Sensors and Materials, Vol. 31, No. 9 (2019) 2703–2717
MYU Tokyo

S & M 1967

Development of Remote Sensing Data Collection and Management Platform via Internet

Chun-Kuan Wu and Hsiung-Cheng Lin*

Department of Electronic Engineering, National Chin-Yi University of Technology, Taichung 41170, Taiwan

(Received January 31, 2019; accepted March 25, 2019)

Keywords: MQTT, Wi-Fi module, GUI, MySQL

Recently, the technology of the Internet of Things (IoT) has been widely used in many disciplines such as remote sensing and control via the Internet in various industries. However, it is still difficult to find solutions directly from an appropriate textbook for learners. For this reason, this paper aims to provide a platform for remote sensing data collection from different locations. The proposed system presents a case study to collect and monitor sensed data, e.g., temperature and humidity, with Wi-Fi modules. The human interface using Visual C# is designed to control the Wi-Fi module that can transmit data up to a distance of 200 m. The collected data can be transmitted to the server via the Message Queueing Telemetry Transport (MQTT) protocol. Once the data is received, it is then uploaded to the MySQL database. Therefore, the historical data from the cloud can be viewed and monitored on line by the PHP dynamic data management system. Experimental results verify that the proposed platform is feasible for practical applications in terms of robust, stable, and fast responses.

1. Introduction

Nowadays, techniques using cyber-physical systems, the Internet of Things (IoT), cloud computing, and so forth are widely applied to industry 4.0-based manufacturing systems in various industries.(1–3) With the rapid development of wireless communication technologies, especially Wi-Fi technology, the Internet can be connected with a WLAN network and a wireless access point up to a 100 m range.(4–7) Moreover, cloud computing has become an increasingly important technology recently. Therefore, the system integration with these technologies is an indispensable issue currently.

There have been many related research studies based on the above techniques in recent years. For example, a room monitoring system using a ZigBee communication module and a sensor module was proposed.(8) It can monitor temperature, humidity, O_2 and CO_2 levels, and atmospheric pressure so that the indoor air quality can be monitored and efficiently controlled using a ZigBee module. In 2016, a home automation system was developed on the basis of ESP8266 with Wi-Fi, TCP/IP stack, and microcontroller capability.(9) Its mobile or web app communicates through the microprocessor via the Message Queueing Telemetry Transport

*Corresponding author: e-mail: hclin@ncut.edu.tw
https://doi.org/10.18494/SAM.2019.2320

(MQTT) protocol. However, for the aforementioned techniques, the data management mechanism with the user interface still requires further development.

The purpose of this study is to provide a set of guidelines for building a platform that can achieve multiple remote sensing data collection and monitoring via the Internet using Wi-Fi modules and the MQTT protocol.(10–18) The proposed system allows different users to log in their own account and perform real-time monitoring, collection, and management of data. The Visual C#-based data acquisition system and graphical user interface (GUI) are designed to receive real-time data from remote temperature and humidity sensors.(19–24)

2. System Structure

In this study, two clients (Company A and Company B) are assumed to be working with the same server at the same time. The system structure is shown in Fig. 1 to mainly consist of Wi-Fi modules, a Visual C# user interface, a data acquisition system in the server, and the MySQL database. Note that the Wi-Fi modules Linkit 7688 and Esp8266 represent Company A and Company B, respectively. Also, the Linkit 7688 and Esp8266 modules are combined with different temperature and humidity sensor modules.

On the basis of the MQTT protocol, the collected data (temperature and humidity) can be transmitted to the server from Wi-Fi modules via the Internet. The server subscribes all device topics and collects the data to be uploaded to the MySQL database. The GUI is designed to show real-time data.

The hardware modules used to set up the proposed system structure are listed as follows.

a. Synology DS215+ is a high-performance server with which a client can communicate using Php, MySQL, MQTT Broker, and so forth.

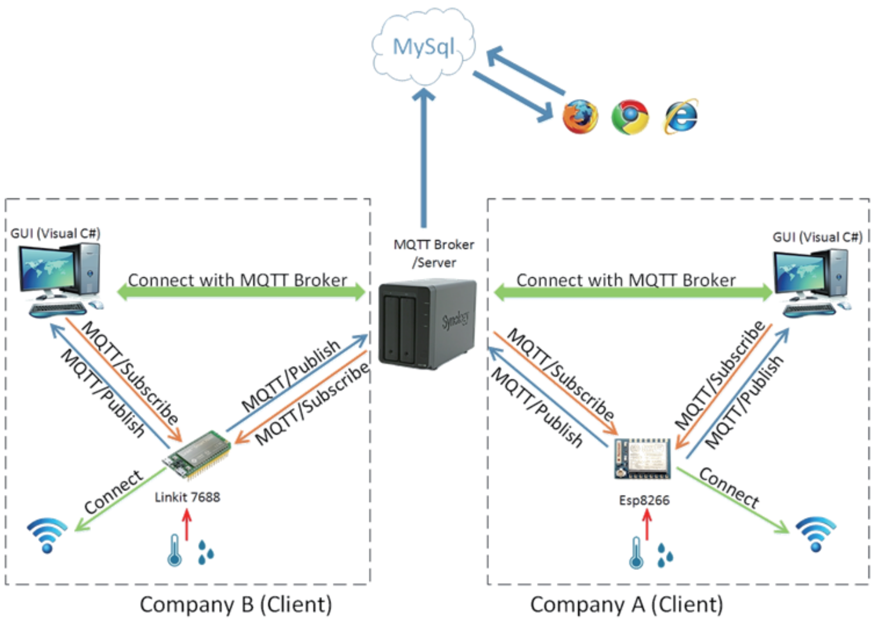
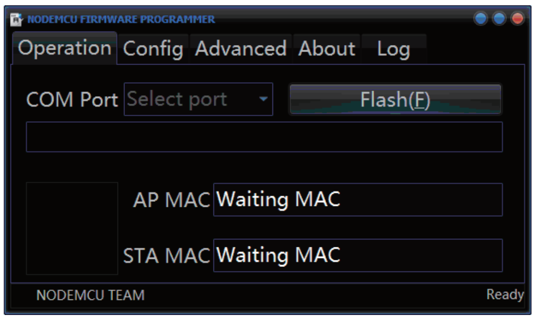


Fig. 1. (Color online) System structure.

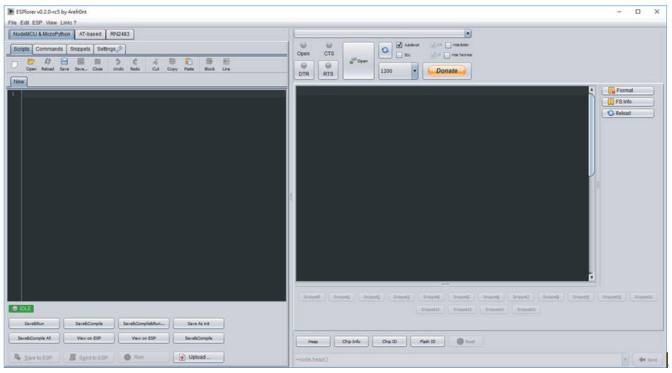
b. Esp 8266-07 is an integrated Wi-Fi chip with a 32-bit MCU, standard digital peripheral interfaces, ADC, and so forth.
c. Linkit Smart 7688 is a Wi-Fi module with GPIO, I^{2}C, I^{2}S, SPI, UART, PWM, an ethernet socket, a 32 MB flash, a 128 MB DDR2 memory, and so forth.
d. DHT11 is a temperature and humidity sensor complex with a calibrated digital signal output. The communication process is in the single-bus data format used between the MCU and the DHT11 sensor.
e. SI7021 is a humidity and temperature sensor with an analog-to-digital converter, signal processing, calibration data, and an I^{2}C interface.

The software packages used to control the proposed system structure are listed as follows.

a. The MQTT protocol is a machine-to-machine (M2M) and IoT connectivity protocol based on TCP/IP for building the network connectivity.
b. NodeMCU is an open source IoT platform that includes firmware running on the ESP8266 Wi-Fi SoC from Espressif Systems, as shown in Fig. 2(a). The firmware uses the Lua programming language. The development environment of NodeMCU is ESPlorer, which is an integrated development environment (IDE) for ESP8266 developers, as shown in Fig. 2(b).
c. OpenWrt is an embedded operating system based on Linux. In Linkit Smart 7688, it can be developed and programmed with Python software, as shown in Fig. 2(c).

