Environmental Monitoring System Using IoT and Cloud Service at Real-Time

Mukesh Ranjan Sahay
Department of Software Engineering
San Jose State University
San Jose, United States of America
mukesh.sahay88@gmail.com

Muthu Kumar Sukumaran
Department of Software Engineering
San Jose State University
San Jose, United States of America
muthu220515@gmail.com

Sudha Amarnath
Department of Software Engineering
San Jose State University
San Jose, United States of America
sudha04.a@gmail.com

Thirumalai Nambi Doss Palani
Department of Software Engineering
San Jose State University
San Jose, United States of America
ptndoss@gmail.com

***Abstract*— In this paper, we have proposed an Internet of Things (IoT) based on a real-time environmental monitoring system. Internet of Things (IoT) plays an important role in today's world through a vast and persistent system of sensor networks concerned to the environment and its parameters. This system deals with monitoring and controlling important environmental conditions like temperature, humidity and CO level using the sensors and then this data is sent to the cloud. This information can be accessed from anywhere over the internet and then the sensor data is presented as graphical statistics in a mobile application. This paper explains and presents the implementation and outcome of this environmental monitoring system which uses the sensors for temperature, humidity and other environmental parameters of the surrounding area. This data can be used to take remote actions to control the conditions. This can also be used to send notifications. The collected data is pushed to the cloud storage and an Android application accesses the cloud and presents the results to the end users. The system employs an Arduino UNO board, DHT11 sensor, ESP8266 Wi-Fi module, which transmits data to AWS IoT Core cloud services using MQTT. An Android application is created which accesses the cloud data and displays results to the end users.**

Keywords*— *Internet of Things; Cloud; Mobile application; Sensor Network; Environmental Monitoring

I. INTRODUCTION

Internet of Things (IoT) is expected to transform the world by making it possible to monitor and control important environmental phenomena using the devices/sensors capable of capturing, processing and transmitting the data wirelessly to the remote storage like the cloud services which stores, analyzes and exposes this data as a meaningful information. This information can be accessed over internet through various front-end user interfaces such as a web page or a mobile application, depending on the need and the intended goal. Internet plays a critical role in this transformation by making it more efficient and reliable, and ensuring a smooth and swift communication of data from these IoT devices to the cloud and from the cloud to the users. The Internet of Things (IoT) is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction. The "things" are capable of sensing, capturing and sending the data such as temperature, pressure, humidity, noise, pollution, object detection, patient vitals etc. Environmental monitoring is an important IoT application which involves monitoring and controlling the surrounding environment and presenting this data for effective actions such as remotely controlling the heating or cooling devices, sending notifications about the current status and a long term data analysis. This paper presents and explains the implementation details and the outcome of an environmental monitoring system which uses the IoT for temperature, humidity and other environmental parameters of the surrounding area. The sensors are placed at various locations to collect the data in order to predict the behavior of a particular area of interest. The main goal of this paper is to design and implement an efficient monitoring system through which the required parameters are monitored remotely using internet and the data gathered from the sensors are stored in the cloud and to project the estimated trend on a mobile application. In this paper we also present a trend based on the results of collected data with respect to the normal or specified ranges of particular parameters. The system consists of a central Arduino UNO board which interfaces at the input with temperature and humidity monitoring sensor DHT11 and at the output with ESP8266 Wi-Fi module (NodeMCU) which transmits the sensed data through Internet to a remote cloud storage. Also, when the NodeMCU get all the DHT11 Sensor data it converts this data into a JSON data and send to a web server. This is a low cost system which gives insight into the design and implementation of a complete IoT application involving all aspects from sensing and wireless transmission to cloud storage and data retrieval from cloud via a mobile application. It involves a detailed study and deployment of Arduino development board, its interfacing with input and output modules such as sensors and Wi-Fi module, the usage of API for sending data to the cloud and development of a mobile application based on the Android operating system. The results of the project show the real-time monitoring of temperature and humidity levels from any location in the world and its statistical analysis. This system can be extended to enable remote controlling of various appliances based on the sensed data.

The proposed system that we are proposing uses the Broadcast based approach wherein multiple devices subscribe for a MQTT topic (T) with the Broker. Temperature and humidity values are received from DHT11 sensor and processed by the ESP8266 NodeMCU. Sensor readings from NodeMCU are published to the AWS IOT core that maintains the Shadow table for this Topic (T).

