Scientific Research Publishing

**Advances in Internet of Things, 2021, 11, 77-93**
https://www.scirp.org/journal/ait
ISSN Online: 2161-6825
ISSN Print: 2161-6817

# IoT Based Greenhouse Real-Time Data Acquisition and Visualization through Message Queuing Telemetry Transfer (MQTT) Protocol

**Vincent de Paul Niyigena Kwizera[1], Zhanming Li[1*], Victus Elikplim Lumorvie[2], Febronie Nambajemariya[3], Xiaowei Niu[1]**

[1]School of Electrical and Information Engineering, Lanzhou University of Technology, Qilihe District, Lanzhou, China
[2]School of Computer and Communication, Lanzhou University of Technology, Qilihe District, Lanzhou, China
[3]School of Electronics and Information Engineering, Lanzhou Jiaotong University, Lanzhou, China
Email: *lizm@lut.edu.cn, vdp.nkwizera@hotmail.com, elivictus2012@gmail.com, febroniemariya@gmail.com, 17803477803@163.com

**How to cite this paper:** Niyigena Kwizera, V.D.P., Li, Z.M., Elikplim Lumorvie, V., Nambajemariya, F. and Niu, X.W. (2021) IoT Based Greenhouse Real-Time Data Acquisition and Visualization through Message Queuing Telemetry Transfer (MQTT) Protocol. *Advances in Internet of Things*, **11**, 77-93.
https://doi.org/10.4236/ait.2021.112006

**Received:** March 24, 2021
**Accepted:** April 27, 2021
**Published:** April 30, 2021

## Abstract

One of the most indispensable needs of life is food and its worldwide availability endorsement has made agriculture an essential sector in recent years. As the technology evolved, the need to maintain a good and suitable climate in the greenhouse became imperative to ensure that the indoor plants are more productive hence the agriculture sector was not left behind. That notwithstanding, the introduction and deployment of IoT technology in agriculture solves many problems and increases crop production. This paper focuses mainly on the deployment of the Internet of Things (IoT) in acquiring real-time data of environmental parameters in the greenhouse. Various IoT technologies that can be applicable in greenhouse monitoring system was presented and in the proposed model, a method is developed to send the air temperature and humidity data obtained by the DHT11 sensor to the cloud using an ESP8266-based NodeMCU and firstly to the cloud platform ThingSpeak, and then to Adafruit.IO in which MQTT protocol was used for the reception of sensor data to the application layer referred as Human-Machine Interface. The system has been completely implemented in an actual prototype, allowing the acquiring of data and the publisher/subscriber concept used for communication. The data is published with a broker's aid, which is responsible for transferring messages to the intended clients based on topic choice. Lastly, the functionality testing of MQTT was carried out and the results showed that the messages are successfully published.

## Keywords

Greenhouse, Sensors, Monitoring System, Internet of Things (IOT), ThingSpeak, Data Visualization, MQTT, Adafruit.IO, MQTT Testing

## 1. Introduction

The Greenhouse system enclosure allows the management and control of the crop ecosystem resulting in increased crop production, longer production time, enhanced product quality, and less defensive chemicals [1]. To achieve this, the agriculture industry is bound to adopt the Internet of Things because its introduction in the agricultural sector can solve many problems for farmers [2]. According to recent statistics, IoT systems installation in the agriculture sector is rising at a compound annual growth rate of 20 percent (20% CAGR). It is estimated in [3] that the number of connected devices is growing from 13 million to 225 million in the period of 10 years (2014-2024). With the current development worldwide, a lot of agricultural lands have been used in other non-agricultural areas, such as buildings, industrial sites, and other infrastructures, which will undoubtedly lower the agricultural production and have negative impacts on environments and economy in general. With this problem, the need of developing new technologies to close and cover this gap is increasing day-to-day. One solution that the community can implement is to create an agricultural system with limited land availability commonly called urban farming or urban agriculture.

Smart farming IoT platform based on edge and cloud computing was proposed in [4]. In this work, Miguel A. Zamora-Izquierdo *et al.* describes the design, development, and evaluation of a technology that provides automated IoT technology to meet severe PA requirements. It is quite challenging to solve these particular problems such as Soil Fertility Management, Crop Disease, Pest Control, Irrigation and Water Management, etc. in precise monitoring of wide-scale agricultural fields. In [5], N. K. Nawandar and V. R. Satpute designed an IoT-based, low-cost, and intelligent module for smart irrigation systems using three modules as physical part and MQTT with HTTP Protocol for transferring information from field location to the user Daponte P. *et al.* in [6] also proposes an overview of the different techniques applicable to precision agricultural monitoring. The drone architecture for multispectral/thermal sensing and DTM/DSM has been discussed.

Researchers in the IoT domain must also consider the cloud platform that manages information and various IoT protocols such as MQTT, CoAP, and AMQP that are used to communicate and transmit data to the cloud as well as to the end-user in order to keep farmers informed about crop conditions from a remote location. It is expected that the technology will be evaluated further in long-term crop cycles in order to optimize activities and adapt to decision support capability, which will be provided to human operators as management solutions.

The rest of the paper is organized as follows: Section 2 describes the Background Theory and related works, Section 3 shows the Structure and Development of IoT system in the greenhouse, MQTT Protocols and its application in Greenhouse monitoring is depicted in Section 4, Greenhouse Data Acquisition and Monitoring System design was described in Section 5, Section 6 shows the Experimental results and Discussions and Finally, Conclusions are in Section 7.

## 2. Background Theory of IoT

The researchers defined the Internet of Things (IoT) in many different ways. In general, it is a concept based on combining information technology, sensor technology, wireless communication technology, and internet technology [7]. In [8], IoT is defined as a network of physical objects and devices that are integrated with an Internet connection. Also, in [9], they defined IoT as a recent communication paradigm that envisions a near future in which the objects of everyday life will be fitted with sensors, microcontrollers, transceivers for digital communication, and appropriate protocol stacks that will make them able to communicate easily with one another and with the user becoming an integral part of the internet.

The Internet of Things (IoT) changed how we live, work, travel, and do business since 1999, when it was first discovered; it is marking a new industrial transformation, known as Industry 4.0, and key in the digital transformation of organizations, cities, and society as well. **Figure 1** shows the roadmap to understand the essence of the Internet of Things in the past decades.

After the presentation on the application of internet-connected RFID in Procter & Gamble's supply chain, Ashton identified a future in which computers can collect data and transform it into useful information without human intervention [10], which is possible with technology such as sensors and the RFID where computers can observe, interpret and understand the environment.