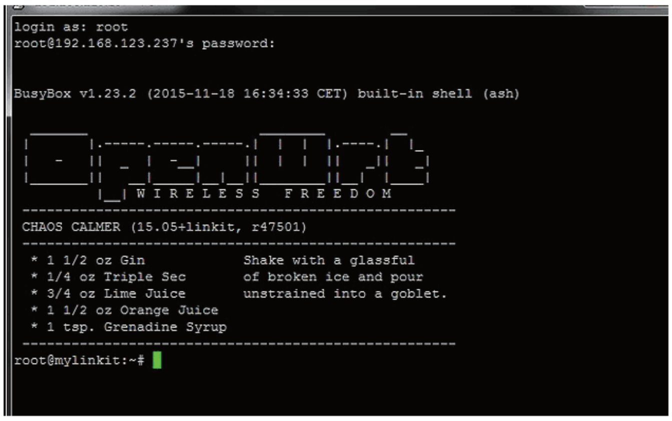


(a)



(b)

Fig. 2. (Color online) Applied software packages. (a) NodeMCU V0.9.5. (b) Interface of ESPlorer.

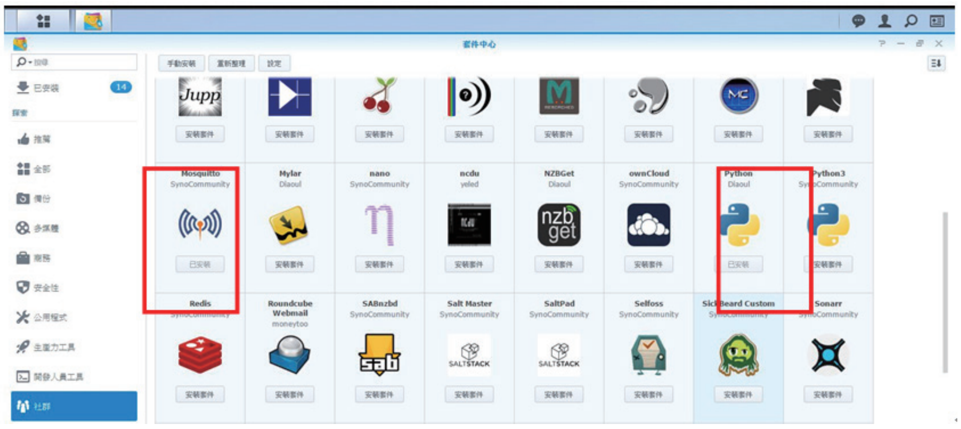


```
login as: root
root@192.168.123.237's password:


BusyBox v1.23.2 (2015-11-18 16:34:33 CET) built-in shell (ash)

  _______                     ________        __
 |       |.-----.-----.-----.|  |  |  |.----.|  |_
 |   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 -----------------------------------------------------
 CHAOS CALMER (15.05+linkit, r47501)
 -----------------------------------------------------
  * 1 1/2 oz Gin            Shake with a glassful
  * 1/4 oz Triple Sec       of broken ice and pour
  * 3/4 oz Lime Juice       unstrained into a goblet.
  * 1 1/2 oz Orange Juice
  * 1 tsp. Grenadine Syrup
 -----------------------------------------------------
root@mylinkit:~#
```

(c)



(d)

Fig. 2. (Color online) (Continued) Applied software packages. (c) OpenWrt in Linkit Smart 7688. (d) Mosquitto and Python packages.

d. MQTT using the Synology DS215+ server can install the Mosquitto package to execute the MQTT protocol and Python package after logging in to the web server, as shown in Fig. 2(d).

3. System Hardware

3.1 Esp8266 with peripheral circuit

Esp8266 with a peripheral circuit is shown in Fig. 3. The USB type-C adapter connects with the USB charger, and LD1117 (low drop fixed and adjustable positive voltage regulator) receives 5 V from the USB charger and thus supplies 3.3 V to Esp8266 and DHT11 (temperature and humidity module).

GPIO 12 in Esp8266 connects with DHT11 (pin 2) to receive the data. To display the communication status, Esp8266 connects two LED lights; the red LED light indicates a successful connection between Esp8266 and the MQTT broker, and the yellow LED light shows that Esp8266 is transmitting the data.

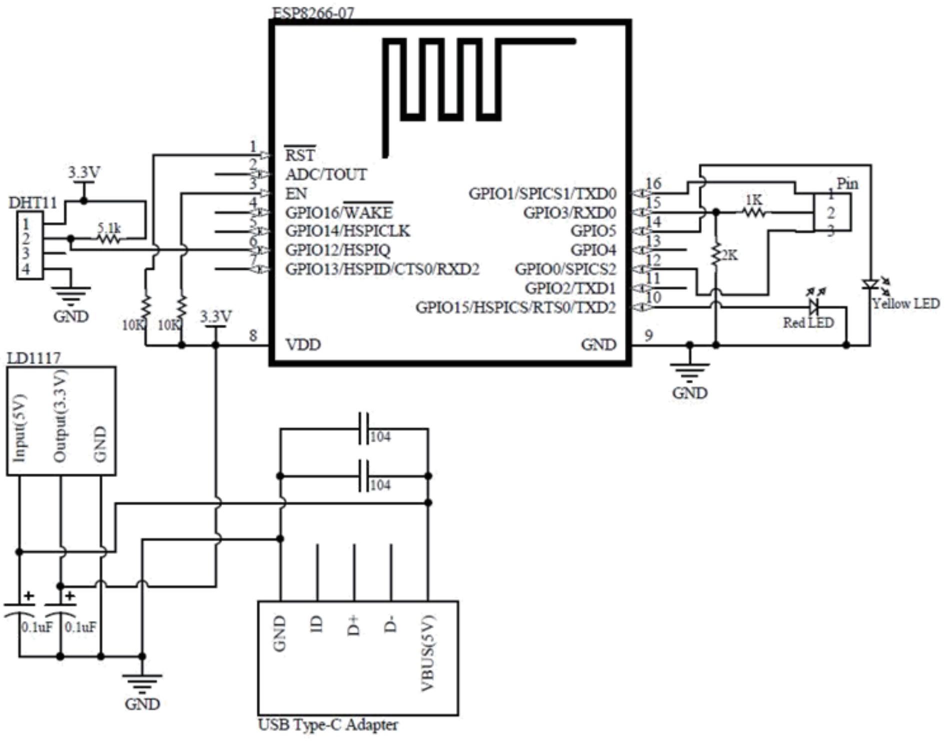


Fig. 3. Esp8266 with peripheral circuit.

3.2 Linkit Smart 7688 with peripheral circuit

Linkit Smart 7688 uses the USB charger to provide 3.3 V for itself and SI7021 (humidity and temperature sensor). The yellow and red LED lights are used to indicate the connection status between the MQTT broker and the data transmission system. In Fig. 4, the GPIO4 and GPIO5 of Linkit Smart 7688 connect with the SCL and SDA of SI7021 to transmit the data.

4. System Software

The proposed system software contains the graphical GUI, Visual C# programming, Wi-Fi module programming, and server programming. On the basis of the MQTT protocol, the topic published/subscribed for every device is shown in Fig. 5. Assume two clients, i.e., user-Company A and user-Company B, are working at this system, where Esp8266 is installed for user-Company A and Linkit Smart 7688 is installed for user-Company B. The transmission data format used among the devices is the JavaScript Object Notation (JSON) format. The human-readable text made up of attribute–value pairs and array data types is used to transmit data objects. For example, the data "{"id":ID, "action":"Auto", "temp":temperature, "humi":humidy}" and the command "{"id":ID, "action":"Auto"}" use the JSON format. The system software is illustrated as follows.

