they were taken.

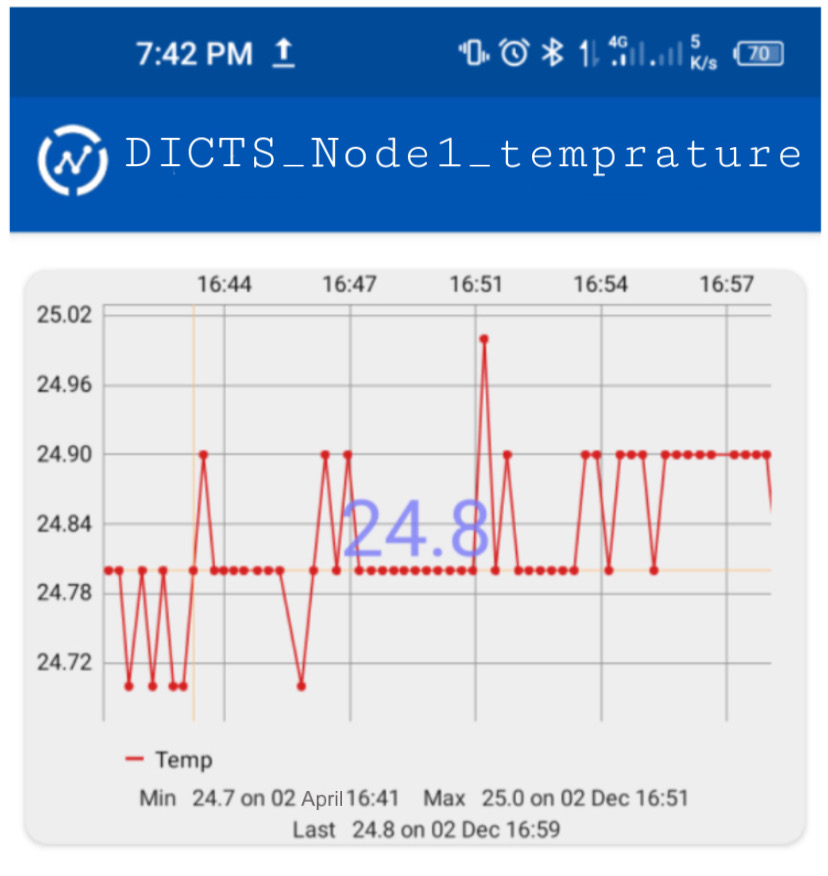


Figure : Mobile view of the ThingSpeak channel

4.4.2 ESP Web Application

The web application that allows remote control of cooling units was made possible because of the ESP web-server. The virtual push-button may then be switched on or off based on the signal sent through the ESP32 Room Board's web server.

Through the IP address supplied to the ESP board, the application may be accessed using any browser on a PC or a smartphone (Fig. ).

