**IOT BASED TEMPERATURE AND HUMIDITY MEASUREMENT SYSTEM**

**INTRODUCTION**

The temperature and humidity of an environment is of uttermost importance to humans, getting to know the current temperature and humidity of the air in the environment. We would be measuring the temperature and humidity by building a system using a list of components stated below.

**STATEMENT OF PROBLEM**

Humans are faced with the challenge of getting to know the temperature and humidity of the air in the environment, to enable them know when to turn on or turn off an air conditioner, generally to give them control over the temperature of their environment. Although some systems have been created to automatically tell the temperature of the environment, we are leveraging the DHT11 sensor and nodemcu to build a system to help solve the problem stated. In this project we would be using the nodemcu to create a communication between the hardware system developed and the google firebase (a database to hold data sent from the microcontroller), an MIT app would be created to fetch the current data from the microcontroller and display it to the user.

**COMPONENTS TO BE USED**

Below are some components we would use:

* ESP8266 nodemcu board
* DHT-11 sensor
* Buzzer
* Jumper wires and a breadboard
* USB cable for uploading the code
* Transistors
* 5v DC power supply
* Motor (fan)

**ESP8266 nodemcu board:**

The ESP8266 WiFi Module is a self contained SOC with integrated TCP/IP protocol stack that can give any microcontroller access to your WiFi network. It creates a communication between the hardware components and the database. It also houses the microcontroller that processes the software codes sent to the hardware components.

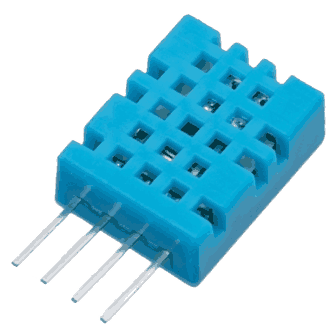


NodeMCU is an open source platform based on ESP8266 which can connect objects and let data transfer using the Wi-Fi protocol. In addition, by providing some of the most important features of microcontrollers such as GPIO, PWM, ADC, and etc, it can solve many of the project's needs alone.

**DHT-11 Sensor:**

DHT11 sensor measures and provides humidity and temperature values serially over a single wire. It can measure the relative humidity in percentage (20 to 90% RH) and temperature in degrees Celsius in the range of 0 to 50°C.

DHT11 is a 4-pin sensor, these pins are VCC, DATA, GND and one pin is not in use shown in fig below.



The DHT11 is a basic, low cost digital temperature and humidity sensor.

**BUZZER:**

An arduino buzzer is also called a piezo buzzer. It is basically a tiny speaker that you can connect directly to an Arduino. You can make it sound a tone at a frequency you set. The buzzer produces sound based on the reverse of the piezoelectric effect.



We simply connect the ground wire to the ground pin on the breadboard, and connect the supply wire to a digital pin on the breadboard connected to the microcontroller.

**JUMPER WIRES AND A BREADBOARD:**

Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each other without soldering. Jumper wires are typically used with breadboards and other prototyping tools in order to make it easy to change a circuit as needed.



Jumper wires typically come in three versions: male-to-male, male-to-female and female-to-female. The difference between each is in the end point of the wire. Male ends have a pin protruding and can plug into things, while female ends do not and are used to plug things into. Male-to-male jumper wires are the most common and what you likely will use most often. When connecting two ports on a breadboard, a male-to-male wire is what you’ll need.