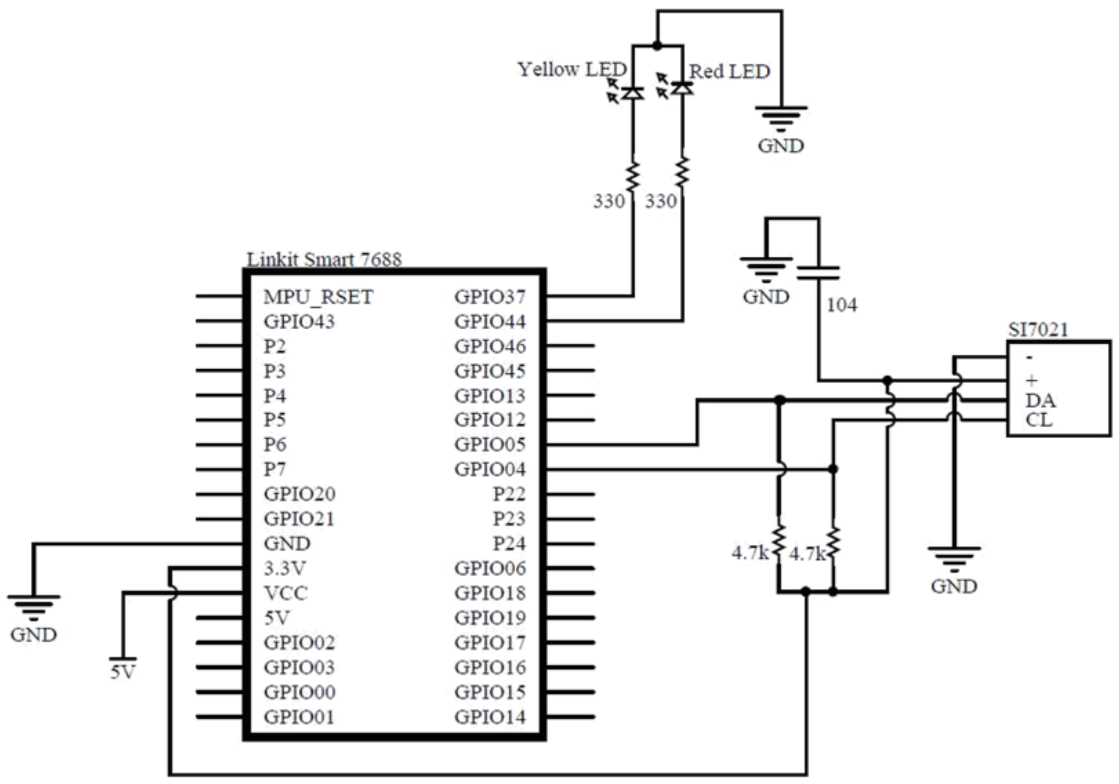


Fig. 4. Linkit Smart 7688 with SI7021.

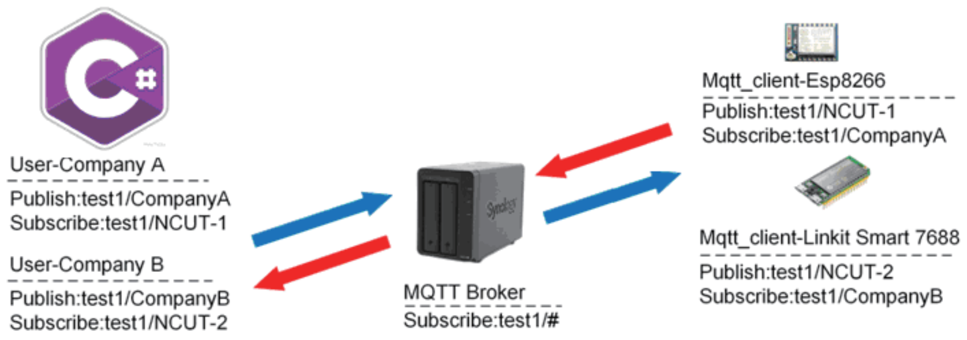


Fig. 5. (Color online) Topic subscribed in every device.

4.1 Wi-Fi module programming

The Wi-Fi module is used to publish the collected data (temperature and humidity) to the MQTT broker (server). The flowchart of Esp8266 programming shown in Fig. 6 is briefly described as follows.

a. After GPIO and variable setting, the Wi-Fi Access Point starts to connect.
b. Check if the Wi-Fi is connected. If yes, go to the next step. Otherwise, keep connecting with the Wi-Fi.

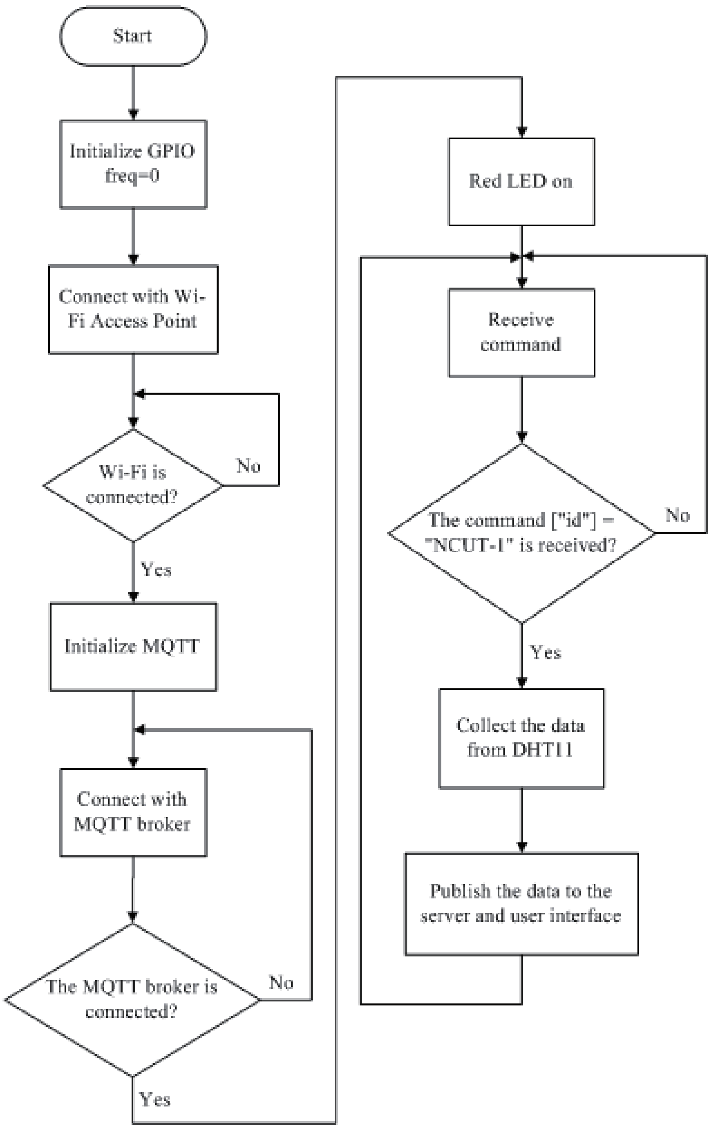

Start
Initialize GPIO freq=0
Connect with Wi-Fi Access Point
Wi-Fi is connected?
No
Yes
Initialize MQTT
Connect with MQTT broker
The MQTT broker is connected?
No
Yes
Red LED on
Receive command
The command ["id"] = "NCUT-1" is received?
No
Yes
Collect the data from DHT11
Publish the data to the server and user interface

Fig. 6. Flowchart of Esp8266 programming.

c. Initialize the MQTT.
d. Connect with the MQTT broker and subscribe a topic.
e. Check if the MQTT broker is connected. If yes, turn on the red LED and go to the next step. Otherwise, go back to step (d).
f. Receive the command.
g. Check if the command ["id"] = "NCUT-1" is received. If yes, go to the next step. Otherwise, go back to step (d).
h. Collect the data from the DHT11 sensor.
i. Publish the data to the server and user interface, and then turn on the yellow LED with a delay period of (fre*1000) ms. Go back to step (d).

4.2 Linkit Smart 7688 programming

The flowchart of Linkit Smart 7688 programming shown in Fig. 7 is briefly described as follows.

a. After the GPIO and I^{2}C and variable setting, the Wi-Fi Access Point starts to connect.
b. Check if the Wi-Fi is connected. If yes, go to the next step. Otherwise, keep connecting with the Wi-Fi.
c. Initialize the MQTT.
d. Connect with the MQTT broker and subscribe a topic.
e. Check if the MQTT broker is connected. If yes, turn on the red LED and go to the next step. Otherwise, go back to step (d).

