An IoT Based Real-Time Environmental Monitoring System Using Arduino and Cloud Service

Saima Zafar
Department of Electrical Engineering
National University of Computer and Emerging Sciences, FAST-NU, Lahore Campus, Lahore, Pakistan
saima.zafar@nu.edu.pk

Ghosia Miraj
Department of Electrical Engineering
National University of Computer and Emerging Sciences, FAST-NU, Lahore Campus, Lahore, Pakistan
ghosia.miraj25@gmail.com

Rajaa Baloch
Department of Electrical Engineering
National University of Computer and Emerging Sciences FAST-NU, Lahore Campus, Lahore, Pakistan
rajaabaloch@gmail.com

Danish Murtaza
Department of Electrical Engineering
National University of Computer and Emerging Sciences, FAST-NU, Lahore Campus, Lahore, Pakistan
danishmurtaza21@gmail.com

Khadija Arshad
Department of Electrical Engineering
National University of Computer and Emerging Sciences, FAST-NU, Lahore Campus, Lahore, Pakistan
khadija.arshad62@gmail.com

***Abstract*—Internet of Things (IoT) is expected to play a major role in our lives through pervasive systems of sensor networks encompassing our environment. These systems are designed to monitor vital physical phenomena generating data which can be transmitted and saved at cloud from where this information can be accessed through applications and further actions can be taken. This paper presents the implementation and results of an environmental monitoring system which employs sensors for temperature and humidity of the surrounding area. This data can be used to trigger short term actions such as remotely controlling heating or cooling devices or long term statistics. The sensed data is uploaded to cloud storage and an Android application accesses the cloud and presents the results to the end users. The system employs Arduino UNO board, DHT11 sensor, ESP8266 Wi-Fi module, which transmits data to open IoT API service ThingSpeak where it is analyzed and stored. An Android application is developed which accesses the cloud and displays results for end users via REST API Web service. The experimental results show the usefulness of the system.**

Keywords-Internet of Things; cloud; mobile application; sensor network; environmental monitoring

I. INTRODUCTION

Internet of Things (IoT) is expected to revolutionize our world by enabling us to monitor and control vital phenomena in our environment through the use of devices capable of sensing, processing and wirelessly transmitting data to remote storage like cloud which stores, analyzes and presents this data in useful form. From the cloud this information can be accessed through various front end user interfaces such as web or mobile applications, depending upon suitability and requirements. Internet lies at the heart of this transformation playing its role in efficient, reliable and swift communication of data from devices to the cloud and from the cloud to the end users. In this new paradigm, the concept of the typical end system or host in the Internet is modified and hosts comprise of devices or things hence the name Internet of Things. The "things" are capable of sensing and transmitting data such as temperature, pressure, humidity, noise, pollution, object detection, patient vitals etc.

Environmental monitoring is an important IoT application which involves monitoring the surrounding environment and reporting this data for effective short term measures such as remotely controlling the heating or cooling devices and long term data analyses and measures. This paper presents the implementation details and results of an environmental monitoring system. The system comprises 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 which transmits the sensed data through Internet to a remote cloud storage open IoT API ThingSpeak. Through ThingSpeak, MATLAB analytics are carried out on data and trigger is generated. A mobile application is developed based on Android operating system and data is retrieved from ThingSpeak for user display from anywhere in the world. The developed 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 comprehensive study and deployment of Arduino development board, its interfacing with input and output modules such as sensors and Wi-Fi module, the usage of ThingSpeak open source API and finally the 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 appliances based on sensed data.

II. RELATED WORK

Lately, IoT has emerged as an area which has gained immense interest of both venture capitalists and tech giants, resulting in a plethora of research activities and business initiatives. Some of the applications which have garnered attention include smart grid, smart city, smart wearable devices and smart home. Almost all of the various IoT applications involve some kind of sensors and transducers normally attached to a microcontroller along with wired or wireless transmission to either a local database or a remote cloud which transforms raw data into useful information which can be effectively utilized. The research and development activities comprise techniques of fabrication of smart objects or devices, suitable wireless technologies, development boards, designing network protocols, applications and much more. In the context of our project, we explored recent work accomplished in the development of useful and interesting applications using low cost development boards such as Raspberry Pi and Arduino. Some of the popular applications developed using these boards include home automation, patient monitoring systems and weather and environmental monitoring systems. In [1], temperature, humidity, light intensity, gas leakage, sea level and rain intensity are measured and the data are sent wirelessly to ThingSpeak using Arduino UNO. This work focuses considerably on MATLAB visualization and analysis. Authors in [2] monitored and controlled environmental conditions like temperature, relative humidity, light intensity and CO_2 level using sensors and LPC2148 microcontroller. The data was sent to ThingSpeak cloud. In comparison with LPC2148, Arduino UNO used in our system is simple, low cost and less complex for a simple application. Authors in [3] present an IoT based real-time weather monitoring system using Raspberry Pi which is complex compared to Arduino due to Python language and Raspbian operating system. An Arduino based weather monitoring system is developed and presented in [4]. Authors imported data from multiple sensors to excel which is cumbersome when compared to ThingSpeak. Authors in [5] designed and developed a wireless sensor network system for environmental monitoring using Raspberry Pi and Arduino. They employed Xbee module to implement the IEEE 802.15.4 standard for data collection from multiple sensor nodes at a base station (Raspberry Pi). Their system can be extended to suit large scale applications, however in the present form, the system lacks cloud connectivity.