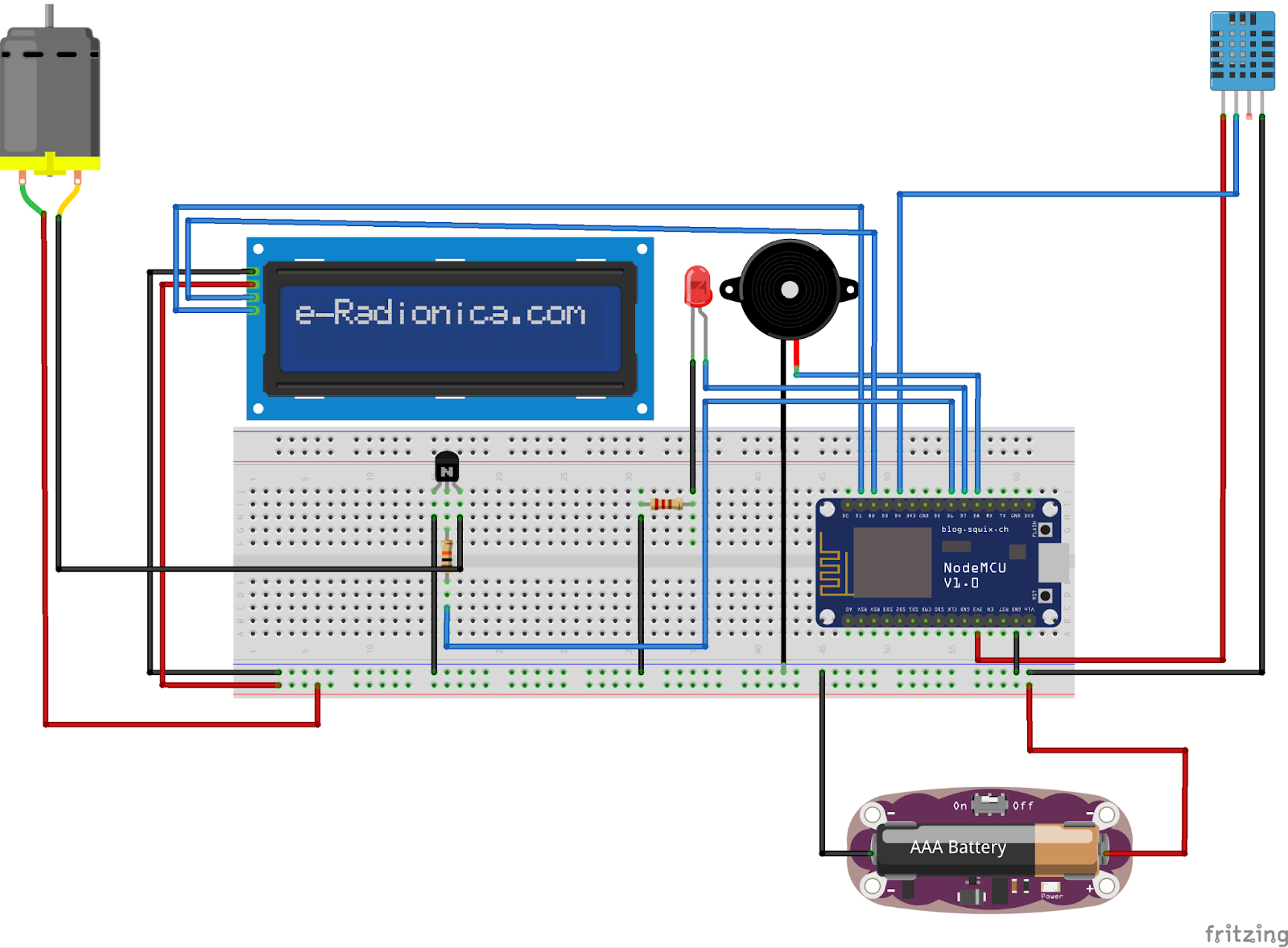
A breadboard is used to build and test circuits quickly before finalizing any circuit design. The breadboard has many holes into which circuit components like ICs and resistors can be inserted.

**REQUIRED SOFTWARES, PLATFORMS AND IDE**

Below are some of the softwares and IDE that would be used for this project;

* Arduino IDE
* Fritzing
* Google firebase
* MIT app Inventor

**SCHEMATIC VIEW OF THE CIRCUIT**



The above figure is a schematic view of the hardware circuit of this project, that shows how the various components are being interfaced with the Nodemcu microcontroller.

**HARDWARE CONNECTION OVERVIEW**

The concept of the project is self explanatory, we have the nodemcu microcontroller processing all of this information and sending data to the google firebase database. As we can see from the schematic diagram displayed above, the dht11 sensor is connected thus: Vcc to the 3v supply of the nodemcu, the Gnd connected to the Gnd supply of the microcontroller, and the data pin connected to pin 4 of the nodemcu. We should not forget to pay keen attention to the mapping of the nodemcu digital pins to that of arduino, since we are programming using the Arduino IDE.