II. RELATED WORK

IoT has emerged as an area of great interest for both the investors and the tech giants, resulting in an overabundance of the research activities and initiatives. Many of the

applications and IoT based concepts have harvested attention including the smart grid, smart city, smart wearable devices and smart home. Mostly all these IoT applications involve sensors and transducers attached to a microcontroller along with a wired/wireless transmission of the data to either a local datastore or a remote cloud service which transforms the raw data into an advantageous information that can be effectively utilized to bring insights in that area. The research and development activities encompass techniques of fabrication of the smart devices, appropriate wireless technologies, development boards, designing network protocols, applications and much more. While working on our project, we explored recent works accomplished in the development of beneficial and exciting applications using low cost development boards such as Raspberry Pi and Arduino. Some of the common applications developed based on these boards include home automation system, patient monitoring systems and weather and environmental monitoring systems. In [1], the data related to temperature, humidity, light intensity, gas leakage, sea level and rain intensity are captured, and then the data is sent wirelessly to ThingSpeak using Arduino UNO. This work is focused significantly on the MATLAB visualization and analysis of the environmental data. In [2], the authors are monitoring the environmental conditions like temperature, relative humidity, light intensity and CO2 level using sensors and LPC2148 microcontroller. The data was sent to ThingSpeak cloud. In contrast with LPC2148, Arduino UNO used in our system is simple, low cost and less complex for such a simple application. In [3], the authors present an IoT based real-time weather monitoring system using Raspberry Pi which is intricate compared to Arduino due to the programming language used and the Raspbian operating system. An Arduino based weather monitoring system is developed and presented in [4]. In this, the data from multiple sensors is imported to the Microsoft Excel which is cumbersome when compared to ThingSpeak or other cloud services. In [5], the authors have designed and developed a wireless network of sensors for environmental monitoring using Raspberry Pi and Arduino. They employed Xbee module to instrument the IEEE 802.15.4 standard for data collection from multiple sensor nodes at a base station (Raspberry Pi). This system can be extended to ensemble large scale applications, however in the present form, the system lacks cloud connectivity.

III. HARDWARE DESIGN

The vital component of the proposed system is a microcontroller module (NodeMCU – ESP8266) which acts as the main processing unit for the entire system. It interfaces with the DHT-11 sensors at the input for receiving temperature and humidity data and with the Wi-Fi module at the output to send the received data to the cloud and web server over the Internet. The microcontroller polls the sensor to fetch the data and sends it over the Internet to the cloud for analysis.

A. Microcontroller

The fundamental hardware component of the proposed system is the microcontroller which interfaces with other components of the system. We have used the NodeMCU (ESP-8266) because of its simplicity, robustness and lost cost. Also, it serves our purpose with ease and convenience [6]. Figure 1 shows a picture of NodeMCU microcontroller used in our system. This microcontroller board is based on the ATmega328P. It has 14 digital input/output pins, 6 analog input pins, a USB connection, 16 MHz quartz crystal, a power jack, and a reset button. It can be powered with a battery. It is programmable with the Arduino IDE (Integrated Development Environment) via a type B USB cable.

Figure 1: NodeMCU Microcontroller board

B. Sensors

For the proposed system, we have selected the DHT-11 sensor for measuring the temperature and humidity for environmental monitoring and preferred a single sensor with both sensing capabilities instead of separate sensors for each parameter. The DHT-11 Temperature & Humidity Sensor provides a temperature & humidity sensor compound with a calibrated digital signal output. It uses the high-class digital-signal-acquisition method and temperature & humidity sensing technology. It guarantees a great consistency and outstanding long-term steadiness. DHT-11 sensor comprises of a resistive-type humidity measurement component and an NTC temperature measurement component. It is small sensor with fast response and high quality. It has a comparatively low cost and a strong anti-interference ability, digital signal output, and precise calibration. It can be easily be interfaced with Arduino UNO board using the DHT library and the connecting wires. Figure 2 shows a picture of the DHT-11 sensor which we used in our system. It has temperature range from 0 to 50°C and humidity range from 20 to 90%RH, and the signal transmission range of 20m. We have installed the DHT library and used it to get the required data.