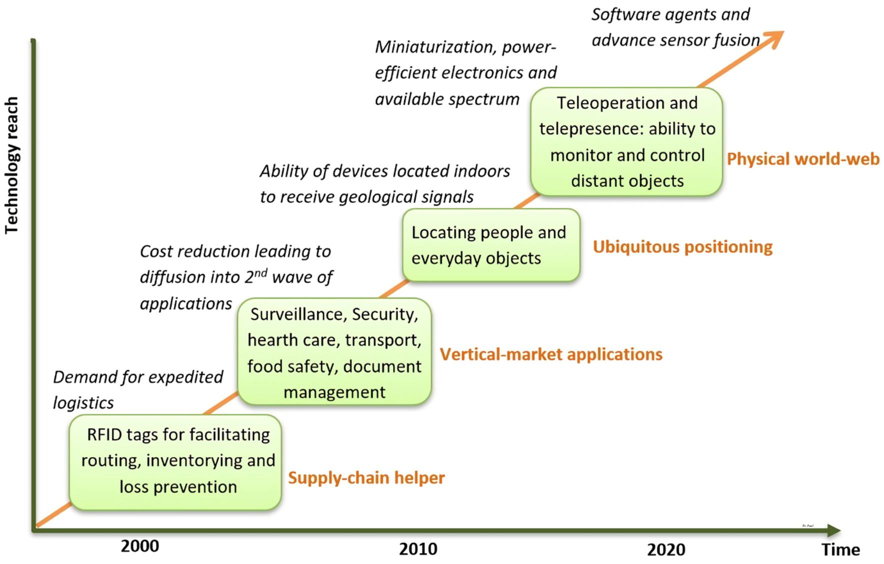

Technology reach
Software agents and advance sensor fusion
Miniaturization, power-efficient electronics and available spectrum
Teleoperation and telepresence: ability to monitor and control distant objects
Physical world-web
Ability of devices located indoors to receive geological signals
Locating people and everyday objects
Ubiquitous positioning
Cost reduction leading to diffusion into 2nd wave of applications
Surveillance, Security, hearth care, transport, food safety, document management
Vertical-market applications
Demand for expedited logistics
RFID tags for facilitating routing, inventorying and loss prevention
Supply-chain helper
2000
2010
2020
Time

**Figure 1.** Technology roadmap for the internet of things (IoT) in last two decades. Source: https://www.nsti.org/directory/org.html?i=2218.

Although RFID's commercial implementations have continued to draw on its barcode-like characteristics to date, RFID provides far more over the long term and aims to be a core technology for the Internet of Things. The quicker check-out is the most significant benefit of a department store using RFID [11].

## 3. Structure and Development of the Greenhouse IoT System

Smart agriculture can be represented as a farm or greenhouse fitted with smart objects; a field network enables information to be transmitted between field-installed devices and a gateway to link the smart agriculture sector to the outside world of the internet. These smart objects make it easier to communicate with farmers with real-time information to manage them. **Figure 2** shows the Conceptual model of the Greenhouse IoT based Monitoring System.

IoT Architecture is a network of various elements, such as sensors, actuators, protocols, cloud servers, which make up an IoT networking system. It typically consists of distinct layers that allow system administrators to monitor and analyze the system working. Although some researchers represent it into more layers and name them accordingly based on their conceptions, in [12], Sethi P. *et al.* differentiated between 3-layers and 5-layers IoT architecture, and in [13] M. Burhan *et al.* in their comprehensive survey on IoT elements, Layered Architectures and Security Issues they compared 3-4-5 layers architecture, more than three layers can be particularly applicable in edge or fog computing IoT system, for 4-layer architecture the support layer that secure cloud computing is added whereas for 5-layer architecture, the business layer is added in order to manage data and applications, business and all profits within the system and the network layer is divide into two layers namely; the transport and processing layers. Since the system can have other particular layers between the physical and the network layers, the general architecture of the IoT system is described by three layers are physical, network, and device layers.

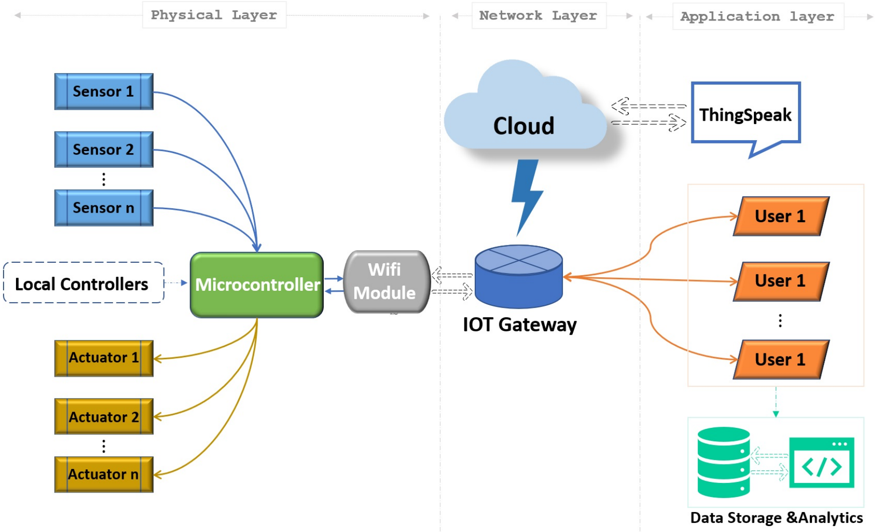


**Figure 2.** Conceptual model of the IoT based monitoring system.

### 3.1. Physical Layer

The physical layer of the architecture also referred to as the perception layer, consists of physical objects that include node sensors and other connected devices that remotely acquire real-time data as per the needs of the system for various cultivation functions [12], including edge devices that provides access to the internet and instantly transmits collected data to the cloud through Wi-Fi and lastly actuators that interact with their environment to facilitate proper physical action where it is needed within the system. There are several protocols and common technology for this layer. Some of the common protocols and standards for the physical layer include Ethernet, Zigbee [7], Bluetooth Low Energy (BLE) [14], WiMAX, IEEE 802.15.4, IEEE802.11ah, RFID, Cellular (GPRS/2G/3G/4G/5G) [15], LTE-MTC, CDMA and so on, all that is adapted to variable applications and network points and are used for interconnecting computers and other mobile devices within the system.

### 3.2. Network Layer

The network layer as the intermediary layer is the collection of all hardware and protocols that comprise all network communications, including switching and routing, protocols and translation between them, network-level security, and all to ensure the reliability of the network that takes place in the IoT system [16], data collected from physical layer devices need to be transferred and processed, that's where the network layer involves in linking these devices and transfer data to other smart objects, such as servers and network devices through the intermediary. The intermediary portal is typically an internet gateway located at the edge of the connected devices, sometimes containing a proxy server. The data is gathered and often processed before being transmitted to the end-user over the internet [17]. The network layer is subdivided into two sublayers; the packet encapsulation layer that forms the packets and the routing layer for the transmission, all with common protocols that include RPL (Routing Protocol for Low-Power and Lossy Network) [18], 6LoWPAN (IPv6 Low Power Personal Area Network) [19], CARP (Channel-Aware Routing Protocol) [20] and many more which allow network layer to forms the packets and performs packet transmission from source to destination.

### 3.3. Application Layer

The application layer is the third layer of the IoT framework. The layer that describes all applications that use IoT technologies, enabling the deployment of the IoT, assuring the communication between users and the system referred to at this moment as the Human Machine Interface (HMI) in so-called software as a Service (SaaS) [13] [21] and is responsible for providing application-specific services to the user, for the implementation of a smart greenhouse where, for example, users can press the button on the app to turn on or off the fan. This layer is of considerable significance as functions such as monitoring, system management, and analytics are all implemented in this layer. The application layer consists of both the ZigBee Device Object (ZDO) [22] and the application support sublayer (APS) [23], which is a layer in Zigbee that provides an interface between network layer and application for providing the necessary services like a function call and API to the users whenever the ZigBee standard is used in the physical layer. Common Standard protocols in this layer include MQTT, AMQP, CoAP, HTTP/2, XMPP, WebSocket, and many more that provides communication to the end-user.

## 4. MQTT Protocols and Its Application in Greenhouse Monitoring

The smart system differs from its conventional counterpart by how the ordinary automatic system remains in off mode in the case of a failure. In contrast, the Smart system can communicate to other systems if it detects any difficulties and, if required, reports a fault to the user or instantly calls for assistance.