Also we can see the buzzer which serves as an alarm trigger when the temperature exceeds a stipulated range, it is connected as thus; The Gnd pin is connected to the Gnd supply of the microcontroller, and the data pin is connected to pin 8 of the nodemcu.

An LED is also connected to this system, and it signifies that the system is actively connected to the internet. Its connection is thus; The negative pole of the LED is connected through a 330 ohms resistor to the Gnd supply of the microcontroller, and the positive pole connected to the digital pin 7 of the nodemcu.

A power supply is connected to the system to power up some components that require more than the 3v supplied by the nodemcu microcontroller.

Also, a 16x2 i2c LCD is connected to the system for on-site supervision of the temperature and humidity of the environment, it is connected as thus; The LCD needs a minimum of 5v power supply to come on, so its VCC is connected to the 5v supply that was provided on the nodemcu. Its Gnd is connected to the Gnd supplied. Its SDA and SCL pins are connected to pins 2 and 1 respectively.

Lastly, a DC motor is connected as a Fan. According to our program, the fan turns on when the  temperature exceeds a certain range. An NPN transistor is used to connect this fan to the system. The transistor is used as a switch to trigger the fan when the data pin it is connected to is made high. It is connected to pin 6 of the microcontroller.

If these steps are followed cautiously, we would be able to build this system and it would be working effectively, take note of the effect of noise on the system, make sure all data pins are properly connected, check that the pins match the description on the code.

N.B: Always check for the board you are uploading to before uploading the program.

Nodemcu accepts programs when it is sent to “NodeMCU 1.0 (ESP-12E Module)”.

**CODE VIA GITHUB**

Below is a link to the code for this project:

https://github.com/omenap/IOT-based-temperature-and-humidity-measuremnet-system.

**CODE OVERVIEW**

**#include <LiquidCrystal\_I2C.h>**

**#include <Adafruit\_Sensor.h>**

**#include <DHT.h>**

**#include <ESP8266WiFi.h>**

**#include <FirebaseESP8266.h>**

**#include <ESP8266WebServer.h>**

The above lines of codes are libraries that were called to help with the various components. A library is an already made set of codes we call or include in a program.

**ESP8266WebServer webserver(80);**

**LiquidCrystal\_I2C lcd(0x27,16,2);**

This part of the code initializes the nodemcu and the LCD, defining the resolution of the screen.

**lcd.init();**

**lcd.backlight();**

**dht.begin();**

**lcd.init();**

This part of the code initializes the LCD and DHT11 temperature sensor, it is like a startup trigger for the LCD backlight to come on, and the DHT11 temperature to begin.

**Firebase.begin(FIREBASE\_HOST, FIREBASE\_AUTH);**

This part of the code initializes the google firebase database and defines the data it needs.

**WiFi.mode(WIFI\_STA);**

**WiFi.begin("8pxX2", "coldmineral1");**

This part of the code defines the wifi mode as a wifi, the ESP8266 can also be used as a hotspot, but for this project wwe would be using it as a wifi. Also we declare the hotspot details we are connecting to  it can be changed.

**t = dht.readTemperature();**

**h = dht.readHumidity();**

This is the main part of the code, it triggers the DHT sensor to read the temperature and humidity of the environment and stores it in the variable declared for it.

**currenTime = millis();**

**if(currenTime - lasTime >= interval){**

**lasTime = currenTime;**

**lcdprint();**

**Firebase.setString(firebaseData,"Environment",str);**

**}**

This part of the code sets the system to send data at an interval using the millis time function. Data is only sent when the condition is meant, and it helps the concurrency process of the  system.

Other lines of the code are self explanatory, read carefully and follow all syntax and make changes when your hardware connection differs from the circuit diagram shown above.

**CONCLUSION**

Overall this project requires a lot of carefulness and cautiousness, because we are working with some sensitive components and microcontrollers. Diligence and patience are also important, sometimes little errors in the code segment can be very hectic to point out, that’s why this documentation is here to guide you though.