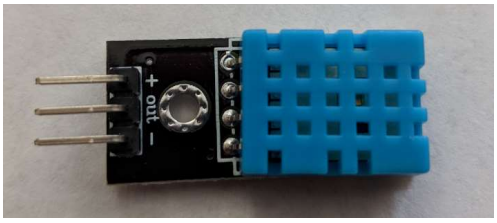

Figure 2: DHT-11 sensor for temperature and humidity

C. Wifi Module

The system uses the WiFi module ESP8266 in order to upload sensor data from DHT-11 to the cloud datastore

(Figure 3). It is a low cost WiFi microchip with full TCP/IP stack and works on the 3.3V that is provided by NodeMCU in our system. This module is constructed through the AT commands and needs the essential sequence to be used as a client. The module can work as both client and server. It acquires an IP address on being connected to WiFi through which the module then communicates over the Internet. After testing our ESP8266 module, we connected it with NodeMCU and then programmed it to configure ESP8266 WiFi module as TCP client and send data to cloud server (AWS IoT) using MQTT, from where we can get the data (received from the sensors) to visualize and analyze on the mobile application.

Figure 3: ESP8266 WiFi Module

D. *Hardware Block Diagram*

Figure 4 shows the hardware block diagram of the proposed system. The figure also depicts the flow of the system functionality with DHT-11 sensors providing the live readings of temperature and humidity simultaneously to the microcontroller which sends these reading through the Wi-Fi module over the Internet to the cloud.

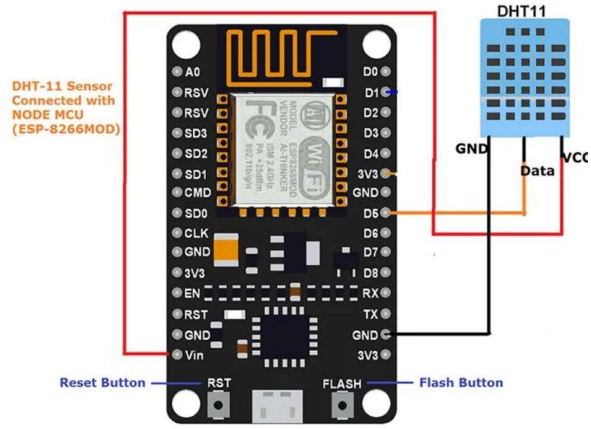

Figure 4: Block diagram and connections

IV. SOFTWARE IMPLEMENTATION

Software as always plays the most important role in the integration and working of any hardware design, so as in our proposed system. There are two parts to our software development: initialization and configuration of hardware, and the development of Android based mobile application as the user interface.

A. *Software for Initialization and Configuration of Hardware*

Arduino IDE was used to program the NodeMCU microcontroller for data retrieval from sensor and data transmission to the cloud. Once the individual hardware components were tested, we integrated them together. Using the Arduino IDE include the libraries from Sketch -> Include Library -> Manage Libraries. Then install the WiFi manager libraries.

Now Install the DHT sensor Library same way we install WiFi manager library but chose “DHT sensor library by Adafruit Version” and select your desired version to install. It is recommended to select latest version of both DHT-11 and WiFi manager Library.