But any such instance of communication is only possible when there is a means of communication, a specific "language" that all the devices in a given IoT environment will share and be able to use. In the Internet of Things, the medium is established by protocols designed explicitly for connected device communication.

Some protocols allow customers to publish messages in the bus-based architecture as in **Figure 3**, for a specific topic that is distributed directly to subscribers of that topic where there is no centralized broker [24].

Hardware will be considered useless without IoT protocols in the IoT technology environment because the IoT protocols allow it to exchange information in an organized and reasonable fashion [9]; some of the standard IoT protocols are CoAP, AMQP, MQTT, XMPP, WebSocket, and many more.

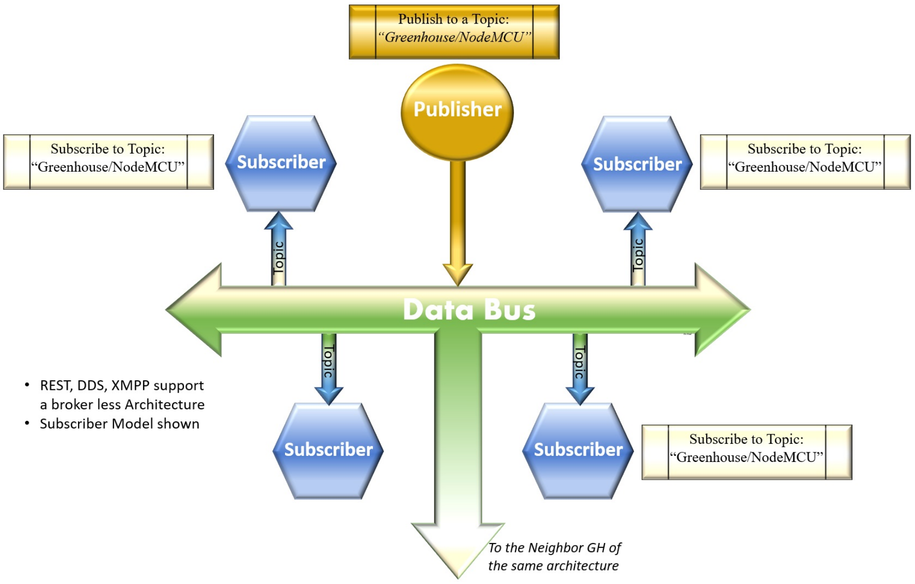


**Figure 3.** Bus-based architecture for data exchanging protocols within the IoT system.

In this paper, a lightweight publish-subscribe-based messaging protocol, MQTT (Message Queuing Telemetry Transport) usage is considered, architected for service terminals at the client terminal through a central broker. MQTT is a frequently used application layer protocol for transferring data between IoT devices, and it uses a TCP/IP-based connection setup process [25].

MQTT brokers are considered to be the key component of any IoT framework based on MQTT since they provide several services to clients. MQTT Publisher publishes messages to the MQTT broker topic; Clients connect to this broker, which then facilitates contact between the two devices; the destination acts as an MQTT client that publishes messages and writes each log as a message; also, every device may subscribe to or register to specific topics. Whenever a client posts a message on a subscribed topic, the broker forwards the message to every subscribed client [26]. Hence MQTT was named data-centric communication or broker-based services (**Figure 4**).

MQTT protocol was released by IBM in the late 1990's and it's now adopted by IoT Hub, a cloud service of Baidu platform for enabling security in two-way communication between smart devices and by Facebook messenger to maintain persistent connection [27]. To ensure that all published data are smoothly received, many levels of QoS (Quality of Services) are supported [28]; those three levels are:

- QoS-0 (at most once): the fastest and reliable that guarantees a message is delivered at least once, but the confirmation is not received whenever the publication is sent.
- QoS-1 (at least once): guarantees that a message is sent to the receiver at least once, duplicates may be received.
- QoS-2 (exactly once): The highest, safest, and most reliable level, guarantees that the intended recipients receive each message only once with the most bandwidth-consuming, thus the slowest. Duplicates are tested to ensure that the message is sent only once. The higher the QoS level, the higher the process becomes complex.

HTTP was the most common and commonly used transfer protocol, but MQTT has rapidly gained popularity over the last few years. Developers have to pick between them wisely in the IoT systems design; **Table 1** depicts the major difference between the two.

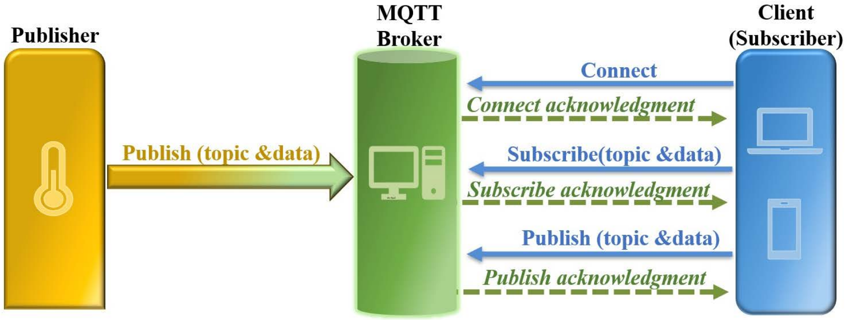


**Figure 4.** MQTT clients-broker, publish/subscribe model architecture.

**Table 1.** Major differences between MQTT protocol and HTTP protocol.

| HTTP Transport Protocol | MQTT Transport Protocol |
|---|---|
| Document centric | Data-centric |
| Request-response paradigm | Publish-Subscribe paradigm |
| Higher power requirement | low power requirement |
| IoT devices are always ready to receive communication | IoT devices choose when to receive communication |
| Directly communication between IoT agent and IoT devices | Indirectly communication between IoT agent and IoT devices |
| Slow throughput | Faster throughput |

MQTT protocol has a relatively short specification for Complexity and Message Size, just CONNECT, PUBLISH, SUBSCRIBE, UNSUBSCRIBE and DISCONNECT types that are meaningful for usage. Table 2 shows the common MQTT implementation.

## 5. System Design

This study proposes an IoT-based Greenhouse Monitoring that contributes to the various range of agricultural greenhouse; the system not only allows intelligent monitoring of the internal climate and working conditions of the greenhouse but also enables the supervisor to control the greenhouse over the internet wherever he/she is through the interface.

In the proposed model's architecture, the data collected experience various stages during their transition from sensors to interfaces and occasionally to actuators. These stages have a significant impact on technologies used for IoT context. These stages are divided into six main data flow phases: sensing, communication/transport, processing, storage, analysis, and actuation. Figure 5 illustrates both stages and phases in the Layered generic Architecture of the proposed system.

Using smart farming by IoT technology helps farmers minimize waste generation and improve productivity. Smart greenhouse equipped with an IoT system intelligently monitors and controls internal climate and reducing the need for human intervention.

Following are the leading hardware and software requirements:

1) Hardware Requirements

- *DHT*11 *& DHT*22 is a basic, low-cost digital multifunctional sensor that uses a capacitive sensor module and a negative temperature coefficient (NTC) thermistor to measure the surrounding air and delivers a digital signal of relative humidity and temperature relative to the output pin.
- *Arduino UNO* is an open-source Microchip ATmega328P based microcontroller board developed by Arduino. cc and is programmable with the Arduino IDE.
- *NodeMCU ESP*8266 is an open-source Lua based firmware and development board designed especially for IoT Applications; it runs on the ESP8266 Wi-Fi SOC designed by Espressif with ESP-12 module-based hardware.