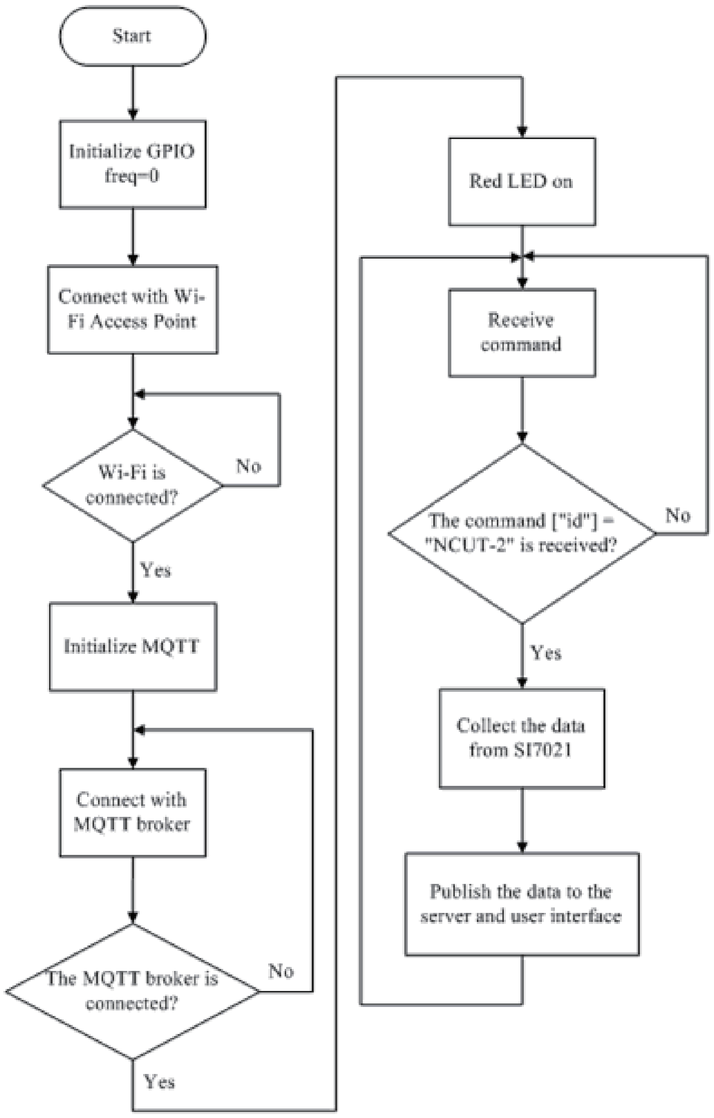

Start
Initialize GPIO freq=0
Connect with Wi-Fi Access Point
Wi-Fi is connected?
No
Yes
Initialize MQTT
Connect with MQTT broker
The MQTT broker is connected?
No
Yes
Red LED on
Receive command
The command ["id"] = "NCUT-2" is received?
No
Yes
Collect the data from SI7021
Publish the data to the server and user interface

Fig. 7. Flowchart of Linkit Smart 7688.

f. Receive the command.
g. Check if the command ["id"] = "NCUT-2" is received. If yes, go to the next step. Otherwise, go back to step (d).
h. Collect the data from the SI7021 sensor.
i. Publish the data to the server and user interface, and then turn on the yellow LED with a delay period of (fre*1000) ms. Go back to step (d).

4.3 Server programming

Server programming uses the Python software package for data communication. The flowchart shown in Fig. 8 is briefly described as follows.

a. Connect with the MySQL server.
b. Connect with the MQTT Broker.
c. Subscribe every topic.
d. Start to receive the data from every client.
e. Check if the data is received. If yes, go to the next step. Otherwise, go back to step (c).
f. Send the data by data ["id"] to the designated database. Next, go back to step (c).

4.4 Visual C# programming

Visual C# programming is used to transmit control signals and monitor the collected data. The main procedure is briefly described as follows, and its flowchart is shown in Fig. 9.

a. Log in with an authorized account.
b. Check if the account is correct. If yes, go to the next step. Otherwise, go back to step (a).

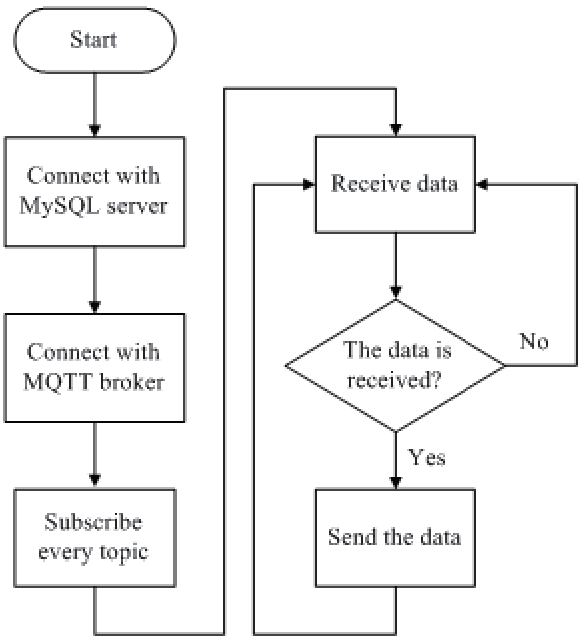


Fig. 8. Flowchart of server programming.

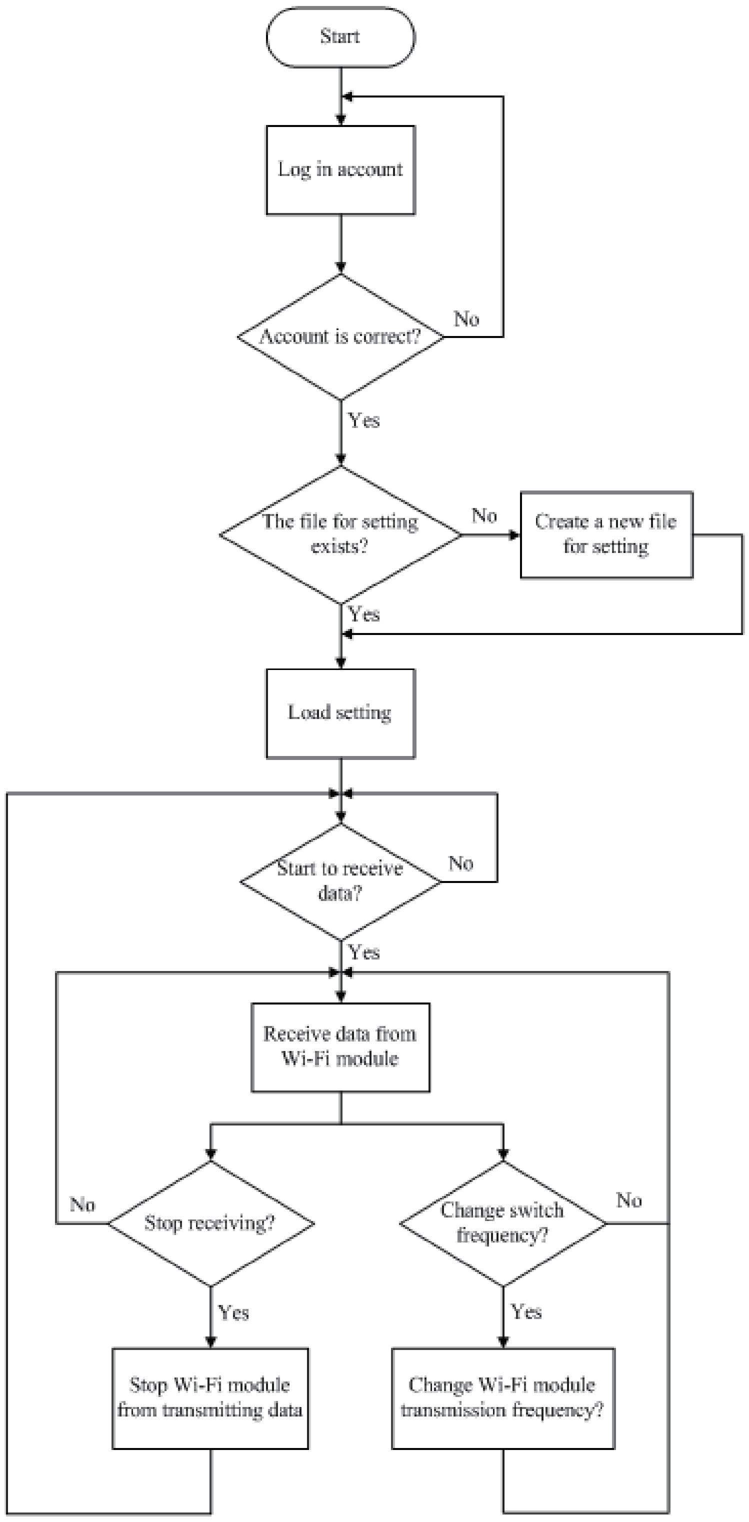

Start
Log in account
Account is correct?
No
Yes
The file for setting exists?
No
Create a new file for setting
Yes
Load setting
Start to receive data?
No
Yes
Receive data from Wi-Fi module
No
Stop receiving?
Change switch frequency?
No
Yes
Yes
Stop Wi-Fi module from transmitting data
Change Wi-Fi module transmission frequency?

Fig. 9. Flowchart of Visual C# programming.

c. Check if the file for setting exists. If yes, load the setting data. Otherwise, create a new file and load the setting.
d. Check if the “Start to receive” button is pressed. If yes, go to the next step. Otherwise, continue the same procedure until the “Start to receive” button is pressed.
e. Receive the data from Wi-Fi modules. Three parallel operation branches are then working independently as follows.

Branch (1):

i. Check if the “Stop receiving” button is pressed. If yes, go to the next step. Otherwise, go back to step (e).
ii. Stop the Wi-Fi module from transmitting data. Then, go back to step (d).

Branch (2):

i. Check if the “Switch frequency” button is pressed. If yes, go to the next step. Otherwise, go back to step (e).
ii. Change the Wi-Fi module transmission frequency. Then, go back to step (e).