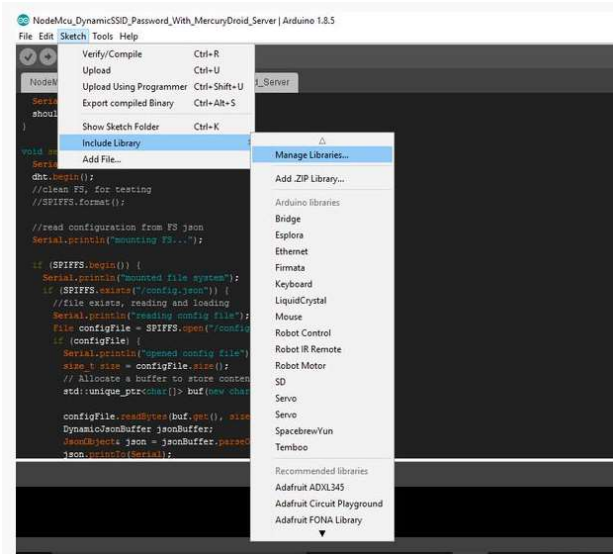

Figure 5: Library installation using the Arduino IDE

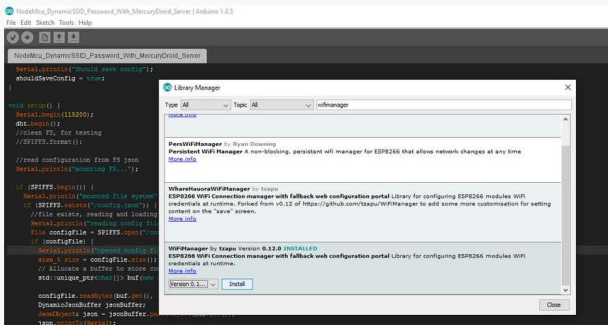

Figure 5.1: WiFi Manager Library installation

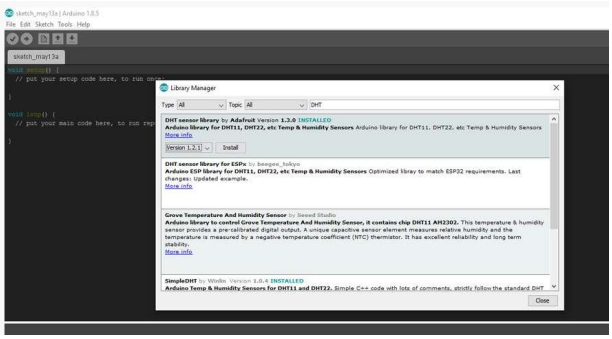

Figure 5.2: DHT Library installation

First of all, the Wi-Fi shield was initialized and then configured the Wi-Fi module ESP8266 as a TCP/IP client. Once the sensor data is read by the NodeMCU microcontroller it is uploaded to the cloud, we use the IoT analytics service of cloud services to aggregate, visualize and analyze live data streams. The Wi-Fi module sends data to the cloud through its assigned IP.

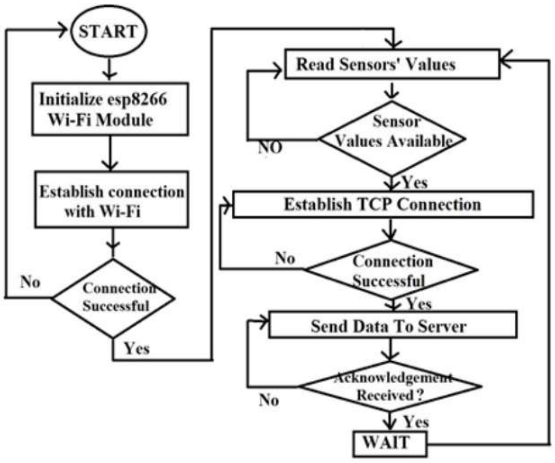

Figure 6: Flowchart of the program

B. Transferring data from the sensor to cloud using MQTT

Message Queuing Telemetry Transport (MQTT) is a widely used Publish-Subscribe based messaging model that uses UDP as a transport protocol. Most of the IoT communications between the devices require a simple, fast, bandwidth efficient, and do not need more overhead for the data transmissions. Because of these reasons, MQTT has an edge over HTTP where in the latter protocol uses TCP, is slower, consumes nearly ten times the bandwidth and is more power consuming. MQTT is client-server-based model where in the server preferably called the MQTT Broker is responsible to handle the requests of one or many clients. The messaging actions between the clients and the MQTT Broker are grouped into Publish and Subscribe models. The Client that intends to send its data usually Publishes it in the JSON format to the IoT MQTT Broker. The Broker maintains the data in its shadow database table and sends this data to the other Clients which are subscribed for the MQTT Topic with the Broker. Since the MQTT protocol is lightweight, the average transmission time for the data transfer from the publishing client to the receiving client that subscribes for this Topic from the MQTT Broker is almost in real time.

C. Android Application

We have created an Android application using the Android Studio Integrated Development Environment (IDE) and Java programming language [8]. The Android application get the data captured by the DHT-11 sensors from the cloud using the needed services. Using the REST API request methods such as GET, POST, PUT, and DELETE, we can create a channel and update its feed, update an existing channel, clear a channel feed, and delete a channel. The application gets the data from the cloud using the REST GET method. The received JSON response from the cloud is then populated using JSON Parser. The end users run the Android application and it allows them to monitor the real-time temperature and humidity readings for the monitored area e.g. a room. We have provided a user friendly and simple user interface for the user to view and interpret the required information from the data. Along with the data we are presenting some helpful outcomes and trends using the sensor data. Apart from the temperature and humidity data from the sensors we have used the live data from the National weather services related to other environmental parameters for deriving some useful facts and presenting it to the user in the Android application.