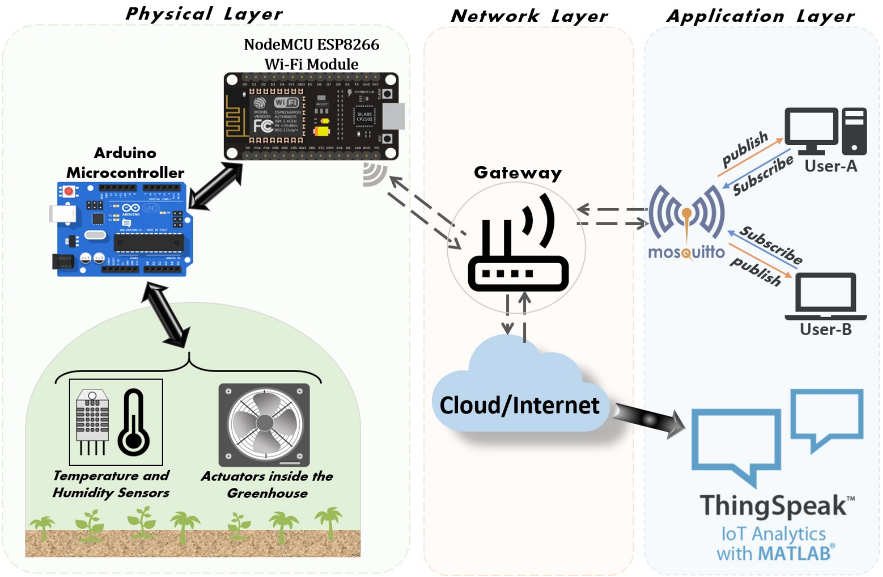


Figure 5. Layered generic architecture of the proposed IoT agriculture greenhouse monitoring system.

2) Software Requirements

- *The Arduino IDE* (Integrated Development Environment) is a cross-platform computer application that is used to write, edit and upload programs to Arduino and other compatible boards; it is written in functions from C and C++ programming.
- *ThingSpeak*TM is an open-source IoT (Internet of Things) analytics platform that allows users to collect, visualize and analyze live data streams in the cloud. It forms an API (Application Programming Interface) for storing and retrieving live data from sensor Nodes using the HTTP and MQTT protocol over the internet or over the LAN (Local Area Network).
- *Adafruit IO* is a cloud service/platform designed to display, respond, and interact with an IoT project's data, run it and manage it in private mode without being controlled by the user.

## Experimental Setup and System Configuration

The configuration of the greenhouse Data Acquisition system consists of three parts, connecting the physical parts, configuring the ThingSpeak platform, and configuring the Adafruit IO broker and the laptop computer used for testing has the following properties; Lenovo Yoga laptop with 512 GB hard drive and 16 GB of RAM. The network bandwidth is provided by a Lenovo android Phone with 128 GB of storage and 6 GB of RAM. The network should not be a limiting factor for the research.

Connecting and configuring the system's physical part is the easy part of the system, but where more mistakes are to be made without caution, > connecting sensor to Arduino and NodeMCU ESP8266 > connect it to the internet, the algorithm to connect it on the internet is shown in Table 3.

Table 2. Common MQTT Implementation platform.

| Implementation | Developer | Type of protocol Service | Programming Language |
|---|---|---|---|
| HiveMQ | HiveMQ (Germany) | Broker and Client (Separately) | Java |
| Adafruit IO | Adafruit (US) | Client and Broker | Ruby, Node.js, Python |
| Mosquito | Eclipse (UK) | Client and Broker | C |
| EMQ X | EMQ (China) | Broker | Erlang |
| Paho MQTT | Eclipse (UK) | Client | C, C++, Java, Rust, Ruby, Node.js, Python |
| MQTT-C | Liam Bindle (Canada) | Client | C |

Table 3. An algorithm to connect NodeMCU ESP8266 to Wi-Fi access point.

```
wifi. setmode (wifi. STATION)
station_cfg = {}
station_cfg. ssid = "SSID"
station_cfg.pwd = "password"
station_cfg. save = false
wifi.sta.config (station_cfg)
```

Configuring ThingSpeak, it's all about visiting ThingSpeak.com; create a free MathWorks account or sign in to ThingSpeak using an existing account if you have it > Select the ThingSpeak channel you want your data to stream into; if you are new, have to create a new channel > enter the name and description > add fields then save, once created it you got the Channel ID, Author name and access control type and keys as in Figure 6.

Deploying the MQTT communication protocol was done by configuring the Adafruit IO broker and connecting it to the sensor and client > Set Adafruit IO username in the IO_USERNAME define, and Adafruit IO key in the IO_KEY define, sets up the temperature and humidity feeds, Table 4 shows the connection details of Adafruit IO MQTT broker. The Adafruit IO MQTT API only supports QoS level 0 (at most once) and QoS level 1 (at least once).

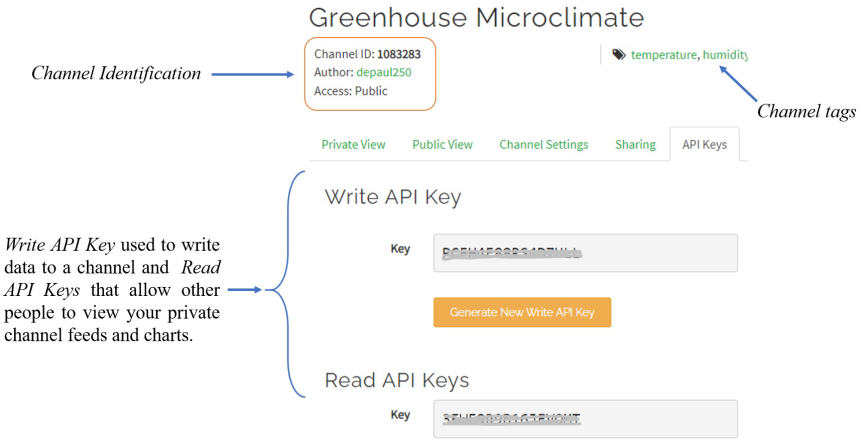
## 6. Experimental Results and Discussions

### 6.1. Without MQTT Protocol

In Figure 7, Arduino IDE's serial monitor shows both the real-time data of temperature and relative humidity after the connection to the internet is established smoothly. Here the message is "Send to the IoT platform (ThingSpeak)", indicate that data are being sent to the internet.

After processing the data, air temperature and humidity data for environmental parameters were sent to the ThingSpeak platform in time series cycles using the HTTP protocol. The ThingSpeak website works as a server, and NodeMCU was as the client.