5. Performance Results

The proposed system supports two clients (Company A and Company B) in this case study. Every client is provided the Wi-Fi module, temperature- and humidity-sensing module, and GUI operation panel.

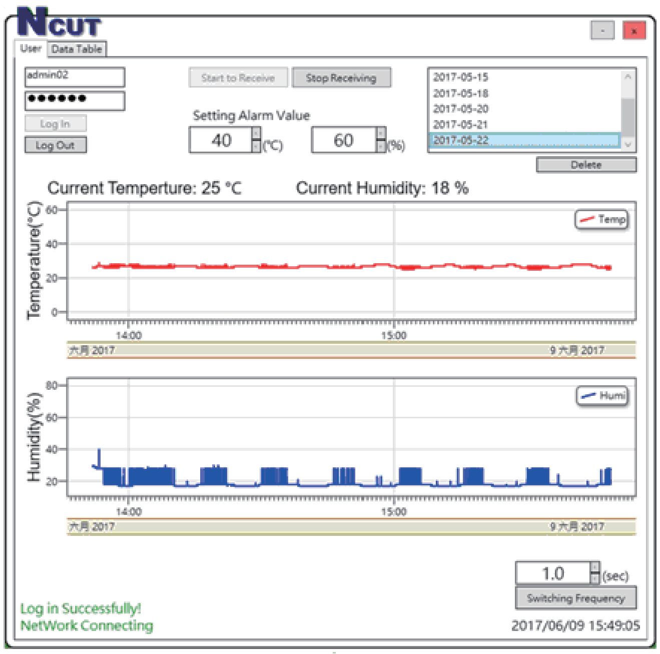
5.1 Connection between user interface and Wi-Fi module

The first step to implement the proposed system is to connect the client interface (Visual C#) with the Wi-Fi module via the MQTT protocol. After logging in the account, the connection will be operated immediately whenever the Wi-Fi module is ready. In Fig. 10(a), the client (Company A) logs in its account and monitors the data received from Esp8266 with DHT11. In Fig. 10(b), the client (Company B) logs in its account and monitors the data received from Linkit Smart 7688 with SI7021. It is known that the sensor SI7021 has a higher accuracy than the sensor DHT11. Therefore, under the same environment in the laboratory room, note that Fig. 10(b) shows a better performance result than Fig. 10(a), especially in terms of humidity. The data tables generated from Esp8266 with DHT11 and Linkit Smart 7688 with SI7021 are shown in Figs. 10(c) and 10(d), respectively.

The historical data at 5/21 is shown in Fig. 11. Note that the air conditioner was turned off at 15:10–15:20; thus, the graph presented a smooth curve.

5.2 Data upload to server

When the server is operated, the data can be collected and sent to the MySQL database immediately. In Fig. 12, every Wi-Fi module (Esp8266, Company A; Linkit Smart 7688, Company B) transmits the data to the server. Next, the server then uploads



admin02

●●●●●●

Setting Alarm Value 40 (°C) 60 (%)

2017-05-15
2017-05-18
2017-05-20
2017-05-21
2017-05-22

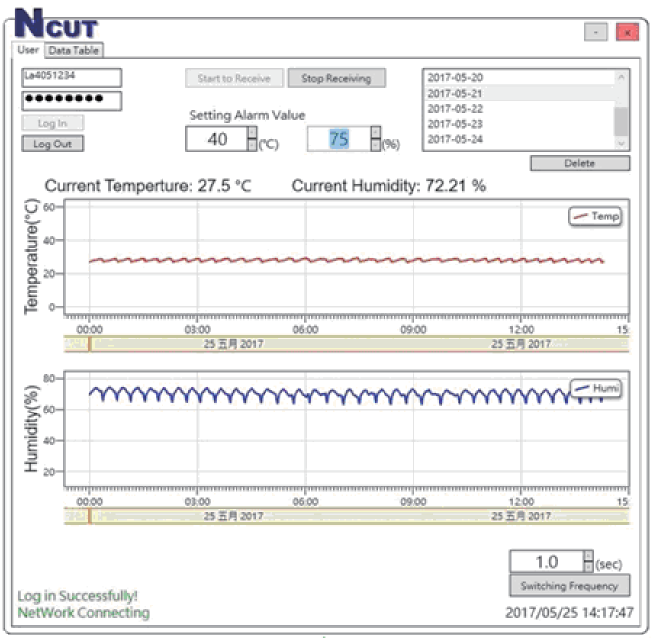
Current Temperture: 25 °C Current Humidity: 18 %

1.0 (sec)

Log in Successfully!
NetWork Connecting

2017/06/09 15:49:05

(a)



La4051234

●●●●●●●●

Setting Alarm Value 40 (°C) 75 (%)

2017-05-20
2017-05-21
2017-05-22
2017-05-23
2017-05-24

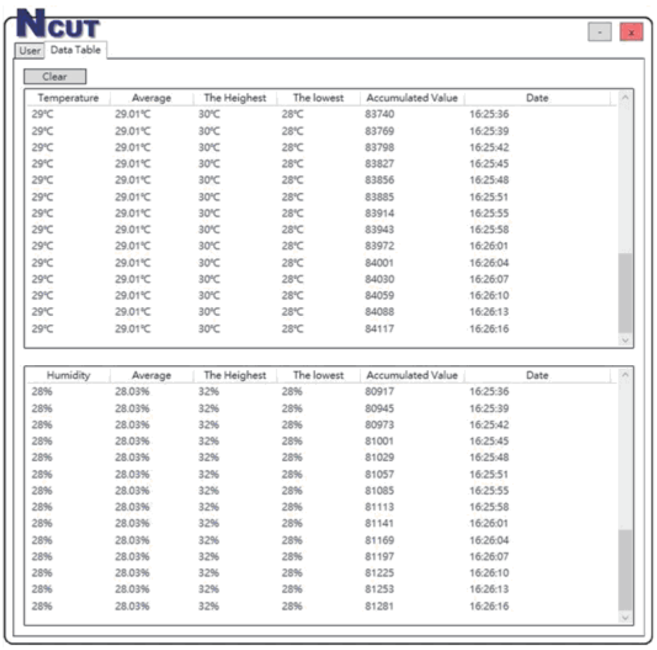
Current Temperture: 27.5 °C Current Humidity: 72.21 %

1.0 (sec)

Log in Successfully!
NetWork Connecting

2017/05/25 14:17:47

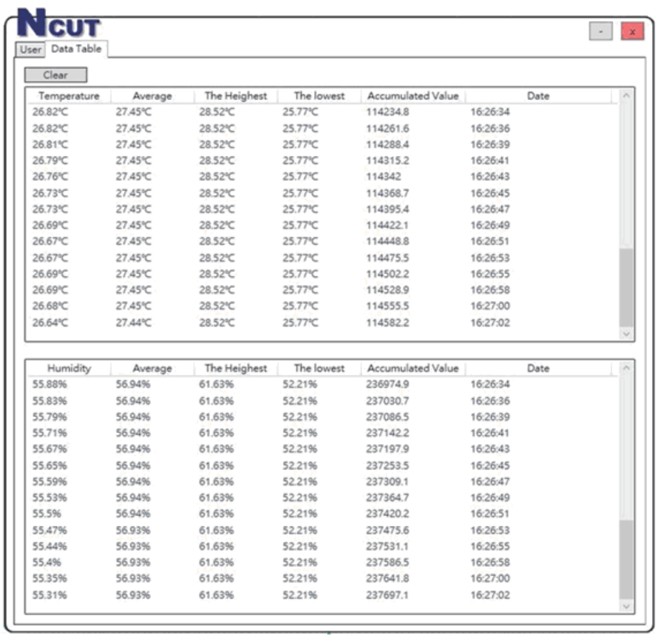
(b)



Temperature	Average	The Heighest	The lowest	Accumulated Value	Date
29°C	29.01°C	30°C	28°C	83740	16:25:36
29°C	29.01°C	30°C	28°C	83769	16:25:39
29°C	29.01°C	30°C	28°C	83798	16:25:42
29°C	29.01°C	30°C	28°C	83827	16:25:45
29°C	29.01°C	30°C	28°C	83856	16:25:48
29°C	29.01°C	30°C	28°C	83885	16:25:51
29°C	29.01°C	30°C	28°C	83914	16:25:55
29°C	29.01°C	30°C	28°C	83943	16:25:58
29°C	29.01°C	30°C	28°C	83972	16:26:01
29°C	29.01°C	30°C	28°C	84001	16:26:04
29°C	29.01°C	30°C	28°C	84030	16:26:07
29°C	29.01°C	30°C	28°C	84059	16:26:10
29°C	29.01°C	30°C	28°C	84088	16:26:13
29°C	29.01°C	30°C	28°C	84117	16:26:16

