Wireless Sensor Network for Remote Temperature Monitoring in the field of Agriculture

A course project in the domain of wireless sensor networks for the course Internet of Things (I3010)

UNDERTAKEN BY

ANAYA PAWAR

(T. Y. Q 5, G. R. 1710263)

ANEESH PODUVAL

(T. Y. Q 6, G. R. 1710045)

SARTHAK CHUDGAR

(T. Y. Q 16, G. R. 1710202)

JOHNATHAN FERNANDES

(T. Y. Q 37, G. R. 1710168)

UNDER THE GUIDANCE OF **PROF. (DR.) MANISHA RAJESH MHETRE**

TABLE OF CONTENTS

PREFACE

Wireless Sensor Networks are a group of devices used to collect data over a wide area. They can be deployed in a variety of **topologies** and in various **environmental locations** in order to monitor **multiple parameters** and **transmit** that data into a human level interface system. WSNs have found their way into several fields in day to day life, from military to medical to transport to monitoring to agricultural.

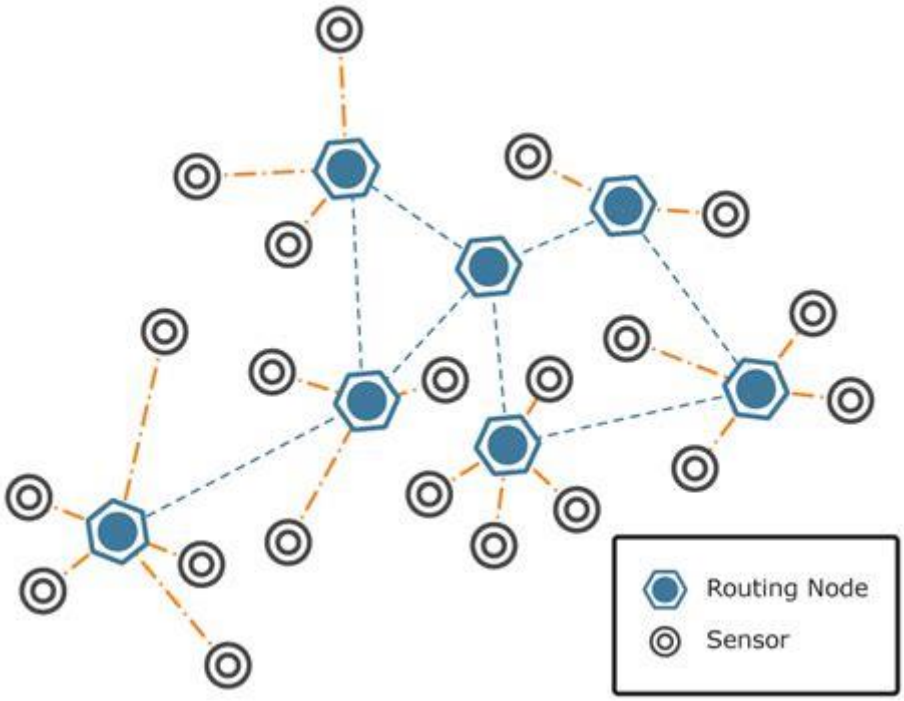

Figure 1: Wireless Sensor Networks

Focusing on that last point, the aim of this course project is to design and create a system that will assist specialists in the **agricultural** field by detecting the **temperature** in an environment and deliver it to a user at a **remote** location. By monitoring the system, agriculturalists will be able to remotely monitor the temperature of different areas of their farms and adjust the parameters of their workspace as required.

CASE STUDY

Soil is for the farmer what the pulse is for the doctor. It helps them take decisions about when to irrigate, when and what to sow, use nutrients and so on.

While some farmers have indigenous knowledge of detecting soil temperature and health, such knowledge is confined to only a few. Taking farming decisions on the basis of soil temperature and health have become even more difficult in the age of climate change.

Figure 2: A SoilSens station set up in a field

Farmers facing farming decision predicaments have started using Soilsens which is a low cost smart soil temperature monitoring system .Soilsens product line is developed by Proximal Soilsens Technologies Pvt.Ltd , incubated at Indian Institute of Technology Bombay(IITB), Mumbai with support from the Ministry of Department of Science and Technology (DST) and Ministry of Electronics and Information Technology (Meity).