Figure 8 shows the graphical output at the ThingSpeak server, which can be accessed only after signing in to the ThingSpeak website with a username and password that has been created, and the cloud provides a reliable output. With the Channel ID and Read API key, environmental parameters data can be used



Greenhouse Microclimate

Channel ID: 1083283
Author: depaul250
Access: Public

*Channel Identification*

temperature, humidity

*Channel tags*

Write API Key

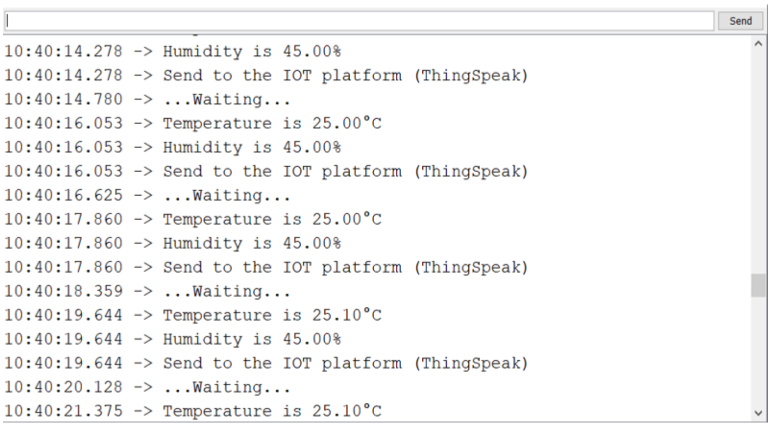
Key: [redacted]

Read API Keys

Key: [redacted]

*Write API Key* used to write data to a channel and *Read API Keys* that allow other people to view your private channel feeds and charts.

Figure 6. ThingSpeak platform configuration.



```
10:40:14.278 -> Humidity is 45.00%
10:40:14.278 -> Send to the IOT platform (ThingSpeak)
10:40:14.780 -> ...Waiting...
10:40:16.053 -> Temperature is 25.00°C
10:40:16.053 -> Humidity is 45.00%
10:40:16.053 -> Send to the IOT platform (ThingSpeak)
10:40:16.625 -> ...Waiting...
10:40:17.860 -> Temperature is 25.00°C
10:40:17.860 -> Humidity is 45.00%
10:40:17.860 -> Send to the IOT platform (ThingSpeak)
10:40:18.359 -> ...Waiting...
10:40:19.644 -> Temperature is 25.10°C
10:40:19.644 -> Humidity is 45.00%
10:40:19.644 -> Send to the IOT platform (ThingSpeak)
10:40:20.128 -> ...Waiting...
10:40:21.375 -> Temperature is 25.10°C
```

Figure 7. Evolution of the output data in serial monitor.

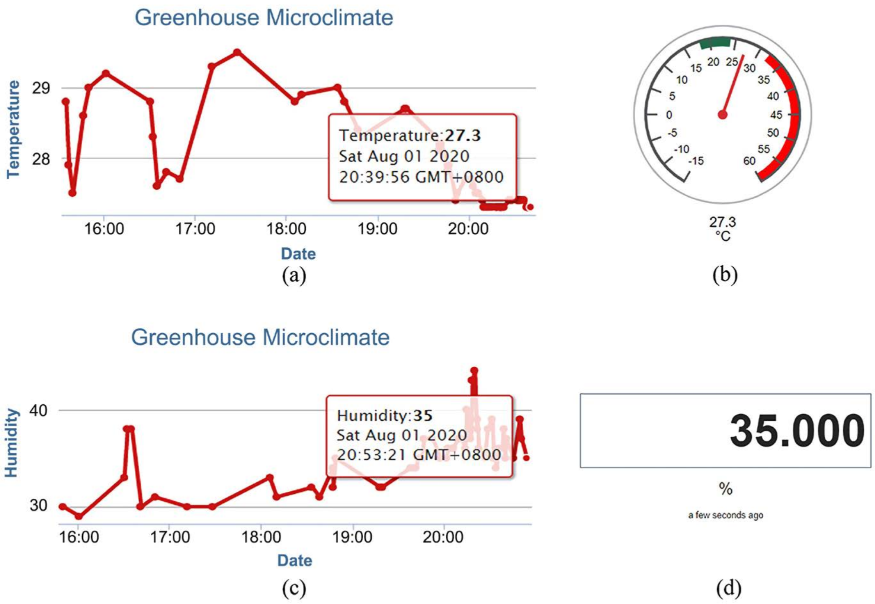


Figure 8. Temperature and Humidity in ThingSpeak (a) temperature data vs. the time, (b) gauge showing the temperature variations, (c) temperature data vs. the time, and (d) the numerical relative humidity output.

Table 4. MQTT connection details.

| Host | io.adafruit.com |
|---|---|
| Secure (SSL) Port | 8883 |
| Insecure Port | 1883 |
| MQTT over WebSocket | 443 |
| Username | Adafruit IO Username |
| Password | Adafruit IO Key |

for MATLAB analysis. Figure 9 shows the performances in MATLAB during the run process of data acquisition in ThingSpeak. As the system work through the internet, for unstable internet, data pause for a while and continues when the internet is stable with the big change in magnitude of temperature as in (a), all data acquired can be processed directly in MALTLAB as shown in (b) to (d).

## 6.2. With MQTT Protocol

From Figure 5, it is clear that the gateway and the internet lie between the controller part (in the physical layer) and interface (application layer) and become the center of the whole IoT system, all connected wirelessly. MQTT just specifies the transport, and mostly the application layer, which determines how data is handled, transferred, and possibly stored, and finally, how clients are authorized in the system.

MQTT is a perfect communication protocol to share small quantities of data between IoT devices. In this study, it is to publish temperature and humidity readings from a DHT11 sensor with the NodeMCU ESP8266 to various MQTT topics so that any device or any platform that supports MQTT as clients can subscribe to such topics and obtain real-time data readings, as shown in Figure 10.

It requires building web-based dashboards that can visualize the real-time incoming data; using Adafruit IO's MQTT broker is a great way to connect the IoT system over the internet. From any modern web browser, dashboards enable users to visualize data and control Adafruit IO connected projects. Sensor values are sent to IO feeds as numeric values, such as 24˚C for temperature. When sent to a topic, it can be a raw number or just a string value; the format of a topic is as follow: Username/feeds/temperature and Username/feeds/humidity.

The relationship between humidity and temperature inside the greenhouse is clarified in the first chapter. The measurements' results confirm the concept that lowering relative humidity raises the temperature of the air and vice versa. Figure 11 displays a graph of air temperature variations and relative humidity taken by a sensor in two hours and published to the broker. The comparison between the values of air temperature and relative humidity indicates how the variables are related together. For example, when considering the point at 4:26:00PM, drops in relative humidity are level correspond at the same time to the rise of air temperature level.