Humidity	Average	The Heighest	The lowest	Accumulated Value	Date
28%	28.03%	32%	28%	80917	16:25:36
28%	28.03%	32%	28%	80945	16:25:39
28%	28.03%	32%	28%	80973	16:25:42
28%	28.03%	32%	28%	81001	16:25:45
28%	28.03%	32%	28%	81029	16:25:48
28%	28.03%	32%	28%	81057	16:25:51
28%	28.03%	32%	28%	81085	16:25:55
28%	28.03%	32%	28%	81113	16:25:58
28%	28.03%	32%	28%	81141	16:26:01
28%	28.03%	32%	28%	81169	16:26:04
28%	28.03%	32%	28%	81197	16:26:07
28%	28.03%	32%	28%	81225	16:26:10
28%	28.03%	32%	28%	81253	16:26:13
28%	28.03%	32%	28%	81281	16:26:16

(c)

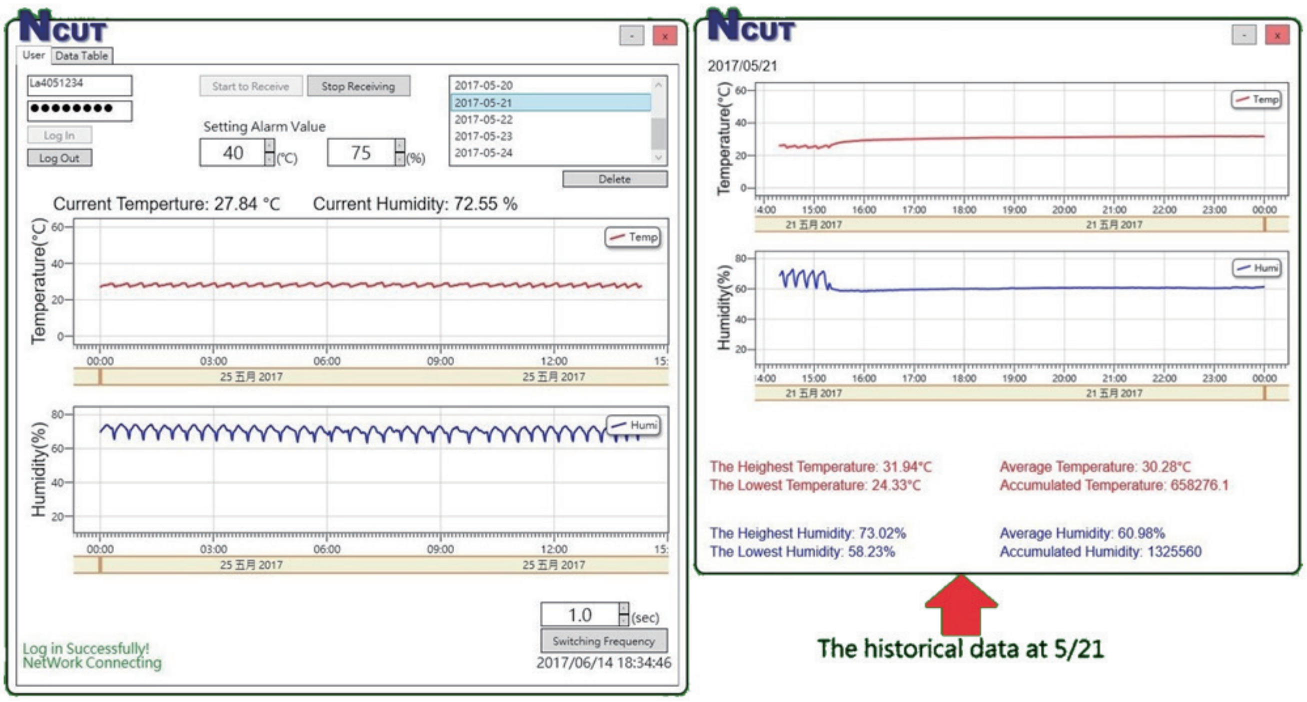


Temperature	Average	The Heighest	The lowest	Accumulated Value	Date
26.82°C	27.45°C	28.52°C	25.77°C	114234.8	16:26:34
26.82°C	27.45°C	28.52°C	25.77°C	114261.6	16:26:36
26.81°C	27.45°C	28.52°C	25.77°C	114288.4	16:26:39
26.79°C	27.45°C	28.52°C	25.77°C	114315.2	16:26:41
26.76°C	27.45°C	28.52°C	25.77°C	114342	16:26:43
26.73°C	27.45°C	28.52°C	25.77°C	114368.7	16:26:45
26.73°C	27.45°C	28.52°C	25.77°C	114395.4	16:26:47
26.69°C	27.45°C	28.52°C	25.77°C	114422.1	16:26:49
26.67°C	27.45°C	28.52°C	25.77°C	114448.8	16:26:51
26.67°C	27.45°C	28.52°C	25.77°C	114475.5	16:26:53
26.69°C	27.45°C	28.52°C	25.77°C	114502.2	16:26:55
26.69°C	27.45°C	28.52°C	25.77°C	114528.9	16:26:58
26.68°C	27.45°C	28.52°C	25.77°C	114555.5	16:27:00
26.64°C	27.44°C	28.52°C	25.77°C	114582.2	16:27:02

Humidity	Average	The Heighest	The lowest	Accumulated Value	Date
55.88%	56.94%	61.63%	52.21%	236974.9	16:26:34
55.83%	56.94%	61.63%	52.21%	237030.7	16:26:36
55.79%	56.94%	61.63%	52.21%	237086.5	16:26:39
55.71%	56.94%	61.63%	52.21%	237142.2	16:26:41
55.67%	56.94%	61.63%	52.21%	237197.9	16:26:43
55.65%	56.94%	61.63%	52.21%	237253.5	16:26:45
55.59%	56.94%	61.63%	52.21%	237309.1	16:26:47
55.53%	56.94%	61.63%	52.21%	237364.7	16:26:49
55.5%	56.94%	61.63%	52.21%	237420.2	16:26:51
55.47%	56.93%	61.63%	52.21%	237475.6	16:26:53
55.44%	56.93%	61.63%	52.21%	237531.1	16:26:55
55.4%	56.93%	61.63%	52.21%	237586.5	16:26:58
55.35%	56.93%	61.63%	52.21%	237641.8	16:27:00
55.31%	56.93%	61.63%	52.21%	237697.1	16:27:02

(d)

Fig. 10. (Color online) User interface and Wi-Fi modules. (a) Client interface from Esp8266 with DHT11. (b) Client interface from Linkit Smart 7688 with SI7021. (c) Data table from Esp8266 with DHT11. (d) Data table from Linkit Smart 7688 with SI7021.



La4051234

Setting Alarm Value 40 (°C) 75 (%)

2017-05-20
2017-05-21
2017-05-22
2017-05-23
2017-05-24

Current Temperture: 27.84 °C Current Humidity: 72.55 %

1.0 (sec)

Log in Successfully!
NetWork Connecting

2017/06/14 18:34:46

2017/05/21

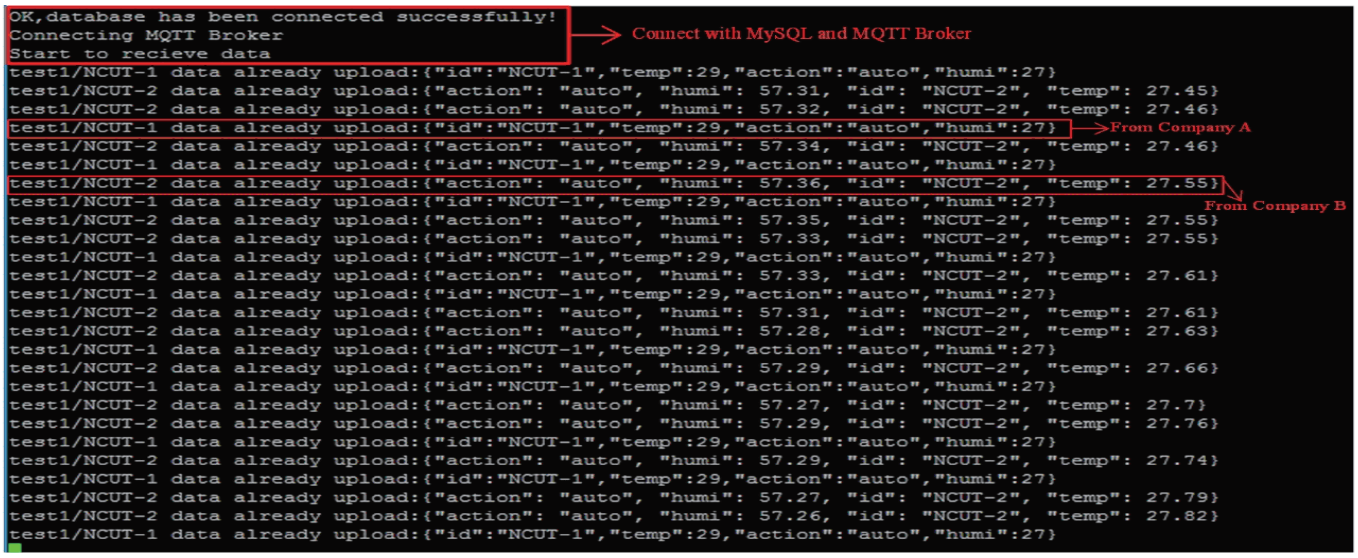
The Heighest Temperature: 31.94°C
The Lowest Temperature: 24.33°C