COMPARING THE TECHNOLOGIES USED

A. AWS IoT vs Azure IoT Hub vs Google Cloud IoT

Amongst the current available Cloud IoT solutions in the market, these three offers the most notable services for IoT Cloud Computing – AWS IoT, Azure IoT Hub and Google Cloud IoT.

Comparison Topics	AWS IoT	Azure IoT Hub	Google Cloud IoT
Market Share (2018)	52% (Leading in the industry)	31%	19%
IoT Cloud Services	IoT Core, Greengrass, IoT Device Manager, IoT – 1 Click, IoT Device Defender, FreeRTOS	IoT Hub, IoT Edge, IoT Central, IoT Solution Accelerators	Cloud IoT Core, Cloud Pub/Sub, Cloud IoT Edge
Analytics	IoT Analysis, Amazon Kinesis, Quicksight, Cloud Watch	Azure Monitor, Azure Storage, Stream Analysis, Power BI, Azure Event Hubs	Big Query, Compute Engine, App Engine, Stackdriver
Supported Protocols	MQTT, HTTPS	MQTT, HTTPS, AMQP	MQTT, HTTPS

SDK	Java, Python, C, Android, iOS, Node.JS	.NET, Java, C, Node.JS, Android, Native iOS	Android, Go, Java, .NET, JavaScript, C, C++, PHP, Python, iOS
Authentication and Security	IAM Service, Cognito User Service, X.509 Client Authentication, TLS for device – cloud encryption	X.509 certificates, Token based (SAS) per device, TLS for device – cloud encryption	X.509 certificates and private keys, IAM users and groups, RSA, Elliptic curve, TLS for device – cloud encryption
Pricing	Per number of messages – each message size 1Kb	Number of messages/ per unit - each message size 4kb	Based on bandwidth usage
Communication	Telemetry, Command based, JSON Web tokens	Telemetry, State based commands usage, JSON Web tokens	Telemetry, Command based, JSON Web tokens

Compared to all others, AWS IoT platform offers a vast IoT Cloud Services and Applications and the projects are generally easy to start with AWS IoT. AWS offers an impressive infrastructure for the IoT applications that can be integrated across a sheer number of AWS Services, Products, Analytics and Powerful AI. Considered in our project, the messaging volume and the bandwidth less than 1Kb per message, AWS is cost effective as well.

B. IoT Stack Protocols: MQTT vs AMQP

For the IoT communication between the embedded devices (e.g. NodeMCU) and the IoT core, there are few protocols available. Of these MQTT and AMQP (Advanced Messaging Queuing Protocol) are widely used in the industry. The choice of the protocols depends on the kind of data and the level of security needed in the communication. MQTT is designed to be used where low overhead and bandwidth is of concern. Whereas AMQP has a very rich set of advanced features and provide more overhead than MQTT. So, we have chosen MQTT over AMQP as a communication protocol between IoT Core and the Clients.

EXPERIMENT RESULTS

A complete design of the proposed environmental monitoring system is shown in Figure 4 and the implemented design is shown in Figure 8, which shows the integration of all hardware components in working condition. DHT-11 and ESP8266 are connected to the NodeMCU microcontroller. DHT-11 and the cloud services are interfaced using the Arduino IDE. The end user Android application connects with cloud and displays the captured data and the relevant information from the analysis of this data.

Here we present the experimental results of the proposed system and display it to the mobile user through the mobile application and the graphical record of temperature and humidity monitoring, and other environmental parameters.

The proposed system use the MQTT protocol for sending the input values from the DHT11 sensor placed in different geo-locations with multiple mobile devices on the receiving edge. These devices subscribe to the same MQTT topic and through NodeMCU the message gets published to that shared topic.