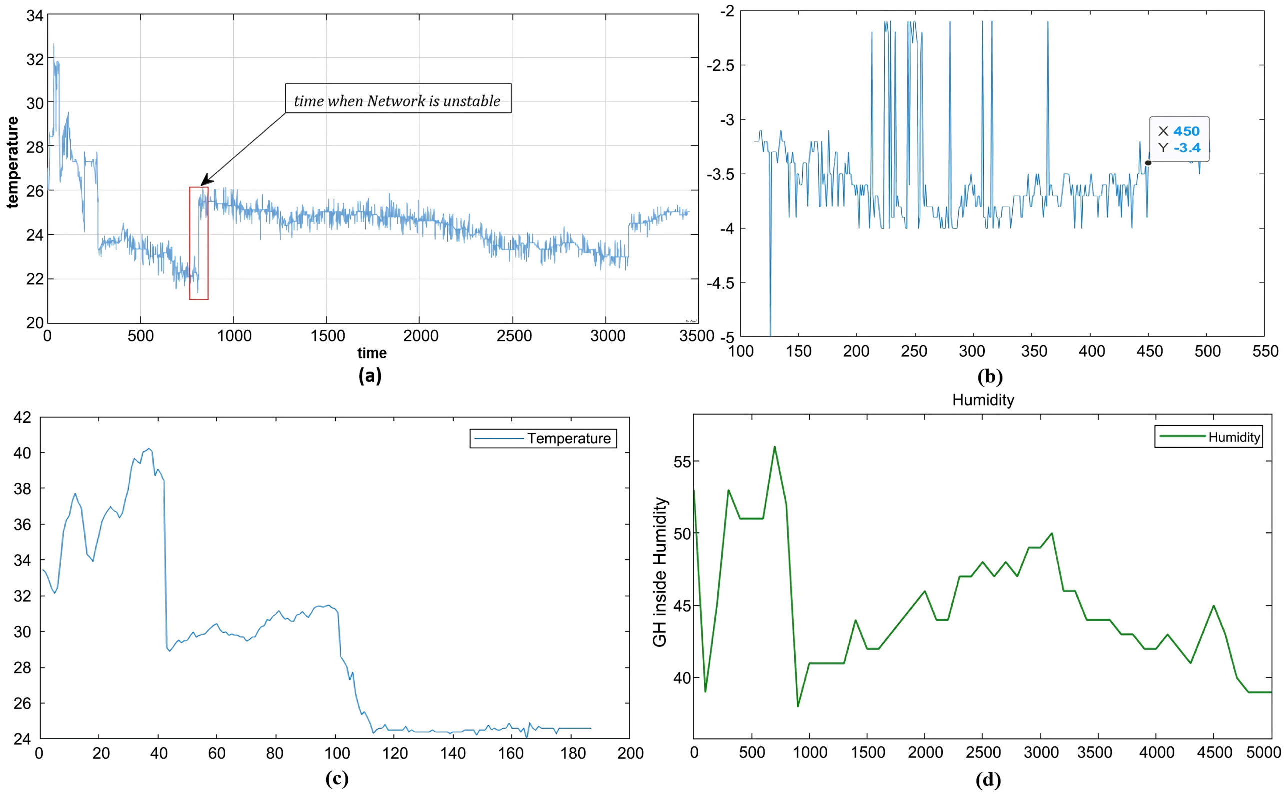
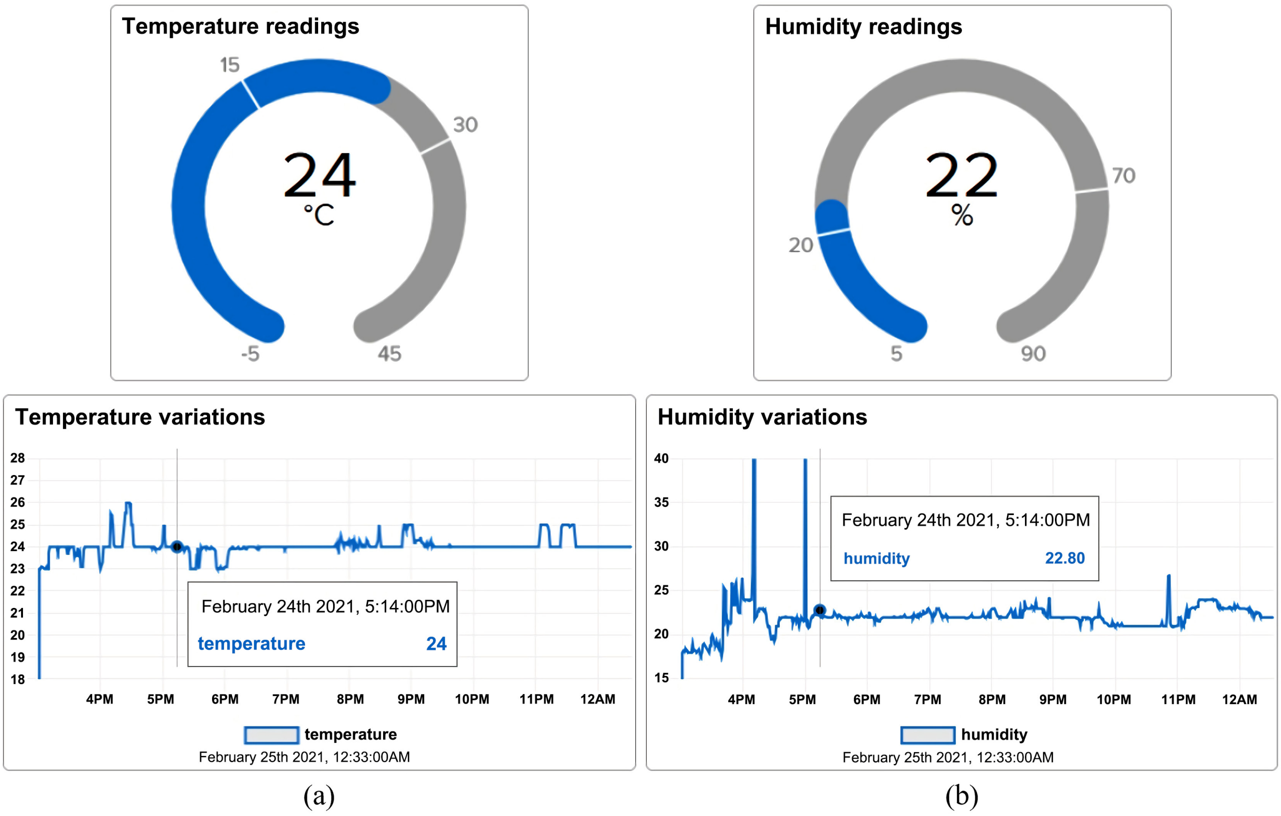


**Figure 9.** Air temperature and relative humidity data MATLAB visualizations from ThingSpeak.


Temperature readings

24
°C

Humidity readings

22
%

Temperature variations

February 24th 2021, 5:14:00PM
temperature 24

February 25th 2021, 12:33:00AM

Humidity variations

February 24th 2021, 5:14:00PM
humidity 22.80

February 25th 2021, 12:33:00AM

**Figure 10.** Real-time data readings and variations temperature (a) and humidity (b) published to the client interface.