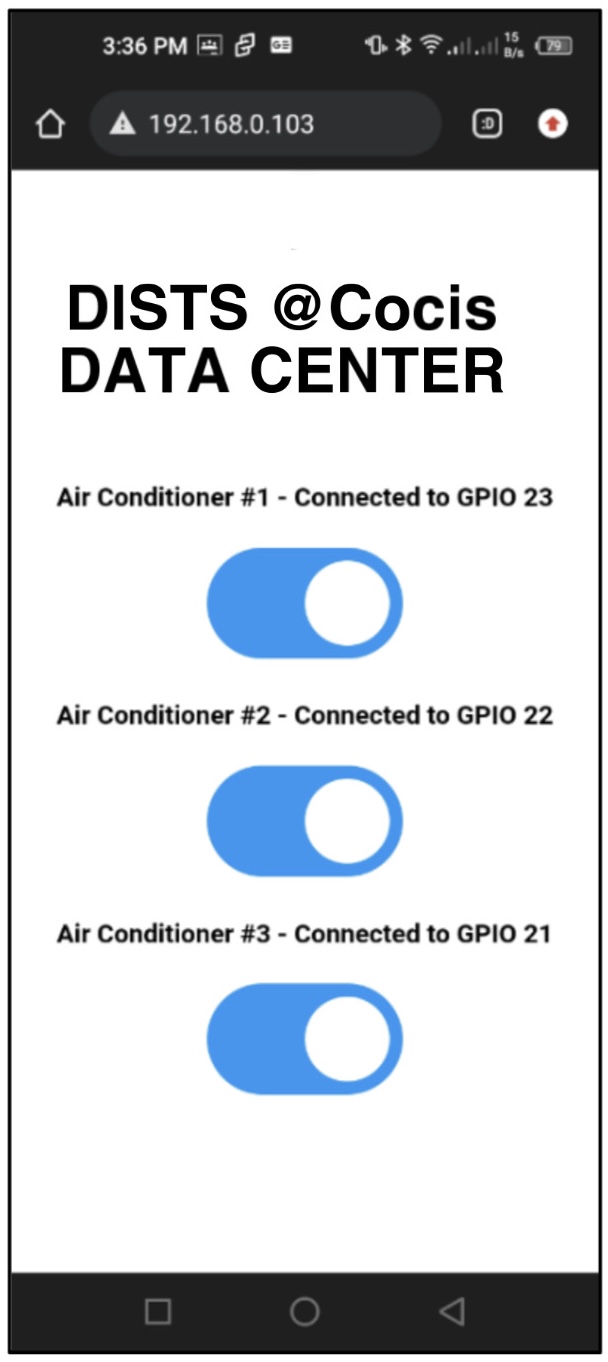


Figure : A user interface to control ESP board pins

The virtual toggle buttons (Fig. ) might transmit a high or low signal to the board pin attached to the cooling device. They then send the input signal to the ESP board to control the related device connected directly to the pins of the ESP board.

Figure : Toggle button in high signal to switch off the AC1

3.7 System Integration and Testing

The suggested system was tested at DICTS, College of Computing and Information Sciences in Makerere University. To guarantee that the system operates as expected by the end user, both unity and integration testing were required.

3.7.1 Integration Testing

Each part of the system is integrated into the whole system for integration testing to make sure that the entire system can accurately work. The entire system was tested for several days to guarantee that everything worked well. The temperature monitoring system was tested to determine whether a sudden temperature increase would result in a signal, and the cooling device was remotely operated to check if it could accept the signal given by the ESP web server.

3.7.2 Unit Testing

Unit testing is used to detect and correct misbehavior after the system is integrated. The temperature monitoring system is used to determine if data from the data center room is delivered and stored in the cloud. An e-mail message is sent if the temperature level approaches the limit range, and relays must be tested for both automatic and remote control.

3.7.3 Acceptance testing

Acceptance testing was performed to ensure the system's operation, dependability, accuracy, and usability. Test cases were created and executed to ensure the temperature monitoring system was integrated with the temperature sensor and cloud storage, an email message was sent when the temperature level approached the limit range, and relays worked properly for automatic and remote control.

4 Deployment and maintenance

4.1 Deployment Process

To deploy the software, the team first set up the server environment. They chose a cloud-based server and selected a provider that offered reliable uptime and scalability. The team then installed the necessary software dependencies, including the operating system and any required libraries.

Once the server was set up, the team installed the software itself. This involved copying the code from the development environment to the server and configuring it to run in the production environment. They also set up any necessary configuration files and database connections.

After installation, the team tested the software thoroughly to ensure that it was working correctly in the production environment. They also conducted load testing to make sure that the server could handle the expected traffic.

4.2 Maintenance Plan

We developed a comprehensive maintenance plan to ensure that the software remained functional and secure over time. They implemented regular system backups to protect against data loss in case of server failure or cyber attack.

We also scheduled regular updates for both the operating system and the software itself. These updates were designed to address any downtimes that cause security vulnerabilities. Additionally, monitored the system closely to detect any issues and respond to them quickly.

To ensure that the server was always available, as a team we implemented a monitoring system that monitored the status of the server and using a mobile phone application that allows users to remotely monitor and control the temperature of the server room.

regular intervals. Any issues that were detected were flagged for immediate attention.

Overall, the team's deployment and maintenance processes were designed to ensure that the software remained functional, secure, and responsive to changing needs over time. By implementing these processes, the team was able to minimize downtime, protect against data loss, and ensure that the system continued to meet the needs of its users.

5 Conclusion

6 Provide a brief overview of the project status and any next steps

Citation

Alvan, P. U., Aziz, A., Winarno, & Harjito, B. (2019). Server Room Temperature & Humidity Monitoring Based on Internet of Thing (IoT). Journal of Physics: Conference Series, 1306(1). https://doi.org/10.1088/1742-6596/1306/1/012030

Association, G. (2014). Understanding the Internet of Things (IoT). GSMA Connected Living, 1(July), 13.