III. HARDWARE DESIGN

The central unit is a microcontroller (Arduino UNO), and acts as the main processing unit for the entire system, it interfaces with the sensor chip at the input for receiving temperature and humidity readings and interfaces with the Wi-Fi module at the output to send the received data to the cloud over the Internet. The microcontroller polls the sensor to retrieve data and sends it over the Internet to ThingSpeak cloud for analysis.

A. Microcontroller

The central hardware component of our system is the microcontroller which interfaces with other components of the system. Since the system comprises of temperature and humidity monitoring for which a single sensor interface is required and no local storage of data therefore we selected Arduino UNO microcontroller which serves our purpose well due to its simplicity, robustness and low cost [6]. Figure 1 shows a picture of Arduino UNO microcontroller used in our system [6]. 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

Fig. 1. Arduino UNO microcontroller board

B. Sensors

We selected temperature and humidity for environmental monitoring and prefered a single sensor with both sensing capabilities instead of separate sensors for each parameter. For this reason, we selected DHT11 composite sensor chip which gives readings for both temperature and humidity at the same time, it has high reliability and excellent long-term stability. It has small dimensions, low cost, good quality, fast response, strong anti-interference ability, digital signal output, and precise calibration. It can be easily interfaced with Arduino UNO board with the help of DHT library and connecting wires. Figure 2 shows a picture of the DHT composite sensor which we used in our system. It has temperature range from 0 to 50°C and humidity range from 20 to 90%RH. It has signal transmission range of 20m. To interface it with Arduino UNO, we connected the Ground and Vcc of the DHT11 sensor with the Ground and 5V of the Arduino. Then we connect the Data pin of the DHT11 to the pin 2 of the Arduino. Then we installed the DHT library and run the code for getting it started.

C. WiFi Module

In order to upload sensor readings from DHT11 to the open source cloud ThingSpeak, Arduino UNO interfaces at the output with WiFi module ESP8266 (Figure 3). It is a low cost WiFi microchip with full TCP/IP stack. It works on the 3.3V that is provided by Arduino UNO in our system. The module is configured through AT commands and needs the

required sequence to be used as a client. The module can work as both client and server. It gets an IP on being connected to WiFi through which the module and then communicates over the Internet. After testing our ESP8266 module, we connected it with Arduino UNO and then programmed Arduino UNO to configure ESP8266 WiFi module as TCP client and send data to ThingSpeak server which is an open IoT platform to visualize and analyze live data from sensors.

Fig. 2. DHT11 composite sensor for temperature and humidity

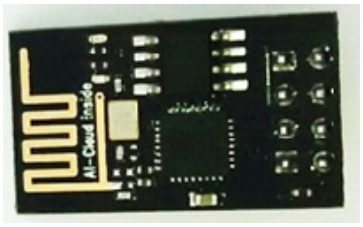

Fig. 3. ESP8266 WiFi module

D. Hardware Block Diagram

Figure 4 shows the hardware block diagram for our system. The figure shows the flow of the system functionality where DHT11 gives live readings of temperature and humidity simultaneously to the microcontroller which sends these reading through the Wi-Fi module over the Internet to the ThingSpeak cloud.

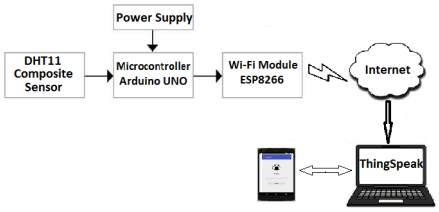

Fig. 4. Hardware block diagram for the environmental monitoring system

IV. SOFTWARE IMPLEMENTATION

Software plays an important role in the integration and working of our hardware design. There are two parts to our software development: initialization and configuration of hardware, and the development of Android based mobile application for user interface.

A. Software for Initialization and Configuration of Hardware

Arduino IDE was used to program the microcontroller for data retrieval from sensor and data transmission to the cloud. After testing the hardware components individually, we integrated them. First of all, the Wi-Fi shield was initialized by sending AT commands in the required pattern to make it work as a client. After the initialization, we configured the Wi-Fi module ESP8266 as a TCP/IP client. Finally we wrote software for reading the sensor data from DHT11 which simultaneously provides both temperature and humidity readings in real-time. Once sensor data is read by the microcontroller and uploaded to the cloud, we use the IoT analytics service of ThingSpeak to aggregate, visualize and analyze live data streams. ThingSpeak provides instant visualizations of the real-time data uploaded by our system [7]. The Wi-Fi module sends data to the cloud through its assigned IP. Once connected to ThingSpeak, an API key is assigned to observe the results over a channel. Therefore we wrote the API key before writing the actual data, and then the data was stored and displayed in the appropriate channel.