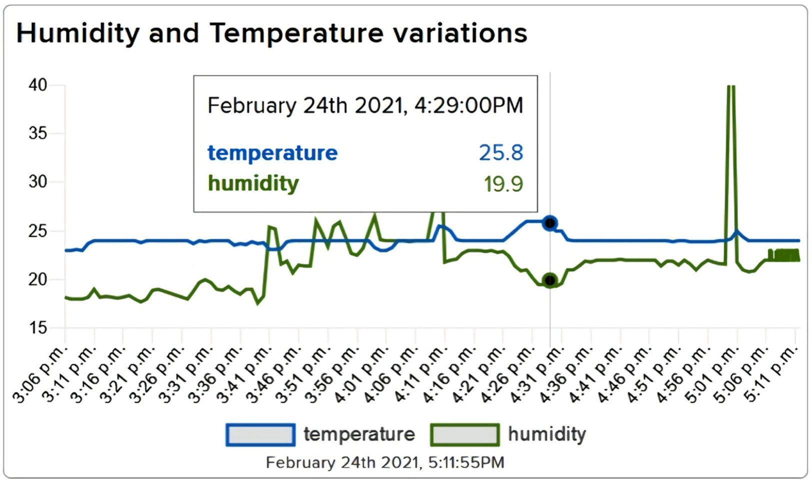
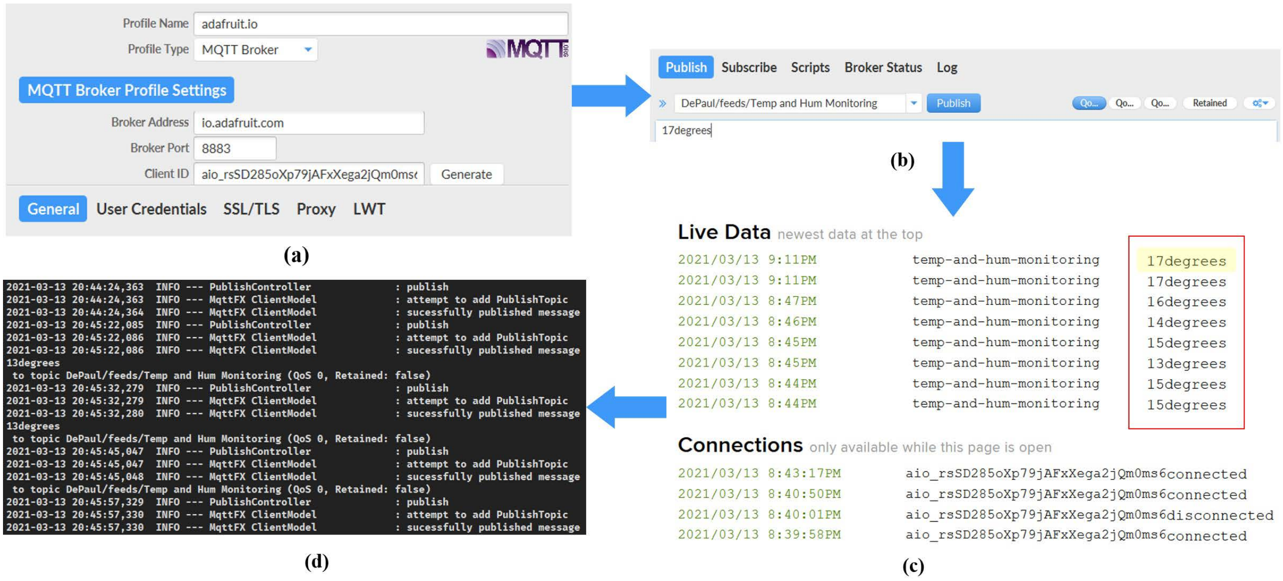


**Figure 11.** Comparison of temperature and humidity variations of two hours (3:06 pm-5:11 pm).



Profile Name: adafruit.io
Profile Type: MQTT Broker

MQTT Broker Profile Settings

Broker Address: io.adafruit.com
Broker Port: 8883
Client ID: aio_rsSD285oXp79jAFxXega2jQm0ms( Generate

(a)

DePaul/feeds/Temp and Hum Monitoring

17degrees

(b)

Live Data newest data at the top

```
2021/03/13 9:11PM     temp-and-hum-monitoring     17degrees
2021/03/13 9:11PM     temp-and-hum-monitoring     17degrees
2021/03/13 8:47PM     temp-and-hum-monitoring     16degrees
2021/03/13 8:46PM     temp-and-hum-monitoring     14degrees
2021/03/13 8:45PM     temp-and-hum-monitoring     15degrees
2021/03/13 8:45PM     temp-and-hum-monitoring     13degrees
2021/03/13 8:44PM     temp-and-hum-monitoring     15degrees
2021/03/13 8:44PM     temp-and-hum-monitoring     15degrees
```

Connections only available while this page is open

```
2021/03/13 8:43:17PM     aio_rsSD285oXp79jAFxXega2jQm0ms6connected
2021/03/13 8:40:50PM     aio_rsSD285oXp79jAFxXega2jQm0ms6connected
2021/03/13 8:40:01PM     aio_rsSD285oXp79jAFxXega2jQm0ms6disconnected
2021/03/13 8:39:58PM     aio_rsSD285oXp79jAFxXega2jQm0ms6connected
```

(c)

```
2021-03-13 20:44:24,363  INFO --- PublishController             : publish
2021-03-13 20:44:24,363  INFO --- MqttFX ClientModel            : attempt to add PublishTopic
2021-03-13 20:44:24,364  INFO --- MqttFX ClientModel            : sucessfully published message
2021-03-13 20:45:22,085  INFO --- PublishController             : publish
2021-03-13 20:45:22,086  INFO --- MqttFX ClientModel            : attempt to add PublishTopic
2021-03-13 20:45:22,086  INFO --- MqttFX ClientModel            : sucessfully published message
13degrees
 to topic DePaul/feeds/Temp and Hum Monitoring (QoS 0, Retained: false)
2021-03-13 20:45:32,279  INFO --- PublishController             : publish
2021-03-13 20:45:32,279  INFO --- MqttFX ClientModel            : attempt to add PublishTopic
2021-03-13 20:45:32,280  INFO --- MqttFX ClientModel            : sucessfully published message
13degrees
 to topic DePaul/feeds/Temp and Hum Monitoring (QoS 0, Retained: false)
2021-03-13 20:45:45,047  INFO --- PublishController             : publish
2021-03-13 20:45:45,047  INFO --- MqttFX ClientModel            : attempt to add PublishTopic
2021-03-13 20:45:45,048  INFO --- MqttFX ClientModel            : sucessfully published message
 to topic DePaul/feeds/Temp and Hum Monitoring (QoS 0, Retained: false)
2021-03-13 20:45:57,329  INFO --- PublishController             : publish
2021-03-13 20:45:57,330  INFO --- MqttFX ClientModel            : attempt to add PublishTopic
2021-03-13 20:45:57,330  INFO --- MqttFX ClientModel            : sucessfully published message
```

(d)

**Figure 12.** MQTT functional testing.

## 6.3. MQTT Test

MQTT protocol as an IoT communication protocol, during the workflow a user have to ensure that the communication between broker and client is reliable as shown in Figure 4, to ensure reliable communication, a test which is an MQTT Functional Testing is carried out using MQTT.fx-1.7.1 in three steps as follows: Establishing the connection and publishing a specified message, subscribe to an MQTT topic and waits for the first message to be posted and then Simulates the publisher disconnecting from the server.

Figure 12 shows the testing of MQTT's broker functionality, the process shown from the configuration of the broker profile and address in (a), after connecting the client to broker, the action of publishing a message through a designated topic is shown in (b), and the results show that the published messages are received well by broker as shown in (c) and then the functionality log shown in (d) demonstrate that all attempts to communicate between client and broker were successful by using QoS-0 and MQTT TLS.

## 7. Conclusion

The climate is one of the factors that affect the growth and production of crops in the agricultural greenhouse. However, the unpredictable weather conditions arising from climate change further degrades the quality and quantity of production. Deploying IoT devices for real-time monitoring is an added value to solve the problem, looking for a long-term solution to food production in order to optimize performance and productivity. This paper presented the different IoT technologies that are applicable in a greenhouse in order to monitor the environmental parameters. A small IoT prototype was designed to develop an algorithm that acquires environmental data and publishes them to the internet through an MQTT protocol, it also makes for a simple visual interpretation of the parameters of the greenhouse's internal climate. MQTT's functionality makes it an ideal choice for transmitting large amounts of sensed data to analytic systems and to cloud platforms, MQTT functionality testing was carried out in this paper to ensure all data is transferred according to the used QoS. Furthermore, this management system allows users to monitor the controllers' operation mounted within the greenhouse from afar making the farming process more practical, precise, and analytical which will improve agricultural management and production.

## Conflicts of Interest

All the authors read and declare no conflict of interest.