Sari, M., & Sadikin, N. (2019). Server Temperature Monitoring System Using Web Based Censor and SMS Gateway. Journal of Physics: Conference Series, 1361(1), 012026 IOP Publishing. https://doi.org/10.1088/1742-6596/1361/1/012026

7] J. P. SIPAN, "Temperature & Humidity Monitoring & Control System Based On Arduino And Sim900a Gsm Shield," International Journal Of Electrical, Electronics And Data Communication, Vol. 5, No. 11, P. 62, 2017.

[8] S. A. Nugroho, " Detektor Suhu Ruangan Dengan Tombol Pengatur Manual Berbasis Mikrokontroler AT89S51," Seminar Riset Unggulan Nasional Informatika Dan Komputer FTI UNSA, Vol. II, No. 2, Pp. 15-21, 2013.

[9] Santos, S. (2019). DHT11 vs DHT22 vs LM35 vs DS18B20 vs BME280 vs BMP180 | Random Nerd Tutorials. https://randomnerdtutorials.com/dht11-vs-dht22-vs-lm35-vs- ds18b20-vs-bme280-vs-bmp180/

[10] T. Liu, "Digital-Output Relative Humidity & Temperature Sensor/Module DHT22 (DHT22 Also Named As AM2302)," Aosong Electronics Co.,Ltd.

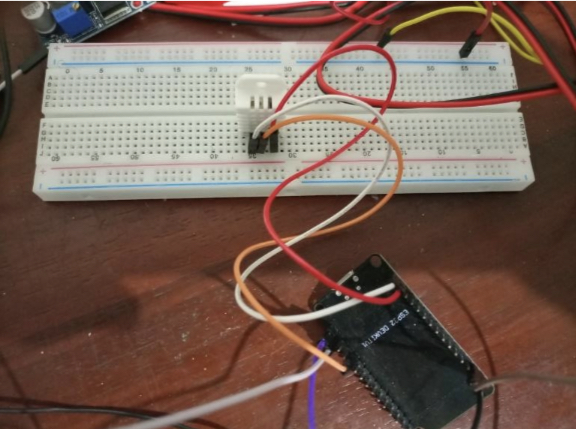
[11] A. H. Saptadi, "Perbandingan Akurasi Pengukuran Suhu Dan Kelembaban Antara Sensor DHT11 Dan DHT22," Jurnal Infotel , Vol. VI, No. 2, Pp. 49-55, 2014.

Appendices (optional)



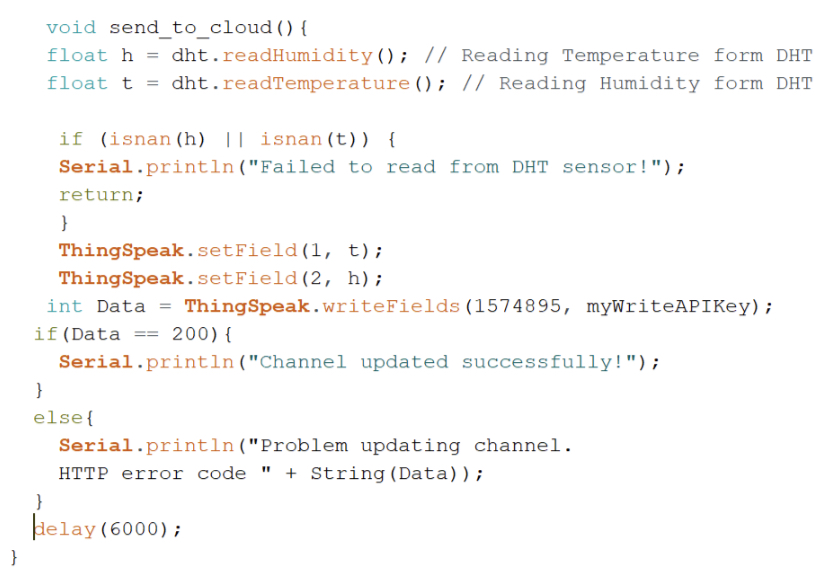
Appendix 1: Module Relay(actuator)

Appendix 2: Prototype implemented on breadboards



Appendix 3: Temperature monitoring system

Appendix 4: Arduino Sketch for Relay Auto-Switching





Appendix 5: PHP Script to Send Email and SMS