B. Android Application

We designed the Android application using the Google App-Inventor Integrated Development Environment (IDE) and Java programming language [8]. The Android application communicates with the microcontroller through ThingSpeak cloud. 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. We sent a Java Script Object Notation (JSON) GET Request to ThingSpeak by using REST API Web Service and channel ID and field number within its parameters. We received the response from ThingSpeak in JSON format and populated the tables in the Android application by 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. The user authentication interface is shown in Figure 5. Our application is efficient, flexible with a user friendly graphic user interface (GUI). The user authentication feature of the application verifies the authorized users through login and password. After authentication, the user views the results in tabular form from the cloud which are shown in the next section.

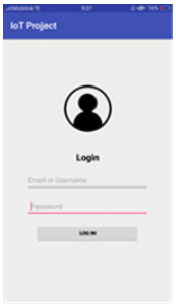

Fig. 5. User authentication interface for Android application

V. EXPERIMENTAL RESULTS

The complete design of our environmental monitoring system is shown in Figure 4. The implemented design is shown in Figure 6 which shows the integration of all

hardware components in working conditions. DHT11 and ESP8266 are connected to Arduino UNO. DHT11 and ThingSpeak are interfaced using the Arduino IDE. The Android application connects with ThingSpeak and displays the sensed data. Table I shows the details of the system hardware, cloud and display.

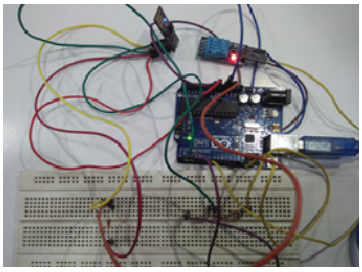

Fig. 6. Hardware Implementation of system design

TABLE I. SYSTEM COMPONENT DETAILS

System Component	Details
Sensor	DHT11
Connectivity	Wi-Fi ESP8266
Microcontroller	Arduino UNO
Cloud	ThingSpeak
User Interface	Android Application

This Section presents the experimental results of our system. We show the sensor data available to the mobile user through the mobile application and the graphical record of temperature and humidity monitoring.

A. Application User Interface

After the authentication phase in which the user enters login and password, the user logs in to the application to view the monitored data. Figure 7 shows the GUI of the Android application which shows the data logging that is a view of the temperature and humidity record.

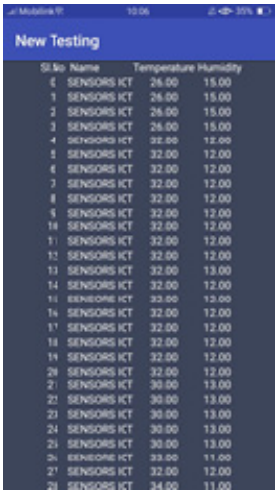

Fig. 7. Temperature and humidity monitoring through the Android application

B. Graphical Record of Temperature Monitoring

Figure 8 shows the record of temperature monitoring over a period of time. The graph is temperature vs. time where the temperature changes are updated after an interval of 15 seconds. In order to accurately test our system, we varied the temperature around the sensor artificially by a lighting system, thus a spike can be viewed in the graph after which the temperature readings settle to average environmental temperature.

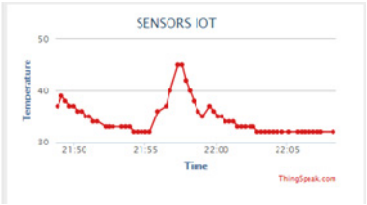

Fig. 8. Record of temperature monitoring

Figure 9 shows the record of humidity monitoring over a period of time. The graph is humidity vs. time where the changes in humidity are updated after an interval of 15 seconds. Similar to 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.

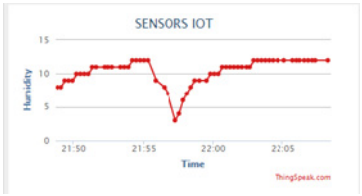

Fig. 9. Record of humidity monitoring

C. Temperature vs Humidity

Figure 10 shows a graph of humidity versus temperature. As expected, temperature and humidity are inversely related to each other. The graph shows that the humidity level decreases with the increase in temperature.

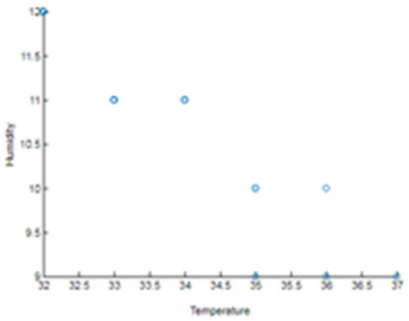

Fig. 10. Humidity versus temperature

VI. CONCLUSION

This paper presents an environmental monitoring system for real-time monitoring of temperature and humidity of surrounding environment. The sensed 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 who can monitor the environment of the area where the hardware is deployed using a smart phone. This system can be extended to implement a home automation system where the monitored values of temperature and humidity can be used to trigger some action and control the devices for heating or cooling via the mobile application. This system is a crucial step in understanding the IoT applications development and implementation and serves as a building block for a number of useful innovations in this direction.

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] https://thingspeak.com

[8] https://developer.android.com