The system is embedded with soil temperature sensor, soil moisture sensor, ambient humidity sensor and ambient temperature sensor. Farmers are informed about the field temperature through a mobile app. This data is also available on cloud.

The mobile app is in different Indian languages to suit Indian farmers. As far as the technology is concerned, it can be used in open fields in all seasons and for different crops. The sensor can be installed at any depth as per the crop requirement and can be installed in any crop and in any soil.

Data is monitored and advisory is given .When the temperature crosses the pre-set threshold value, it alarms the farmer. The data can also be used for early prediction of plant diseases.

Powered by a battery, the system is charged by a solar panel and does not require any external power supply. It is a modular system designed keeping in mind the Indian agriculture community. Height of the system is adjustable and it can be varied from 1 m to 3 m based on the height of the crop.

The SMART system with various sensors, software, dash-board, and mobile app and GSM subscription for 3 months costs approximately 25k which can be reduced with volumes.

Temperature monitoring platforms like Pycno, allMETEO, and Smart Element are also some of the vivid examples of how the application of smart sensing technology in agriculture helps deliver effective temperature notifications directly to farmers' laptops and smartphones, enabling them to immediately take action.

Figure 3: Phytech members unveil PlantBeat

Phytech is another such system whose patented sensors attach gently to a growing stem or fruit to directly monitor the plant's temperature.

Data from a group of sensors is collected by a hub out in the field, which sends the encrypted information to Phytech's cloud servers over the cellular network. Farmers can view real-time data and analyze trends through Phytech's mobile app and website. Armed with precise quantification of their plants' wellbeing, they can start to make better-informed decisions and potentially automate plant care through other smart farm systems.

Phytech systems are in use at hundreds of orchards, fields and greenhouses and can be configured to the needs of dozens of crops.

PROJECT TOOLS

As far as IoT definitions go, our model is a **level 3 deployment system**, consisting of one node which records data and transmit it to the cloud. Data processing is done **client side** on the node, while visualization is done **server side** and is available via website or mobile application.

It uses the "**Exclusive Pair**" communications protocol, wherein the node establishes a connection with the server and then exclusively transfers the acquired data to it.