Average Temperature: 30.28°C
Accumulated Temperature: 658276.1

The Heighest Humidity: 73.02%
The Lowest Humidity: 58.23%

Average Humidity: 60.98%
Accumulated Humidity: 1325560

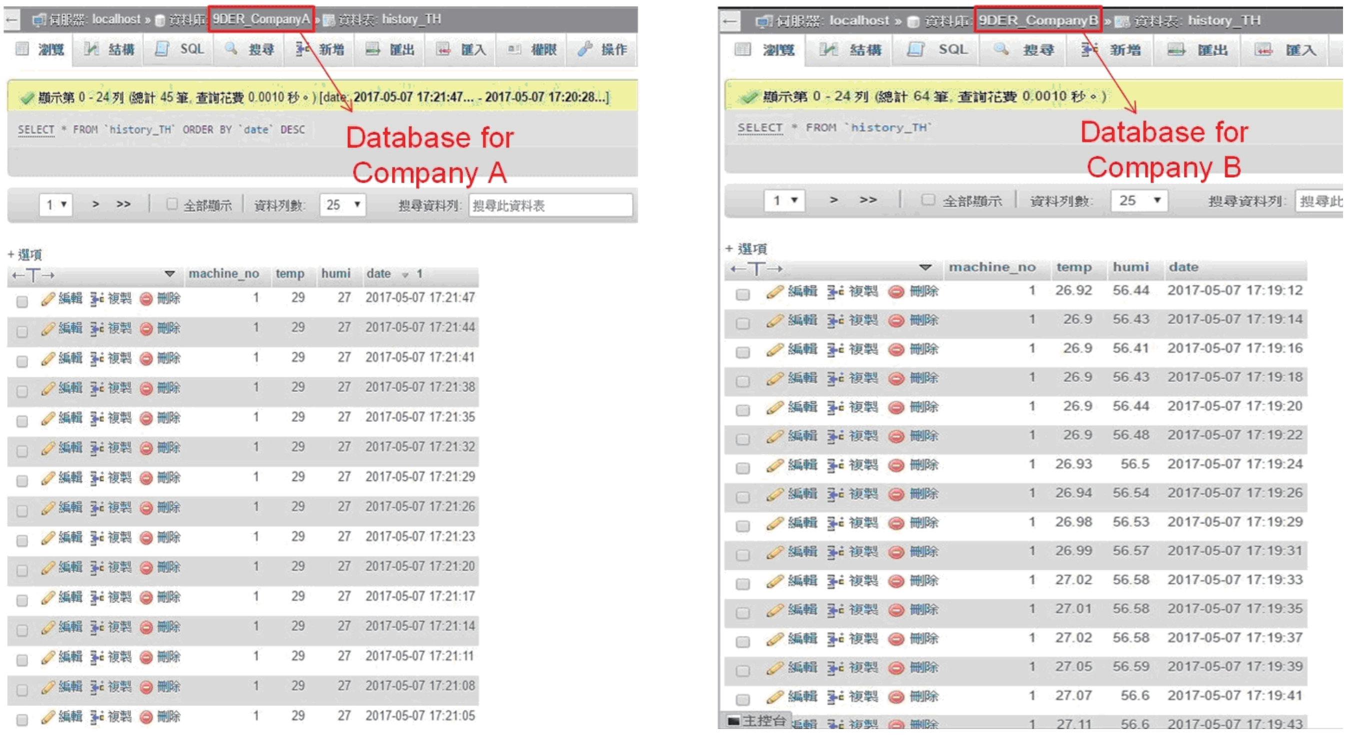
Fig. 11. (Color online) Historical data.



```
OK,database has been connected successfully!
Connecting MQTT Broker
Start to recieve data
test1/NCUT-1 data already upload:{"id":"NCUT-1","temp":29,"action":"auto","humi":27}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.31, "id": "NCUT-2", "temp": 27.45}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.32, "id": "NCUT-2", "temp": 27.46}
test1/NCUT-1 data already upload:{"id":"NCUT-1","temp":29,"action":"auto","humi":27}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.34, "id": "NCUT-2", "temp": 27.46}
test1/NCUT-1 data already upload:{"id":"NCUT-1","temp":29,"action":"auto","humi":27}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.36, "id": "NCUT-2", "temp": 27.55}
test1/NCUT-1 data already upload:{"id":"NCUT-1","temp":29,"action":"auto","humi":27}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.35, "id": "NCUT-2", "temp": 27.55}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.33, "id": "NCUT-2", "temp": 27.55}
test1/NCUT-1 data already upload:{"id":"NCUT-1","temp":29,"action":"auto","humi":27}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.33, "id": "NCUT-2", "temp": 27.61}
test1/NCUT-1 data already upload:{"id":"NCUT-1","temp":29,"action":"auto","humi":27}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.31, "id": "NCUT-2", "temp": 27.61}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.28, "id": "NCUT-2", "temp": 27.63}
test1/NCUT-1 data already upload:{"id":"NCUT-1","temp":29,"action":"auto","humi":27}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.29, "id": "NCUT-2", "temp": 27.66}
test1/NCUT-1 data already upload:{"id":"NCUT-1","temp":29,"action":"auto","humi":27}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.27, "id": "NCUT-2", "temp": 27.7}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.29, "id": "NCUT-2", "temp": 27.76}
test1/NCUT-1 data already upload:{"id":"NCUT-1","temp":29,"action":"auto","humi":27}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.29, "id": "NCUT-2", "temp": 27.74}
test1/NCUT-1 data already upload:{"id":"NCUT-1","temp":29,"action":"auto","humi":27}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.27, "id": "NCUT-2", "temp": 27.79}
test1/NCUT-2 data already upload:{"action": "auto", "humi": 57.26, "id": "NCUT-2", "temp": 27.82}
test1/NCUT-1 data already upload:{"id":"NCUT-1","temp":29,"action":"auto","humi":27}
```

Fig. 12. (Color online) Data transmission.

the data to the MySQL database, as shown in Fig. 13. At a time, all the collected sensing data can be displayed on line from the PHP web, as shown in Fig. 14.



伺服器: localhost » 資料庫: 9DER_CompanyA » 資料表: history_TH

瀏覽 結構 SQL 搜尋 新增 匯出 匯入 權限 操作

顯示第 0 - 24 列 (總計 45 筆, 查詢花費 0.0010 秒。) [date: 2017-05-07 17:21:47... - 2017-05-07 17:20:28...]

```
SELECT * FROM `history_TH` ORDER BY `date` DESC
```

Database for Company A

1 > >> 全部顯示 資料列數: 25 搜尋資料列: 搜尋此資料表

+ 選項

	machine_no	temp	humi	date 1
編輯 複製 刪除	1	29	27	2017-05-07 17:21:47
編輯 複製 刪除	1	29	27	2017-05-07 17:21:44
編輯 複製 刪除	1	29	27	2017-05-07 17:21:41
編輯 複製 刪除	1	29	27	2017-05-07 17:21:38
編輯 複製 刪除	1	29	27	2017-05-07 17:21:35
編輯 複製 刪除	1	29	27	2017-05-07 17:21:32
編輯 複製 刪除	1	29	27	2017-05-07 17:21:29
編輯 複製 刪除	1	29	27	2017-05-07 17:21:26
編輯 複製 刪除	1	29	27	2017-05-07 17:21:23
編輯 複製 刪除	1	29	27	2017-05-07 17:21:20
編輯 複製 刪除	1	29	27	2017-05-07 17:21:17
編輯 複製 刪除	1	29	27	2017-05-07 17:21:14
編輯 複製 刪除	1	29	27	2017-05-07 17:21:11
編輯 複製 刪除	1	29	27	2017-05-07 17:21:08
編輯 複製 刪除	1	29	27	2017-05-07 17:21:05