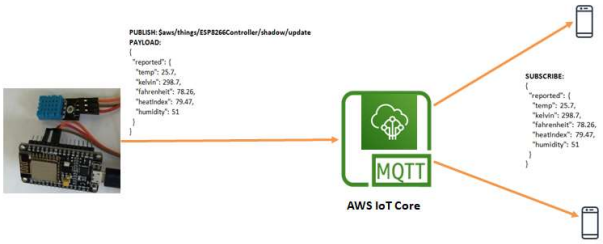

Figure 8: Integrated System

A. Sensor and Cloud

Figure 9 shows the data displayed on the Arduino console.

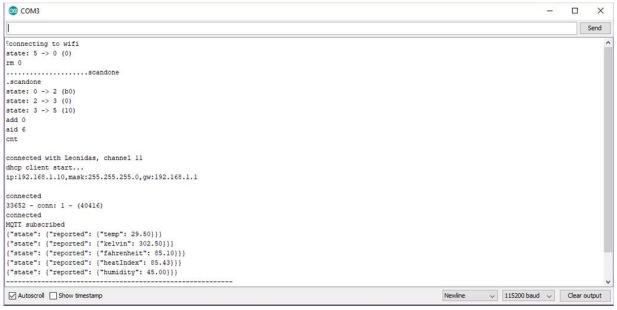

Figure 9: Arduino Console

AWS IoT core Shadow Table for Topic (T) displaying the readings as received from the NodeMCU Client. IOT Core acts as a MQTT Broker in figure 10.

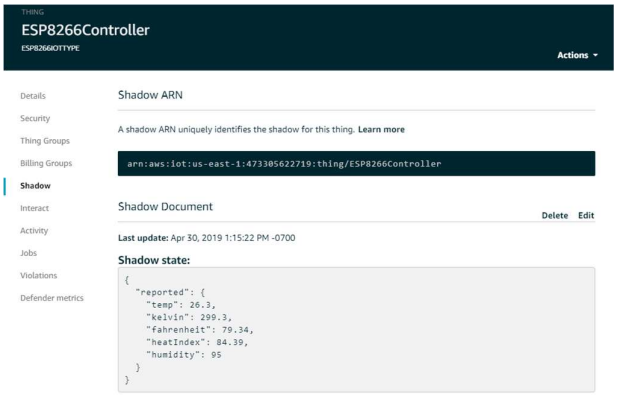

Figure 10: AWS IoT Core Shadow

Figure 11 shows the payload received by the Mobile Client which is subscribed to AWS IOT topic (T) via MQTT.

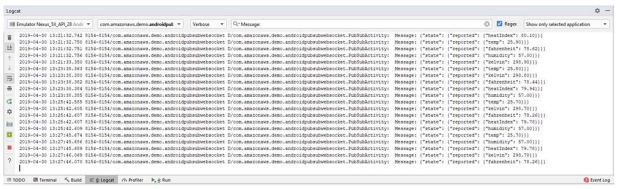

Figure 11: Data received by the mobile client

B. Application User Interface

The Android application shows the data related to the temperature and humidity record captured by the DHT-11 sensors and other environmental parameters fetched from the National Weather Service's APIs.

C. Graphical Record Temperature/Humidity Monitoring

Figure 12 shows the record of temperature monitoring over a period with an interval of 15 seconds. To accurately test the proposed system, we varied the temperature around the sensor artificially by a lighting system, thus a spike can be observed on the graph after which the temperature readings settle to average environmental temperature.

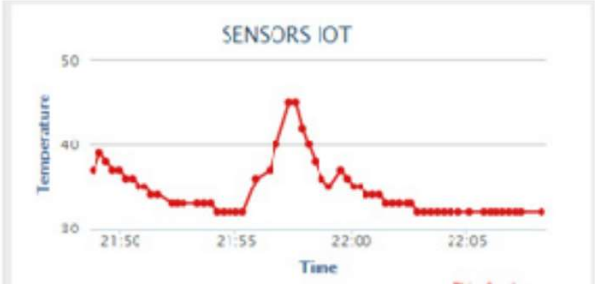

Figure 12: Temperature monitoring record

Figure 13 shows the record of humidity monitoring over a period with an interval of 15 seconds. Like temperature monitoring, we varied the temperature around sensor artificially by a lighting system, the downward spike in humidity can be viewed in the graph after which the humidity readings settle to average environmental humidity.