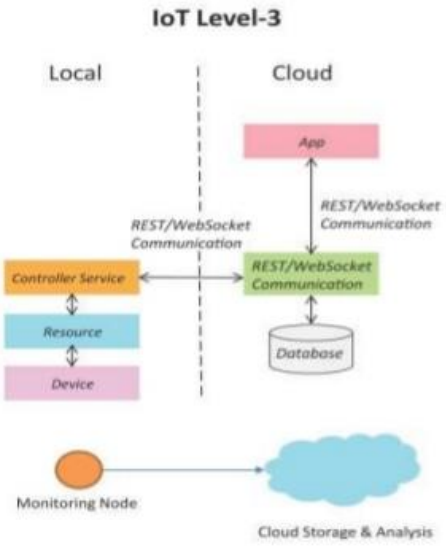

Figure 4: Level-3 IoT system block diagram

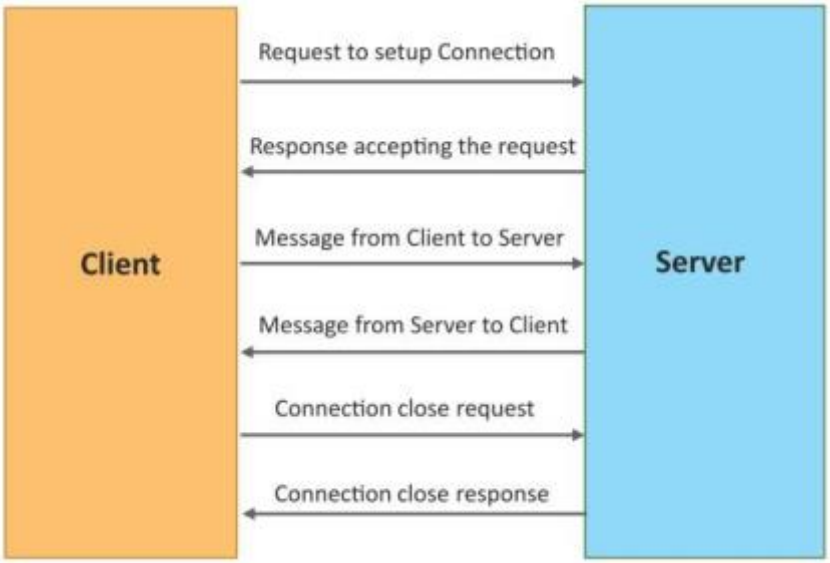

Figure 5: Client-Server communication block diagram

From a hardware point of view, we utilize a **Node Micro Controller Unit** Development Kit (a programmable development board using the Node MCU framework), an **ESP8266 Wi-Fi Module** (Used in conjunction with the NodeMCU to establish an internet connection over a predefined Wi-Fi network) and an **LM35DZ temperature sensor** (a space and cost efficient

temperature sensor). These two, used in conjunction with each other form a single **node** in our Wireless Sensor Network.

As far as software is concerned, **Thingspeak.com** is a website managed by **Mathworks** (whose product **MATLAB** we are all familiar with) geared towards students studying Internet of Things applications. They provide an easy to use interface which allows users to view their data in a multitude of methods. For this project, a free account will suffice.

PROJECT SETUP

We set up the NodeMCU to communicate with the Thingspeak server according to the **circuit diagram** below.

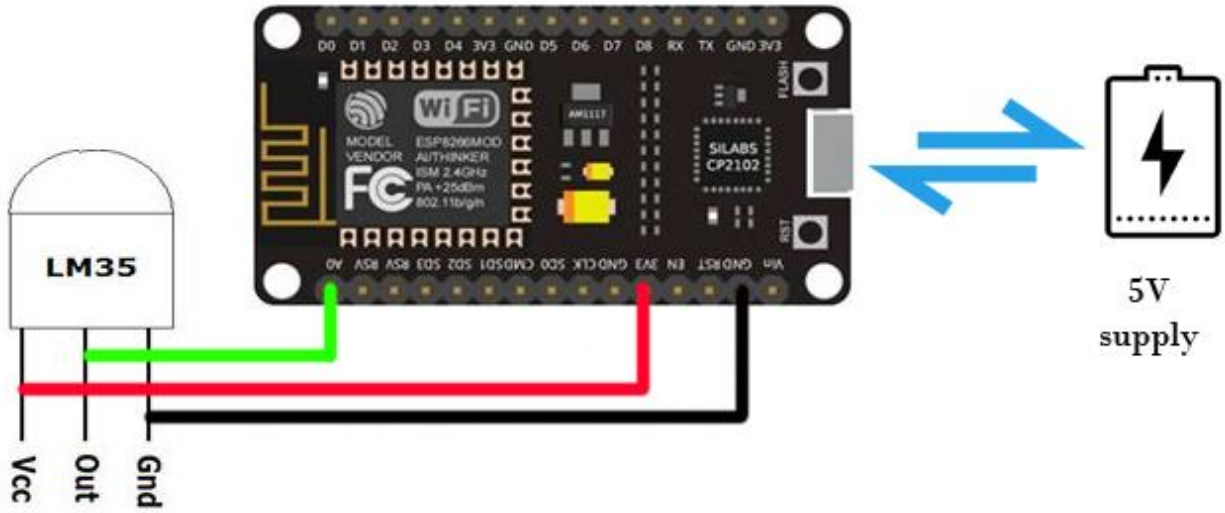

Figure 6: Wiring diagram for connections

While functional, we still have to calibrate and set up the sensor for accurate readings

SENSOR CALIBRATION

Given our background in Instrumentation and Control Engineering, we know that **calibration** of any sensor is the first step to any project involving them.

The LM35DZ has an output voltage of 0 to 1.1 Volts, and it changes **linearly** with temperature. In order to properly calibrate it, we placed the sensor in a temperature controlled environment along with a pre-calibrated temperature sensor and noted down its output at specific temperatures.