伺服器: localhost » 資料庫: 9DER_CompanyB » 資料表: history_TH

瀏覽 結構 SQL 搜尋 新增 匯出 匯入

顯示第 0 - 24 列 (總計 64 筆, 查詢花費 0.0010 秒。)

```
SELECT * FROM `history_TH`
```

Database for Company B

1 > >> 全部顯示 資料列數: 25 搜尋資料列: 搜尋此

+ 選項

	machine_no	temp	humi	date
編輯 複製 刪除	1	26.92	56.44	2017-05-07 17:19:12
編輯 複製 刪除	1	26.9	56.43	2017-05-07 17:19:14
編輯 複製 刪除	1	26.9	56.41	2017-05-07 17:19:16
編輯 複製 刪除	1	26.9	56.43	2017-05-07 17:19:18
編輯 複製 刪除	1	26.9	56.44	2017-05-07 17:19:20
編輯 複製 刪除	1	26.9	56.48	2017-05-07 17:19:22
編輯 複製 刪除	1	26.93	56.5	2017-05-07 17:19:24
編輯 複製 刪除	1	26.94	56.54	2017-05-07 17:19:26
編輯 複製 刪除	1	26.98	56.53	2017-05-07 17:19:29
編輯 複製 刪除	1	26.99	56.57	2017-05-07 17:19:31
編輯 複製 刪除	1	27.02	56.58	2017-05-07 17:19:33
編輯 複製 刪除	1	27.01	56.58	2017-05-07 17:19:35
編輯 複製 刪除	1	27.02	56.58	2017-05-07 17:19:37
編輯 複製 刪除	1	27.05	56.59	2017-05-07 17:19:39
編輯 複製 刪除	1	27.07	56.6	2017-05-07 17:19:41
編輯 複製 刪除	1	27.11	56.6	2017-05-07 17:19:43

主控台

Fig. 13. (Color online) Data collection.

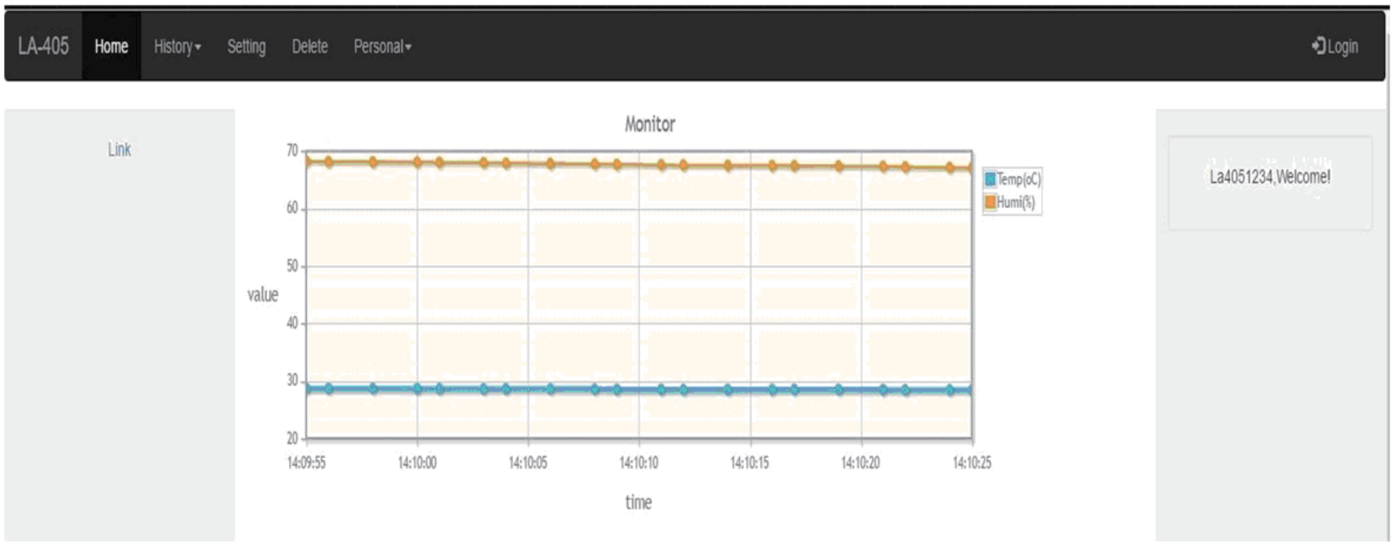


Fig. 14. (Color online) PHP web.

6. Conclusions

In this paper, a real-time remote sensing data collection and monitoring system using the Wi-Fi module-based Visual C# graphical interface in a case study is presented. The illustrative techniques effectively integrate data transmission systems with Wi-Fi modules, a server via the MQTT, MySQL database, and website-based GUI. This means that all clients can be allowed to work independently at the same time, and thus all data can be collected using the same server. Additionally, the website can provide a real-time data monitoring and history tracking capability. In reality, it presents a sufficient background required for Industry 4.0 applications in various industrial disciplines.

Declaration

The authors declare that there is no conflict of interest regarding the publication of this paper.

References

1 J. Lee, B. Bagheri, and H. A. Kao: Manuf. Lett. **3** (2015) 18.
2 F. Shrouf, J. Ordieres, and G. Miragliotta: 2014 IEEE Int. Conf. Industrial Engineering and Engineering Management (IEEM 2014) 697.
3 O. Givehchi, H. Trsek, and J. Jasperneite: 2013 IEEE 18th Conf. Emerging Technologies & Factory Automation (ETFA 2013) 1.
4 A. Fernández, S. del Río, V. López, A. Bawakid, M. J. del Jesus, J. M. Benítez, and F. Herrera: Wiley Interdiscip. Rev.: Data Min. Knowl. Discovery **4** (2014) 380.
5 B. Barbagli, L. Bencini, I. Magrini, G. Manes, and A. Manes: 2011 7th Int. Wireless Communications and Mobile Computing Conf. (IWCMC 2011) 820.
6 X. Li, D. Li, J. Wan, A. V. Vasilakos, C. F. Lai, and S. Wang: Wireless Networks **1** (2015) 23.
7 H. Joh and I. Ryoo: Peer-to-Peer Networking Appl. **8** (2015)567.
8 S. K. Noh, K. S. Kim, and Y. K. Ji: 2013 Int. J. Distributed Sensor Networks **9** (2013) 1.
9 R. Anand and H. Raj: Int. Conf. Internet of Things and Applications (IOTA 2016) 281.
10 G. Dai and Y. Wang: Adv. Comput. Sci. Inf. Eng. (2012) 1.
11 K. Y. Lian, S. J. Hsiao, and W. T. Sung: J. Network Comp. Appl. **36** (2013) 756.
12 A. Di Nisio, T. Di Noia, C. G. C. Carducci, and M. Spadavecchia: 2015 IEEE Int. Workshop on Measurements & Networking (M&N 2015) 1.
13 D. Camps-Mur, A. Garcia-Saavedra, and P. Serrano: IEEE Wireless Commun. **20** (2013) 96.
14 L. Li, X. Hu, and W. Zhang: 4th IEEE Conf. Industrial Electronics and Applications (2009) 403.
15 D. Thangavel, X. Ma, A. Valera, H. X. Tan, and C. K. Y. Tan: IEEE 9th Int. Conf. Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP 2014) 1.
16 U. Hunkeler, H. L. Truong, and A. Stanford-Clark: 2008 3rd Int. Conf. Communication Systems Software and Middleware and Workshops (2008) 791.
17 K. Tang, Y. Wang, H. Liu, Y. Sheng, X. Wang, and Z. Wei: Int. Conf. Information Science and Computer Applications (ISCA 2013) 116.
18 Y. F. Gomes, D. F. Santos, H. O. Almeida, and A. Perkusich: 2015 IEEE Int. Conf. Consumer Electronics (ICCE 2015) 200.
19 J. Bishop and N. Horspool: ACM SIGCSE Bull. **36** (2004) 373.
20 J. Bishop, R. N. Horspool, and B. Worrall: 2002 joint ACM-ISCOPE Conf. Java Grande (2002) 225.
21 M. Monteiro, P. Oliveira, and R. Gonçalves: 10th Int. Conf. Enterprise Information Systems (2008) 449.
22 J. W. Humphries, M. C. Carlisle, and T. A. Wilson: 2003 Annu. ACM SIGAda Int. Conf. (SIGAda 2003) 1.
23 H. L. Cao, G. Van de Perre, R. Simut, C. Pop, A. Peca, D. Lefeber, and B. Vanderborght: 2014 RO-MAN: The 23rd IEEE Int. Symp. Robot and Human Interactive Communication (2014) 555.
24 T. C. Lu, S. H. Hsu, S. J. Tzeng, C. M. Chang, and L. D. Van: 2014 IEEE Int. Conf. Consumer Electronics-Taiwan (ICCE-TW 2014) 179.