## References

[1] Manonmani, A., Thyagarajan, T., Elango, M. and Sutha, S. (2018) Modelling and Control of Greenhouse System Using Neural Networks. *Transactions of the Institute of Measurement and Control,* **40**, 918-929. https://doi.org/10.1177/0142331216670235

[2] Ka, A.R. and Savitha (2017) E-Farming Using Internet of Things (IoT). *International Journal of Latest Trends in Engineering and Technology*, 419-422.

[3] GSMA's Internet of Things, NB-IOT (2018) The future of farming. https://www.gsma.com/iot/resources/chunghwa-nhr-agriculture-iot-case-study/

[4] Zamora-Izquierdo, M.A., Santa, J., Martínez, J.A., Martínez, V. and Skarmeta, A.F. (2019) Smart Farming IoT Platform Based on Edge and Cloud Computing. *Biosystems Engineering*, **177**, 4-17. https://doi.org/10.1016/j.biosystemseng.2018.10.014

[5] Nawandar, N.K. and Satpute, V.R. (2019) IoT Based Low Cost and Intelligent Module for Smart Irrigation System. *Computers and Electronics in Agriculture*, **162**, 979-990. https://doi.org/10.1016/j.compag.2019.05.027

[6] Daponte, P. *et al.*, (2019) A Review on the Use of Drones for Precision Agriculture. *IOP Conference Series: Earth and Environmental Science*, **275**, Article ID: 012022. https://doi.org/10.1088/1755-1315/275/1/012022

[7] Li, Z., Wang, J., Higgs, R., Zhou, L. and Yuan, W. (2017) Design of an Intelligent Management System for Agricultural Greenhouses Based on the Internet of Things. 2017 *IEEE International Conference on Computational Science and Engineering* (*CSE*) *and IEEE International Conference on Embedded and Ubiquitous Computing* (*EUC*), **2**, 154-160.

[8] Aliev, K., Jawaid, M.M., Narejo, S., Pasero, E. and Pulatov, A. (2018) Internet of Plants Application for Smart Agriculture. *International Journal of Advanced Computer Science and Applications*, **9**, 421-429. https://doi.org/10.14569/IJACSA.2018.090458

[9] Atzori, L., Iera, A. and Morabito, G. (2010) The Internet of Things: A Survey. *Computer Networks*, **54**, 2787-2805. https://doi.org/10.1016/j.comnet.2010.05.010

[10] Ibarra-Esquer, J.E. González-Navarro, F.F., Flores-Rios, B.L., Burtseva, L. and Astorga-Vargas, M.A. (2017) Tracking the Evolution of the Internet of Things Concept across Different Application Domains. *Sensors* (*Switzerland*), **17**, 1-24. https://doi.org/10.3390/s17061379

[11] ITU (2005) ITU Internet Reports: the Internet of Things. International Telecommunication Union, Genev.

[12] Sethi, P. and Sarangi, S.R. (2017) Internet of Things: Architectures, Protocols and Applications, **2017**, Article ID: 9324035. https://doi.org/10.1155/2017/9324035

[13] Burhan, M., Rehman, R.A., Khan, B. and Kim, B.S. (2018) IoT Elements, Layered Architectures and Security Issues: A Comprehensive Survey. *Sensors* (*Switzerland*), **18**, Article No. 2796. https://doi.org/10.3390/s18092796

[14] MacKensen, E., Lai, M. and Wendt, T.M. (2012) Bluetooth Low Energy (BLE) Based Wireless Sensors. *Sensors*, 2012 *IEEE*, Taiwan, 28-31 Octorber 2012, 3-6. https://doi.org/10.1109/ICSENS.2012.6411303

[15] Antony, A.P., Leith, K., Jolley, C., Lu, J. and Sweeney, D.J. (2020) A Review of Practice and Implementation of the Internet of Things (IoT) for Smallholder Agriculture. *Sustainability*, **12**, Article No. 3750. https://doi.org/10.3390/su12093750

[16] Wheelus, C. and Zhu, X. (2020) IoT Network Security: Threats, Risks, and a Data-Driven Defense Framework. *IoT*, **1**, 259-285. https://doi.org/10.3390/iot1020016

[17] Villa-Henriksen, A., Edwards, G.T.C., Pesonen, L.A., Green, O. and Sørensen, C.A.G. (2020) Internet of Things in Arable Farming: Implementation, Applications, Challenges and Potential. *Biosystems Engineering*, **191**, 60-84. https://doi.org/10.1016/j.biosystemseng.2019.12.013

[18] Ryan, P. and Watson, R. (2017) Research Challenges for the Internet of Things: What Role Can OR Play? *Systems*, **5**, Article No. 24. https://doi.org/10.3390/systems5010024

[19] Dachyar, M., Zagloel, T.Y.M. and Saragih, L.R. (2019) Knowledge Growth and Development: Internet of Things (IoT) Research, 2006-2018. *Heliyon*, **5**, e02264. https://doi.org/10.1016/j.heliyon.2019.e02264

[20] Basagni, S., Petrioli, C., Petroccia, R. and Spaccini, D. (2012) Channel-Aware Routing for Underwater Wireless Networks. 2012 *Oceans-Yeosu*, Yeosu, Korea (South), 21-24 May 2012. https://doi.org/10.1109/OCEANS-Yeosu.2012.6263538

[21] Symeonaki, E., Arvanitis, K. and Piromalis, D. (2020) A Context-Aware Middleware Cloud Approach for Integrating Precision Farming Facilities into the IoT toward Agriculture 4.0. *Applied Sciences*, **10**, Article No. 813. https://doi.org/10.3390/app10030813

[22] Geng, L. and Dong, T. (2017) An Agricultural Monitoring System Based on Wire-

less Sensor and Depth Learning Algorithm. *International Journal of Online and Biomedical Engineering*, **13**, 127-137. https://doi.org/10.3991/ijoe.v13i12.7885

[23] Chen, J., Chen, Y. and Chung, S. (2013) Implementation of a Greenhouse Crop Remote Monitoring System with IOT Technology. Taiwan, 23-25 October 2013, 1-5.

[24] Vermesan, O. and Friess, P. (2014) Internet of Things Applications: From Research and Innovation to Market Deployment. River Publishers, Denmark, 7-142.

[25] Haripriya, A.P. and Kulothungan, K. (2019) Secure-MQTT: An Efficient Fuzzy Logic-Based Approach to Detect DoS Attack in MQTT Protocol for Internet of Things. *EURASIP Journal on Wireless Communications and Networking*, **2019**, Article No. 90. https://doi.org/10.1186/s13638-019-1402-8

[26] Suciu, V., Balaceanu, C.M., Anwar, M., Pasat, A., *et al.* (2018) Analysis of Agriculture Sensors Based on IoT. 2018 *International Conference on Communications* (*COMM*), Bucharest, Romania, 14-16 June 2018, 423-427. https://doi.org/10.1109/ICComm.2018.8484772

[27] Nastase, L. (2017) Security in the Internet of Things: A Survey on Application Layer Protocols. 2017 21*st International Conference on Control Systems and Computer Science* (*CSCS*), Bucharest, Romania, 29-31 May 2017, 659-666. https://doi.org/10.1109/CSCS.2017.101

[28] Singh, T.A. and Chandra, J. (2018) IOT Based Green House Monitoring System. *Journal of Computer Science*, **14**, 639-644. https://doi.org/10.3844/jcssp.2018.639.644