Temperature(˚C)	0	10	20	30	40	50	60	70	80	90	100

Output(mV)	0	31	63	94	125	155	185	217	247	279	309

Table 1: Output voltage vs. temperature

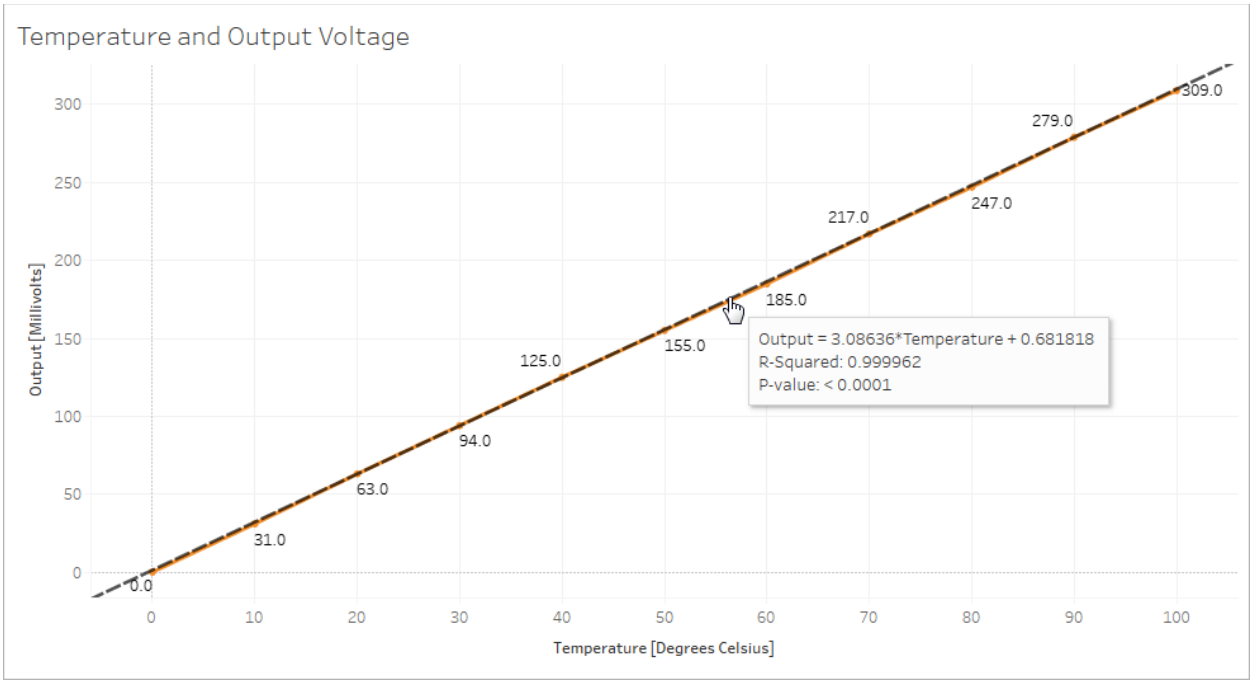

Figure 7: Temperature vs. output voltage graph

Plotting this data allows us to generate an **equation** for temperature in standard linear form as:

$$Temperature = \frac{Output + 0.682}{3.09}$$

Equation 1: Temperature - voltage equation

RESULTS

The recorded temperature can be monitored via **web browser** or in case of a Thingspeak compatible application, by entering the **API read key** "LNC57PVYCQ47G0JT".