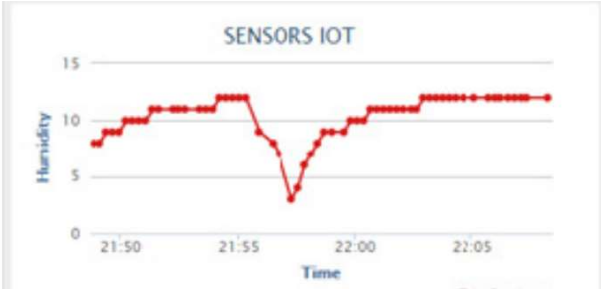

Figure 13: Humidity monitoring record

D. Temprature vs Humidity

Figure 14 shows a graph of humidity versus temperature. As expected, temperature and humidity are inversely related to each other that is the humidity level decreases with the increase in temperature.

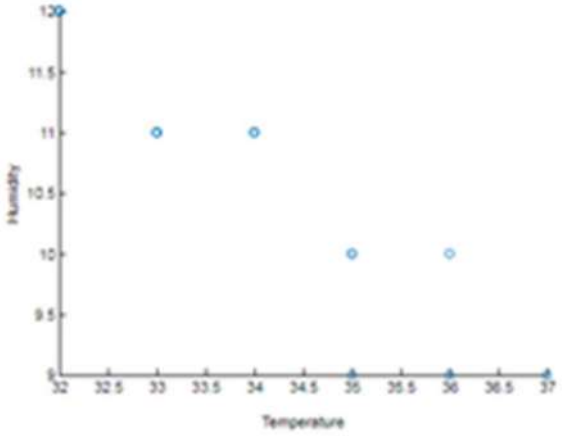

Figure 14: Temperature Versus Humidity

CONCLUSION

This paper presents a real time IoT based environmental monitoring system for monitoring of temperature and humidity of surrounding environment, and to infer some relevant knowledge based on the weather data. The captured data is sent through Wi-Fi to the cloud where both real-time data and its graphical analyses can be viewed. An Android application is developed for the end user to monitor the environment of the area where the hardware is deployed using a smart phone. Also, this system can be used to send the notifications using the one click IoT device about the current condition. This system can be extended to instrument a home automation system by auto triggering some actions and control other devices based on the monitored values of temperature and humidity with the help of the mobile application. The proposed system is a key step in understanding the IoT applications development and implementation. It also serves as a building block for several useful innovations in this direction. The Environmental Sensor Monitoring system furnishes a good paradigm for any Automation System based on Internet of Things (IoT).

ACKNOWLEDGMENT

We thank our professor Chandrasekar Vuppalapati from San Jose State University who provided us this opportunity to work on this research paper and project and guided us on the same. We would also like to show our gratitude to our friends for sharing their pearls of wisdom with us during the course of this research and thank them for reviewing this paper.

REFERENCES

[1] S. Pasha, "ThingSpeak based sensing and monitoring system", International Journal of New Technology and Research, Vol. 2, No. 6, pp. 19-23, 2016

[2] K. S. S. Ram, A. N. P. S. Gupta, "IoT based data logger system for weather monitoring using wireless sensor networks", International Journal of Engineering Trends and Technology, Vol. 32, No. 2, pp. 71-75, 2016

[3] S. D. Shewale, S. N. Gaikwad, "An IoT based real-time weather monitoring system using Raspberry Pi", International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering", Vol. 6, No. 6, pp. 4242-4249, 2017

[4] R. Ayyappadas, A. K. Kavitha, S. M. Praveena, R. M. S. Parvathi, “Design and implementation of weather monitoring system using wireless communication”, Vol. 5, No. 5, pp. 1-7, 2017
[5] S. Ferdoush, X. Li, “Wireless sensor network system design using Raspberry Pi and Arduino for environmental monitoring application”, Procedia Computer Science, Vol. 34, pp. 103-110, 2014
[6] Arduino, Arduino Uno Rev 3 Overview, available at: https://store.arduino.cc/arduino-uno-rev3
[7] Girija, Harshalatha, Shires, Pushpalatha, “Internet of Things (IOT) based Weather Monitoring System”
[8]
[9] MQTT (MQ Telemetry Transport) https://internetofthingsagenda.techtarget.com/definition/MQTT-MQ-Telemetry-Transport
[10] Arduino Development Environment and IDE http://arduino.cc/en/guide/Environment http://arduino.cc/en/main/software
[11] https://developer.android.com