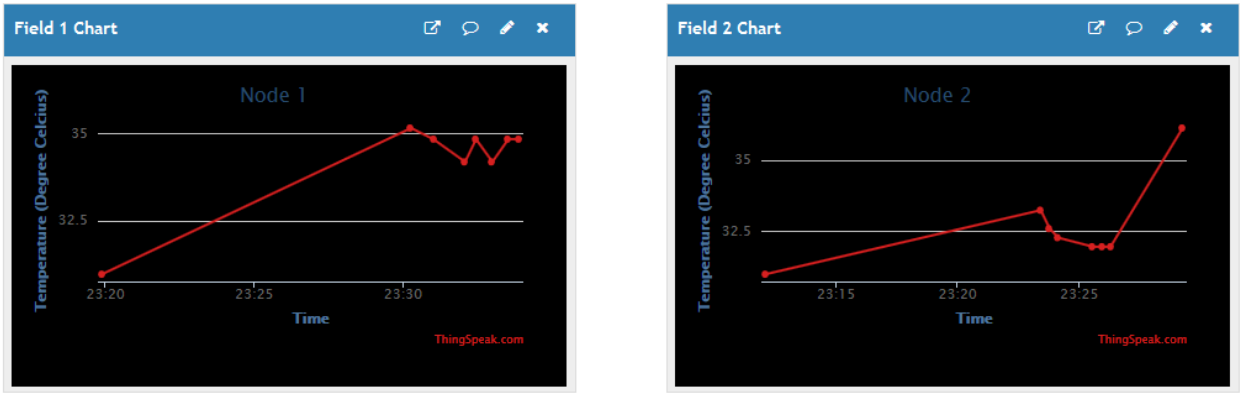

Figure 8: Monitoring graphs

FUTURE SCOPE

Although currently using only two nodes, the **expandable** nature of the Thingspeak server allows us to utilize several more devices (up to **8** with a free account) to create a full scale network if required.

Due to the wide range of compatible sensors available today, we can also use this system to monitor various **other parameters** alongside or in place of temperature.

CONCLUSION

Through this project we both applied previously studied concepts such as **sensor calibration** and newer concepts such as **IoT device infrastructure**, and also learned new concepts which we can apply in future endeavors.

ACKNOWLEDGEMENT

This project would not have been possible without the constant guidance and recommendations from our professor.

BIBLIOGRAPHY

DIvya , Bala - ESP8266: Step by Step Book for ESP8266 IOT, Arduino Nodemcu Dev Kit Kindle Edition

Kranz, Maciej - Building the Internet of Things: Implement New Business Models, Disrupt Competitors, Transform Your Industry

Madisetti, Bahga - Internet of Things (A Hands-on-Approach) 1st Edition

Monk, Simon - Programming Arduino: Getting Started with Sketches, Second Edition (Tab) 2nd Edition

APPENDIX

INDIVUDUAL CONTRIBUTION

- ANAYA PAWAR (T. Y. Q 5, G. R.1710263)
  Project Research, literature survey, case studies.
- ANEESH PODUVAL (T. Y. Q 6, G. R. 1710045)
  Node programming, initial project setup.
- SARTHAK CHUDGAR (T. Y. Q 16, G. R. 1710202)
  Hardware component management, Sensor calibration.
- JOHNATHAN FERNANDES (T. Y. Q 37, G. R. 1710168)
  Final hardware setup, Thingspeak cloud setup, project testing and reporting.

FIGURES

EQUATIONS

TABLES

CODE

```
//IoT Course Project - Wireless Sensor Network for Remote Temperature Monitoring
```

```
#include <ESP8266WiFi.h>//NodeMCU header file
String apiKey = "1M8H86BEC47ZBZFA";//Thingspeak API key
const char* ssid     = "Jon.AP";//WiFi ssid
const char* password = "JOHNATHAN";//WiFi Password
const char* server = "api.thingspeak.com";//Thingspeak server address
float temp_celsius = 0;//Temperature variable
WiFiClient client;//Establish WiFi client
void setup() //setup function, executed once at startup
{
  WiFiServer server(80);//Initialize WiFi server
  Serial.begin(115200);//Start serial comunications
  WiFi.begin(ssid, password);//Authenticate WiFi network
  server.begin();//Start server
}
void loop() //loop function, executed continously
{
  temp_celsius = ((analogRead(A0)+ 0.682)/3.09);//Obtain temperature from output voltage
reading, formula derived graphically
  if (client.connect(server,80))//If devices establishes connection with server
  {
          //Send temperature to server, syntax defined by thingspeak. Field 1 used for node 1
    String postStr = apiKey;
    postStr +="&field1=";
    postStr += String(temp_celsius);
```

```
   postStr += "\r\n\r\n";
   client.print("POST /update HTTP/1.1\n");
   client.print("Host: api.thingspeak.com\n");
   client.print("Connection: close\n");
   client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
   client.print("Content-Type: application/x-www-form-urlencoded\n");
   client.print("Content-Length: ");
   client.print(postStr.length());
   client.print("\n\n");
   client.print(postStr);
   Serial.print("Temperature: ");
   Serial.print(temp_celsius);
   Serial.print(" degrees Celcius");
   Serial.println("%. Send to Thingspeak.");
  }
  client.stop();//Stop client
  delay(10000);//30 sec delay between updates, required by thingspeak
}
```