IOT BASED GARBAGE MONITORING USING ARDUINO

A PROJECT REPORT

of

Measurement and Instrumentation (EEE 2004)

by

SL.No.	Register Number	Name of the Student
1	19BEE0125	MUHAMMED ASHIR VP
2	19BEE0037	NADIR N K
3	17BEE0347	D SAI VARMA

Under the guidance of
PROF.THAMILMARAN

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF ELECTRIAL ENGINEERING

October, 2020

CERTIFICATE

This is to certify that the project work entitled "IOT-Based Garbage Monitoring using Arduino" that is being submitted by Muhammed Ashir VP, Nadir NK and D Sai Varma for Measurement and Instrumentation is a record of bonafide work done under my supervision. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted for any other CAL course.

Name of the Student	Reg. No	Signature
Muhammed Ashir. VP	19BEE0125	[illegible]
Nadir NK	19BEE0037	[illegible]
D Sai Varma	17BEE0347	[illegible]

[illegible]
(Thamilmaran.A)

Place: Vellore **Signature of Faculty**

Date: **PROF.THAMILMARAN**

TABLE OF CONTENTS

1. INTRODUCTION

We are living in an age where tasks and systems are fusing together with the power of IOT to have a more efficient system of working and to execute jobs quickly! With all the power at our finger tips this is what we have come up with.The Internet of Things (IoT) shall be able to incorporate transparently and seamlessly a large number of different systems, while providing data for millions of people to use and capitalize. Building a general architecture for the IoT is hence a very complex task, mainly because of the extremely large variety of devices, link layer technologies, and services that may be involved in such a system. One of the main concerns with our environment has been solid waste management which impacts the health and environment of our society. The detection, monitoring and management of wastes is one of the primary problems of the present era. The traditional way of manually monitoring the wastes in waste bins is a cumbersome process and utilizes more human effort, time and cost which can easily be avoided with our present technologies. This is our solution, a method in which waste management is automated. This is our IoT Garbage Monitoring system, an innovative way that will help to keep the cities clean and healthy.

1.1 The problem

Nowadays, there are tons of flats and apartments which have been built in the rapid urbanization area. This is due to high housing demands which have been drastically risen as a result of migration from villages to cities to find work. In order to accommodate the growing population in the urban area, the government has also constructed more apartment complexes. There are several issues faced by the residents of the flats. One of them is disposal of solid waste. Unlike private houses, the residents of all the apartments use a common dustbin, which tends to fill up very quickly. This overflowing of garbage is a sanitary issue which might cause diseases like cholera and dengue. Moreover it is a waste of fuel to travel around a complex or an area to find that some of the garbage are filled and some are not. Also, on rare days, problems might arise that there is so much garbage that the truck doesn't have enough capacity. The idea struck us when we observed that the garbage truck use to go around the town to collect solid waste twice a day. Although this system was thorough it was very inefficient. For example let's say street A is a busy street and we see that the garbage fills up really fast whereas maybe street B even after two days the bin isn't even half full. This example is something that actually happens thus it lead us to the "Eureka" moment!

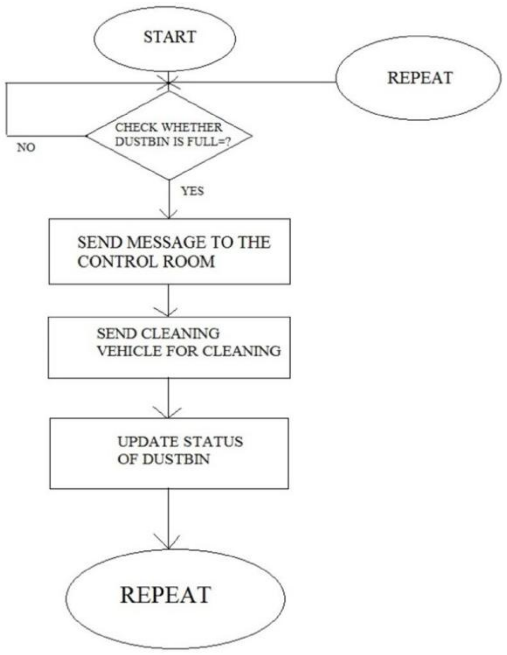

START
REPEAT
CHECK WHETHER DUSTBIN IS FULL=?
NO
YES
SEND MESSAGE TO THE CONTROL ROOM
SEND CLEANING VEHICLE FOR CLEANING
UPDATE STATUS OF DUSTBIN
REPEAT

Figure 1: Flow chart of project

What our system does is it gives a real time indicator of the garbage level in a trashcan at any given time. Using that data we can then optimize waste collection routes and ultimately reduce fuel consumption. It allows trash collectors to plan their daily/weekly pick up schedule. An **Ultrasonic Sensor** is used for detecting whether the trash can is filled with garbage or not. Here Ultrasonic Sensor is installed at the top of Trash Can and will measure the distance of garbage from the top of Trash can and we can set a threshold value according to the size of trash can.

If the distance will be less than this threshold value, means that the Trash can is full of garbage and we will print the message “Basket is Full” on the message and if the distance will be more than this threshold value, then we will print the distance remaining for the garbage vat to be full.

2. HARDWARE REQUIREMENTS

We will need the following hardware to accomplish our project.

1. HC-SR04 ultrasonic sensor.
2. Arduino Uno.
3. GSM module
4. Connecting wires.

2.1 ARDUINO UNO

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board – you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package. The Arduino is a microcontroller board based on the ATmega8. It has 14 digital -input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started .The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter .Revision 2 of the Uno board has a resistor pulling the 8U2HWB line to ground, making it easier to put into DFU mode. Revision of the board has the following new features:

Pin out: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible with both the board that uses the AVR, which operates with 5V and with the Arduino Due that operates with 3.3V. The second one is a not connected pin that is reserved for future purposes.

Stronger RESET circuit.

AT mega 16U2 replace the 8U2.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform.

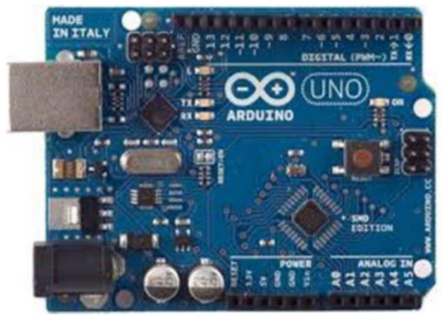


Figure 2: ARDUINO UNO BOARD

Parameters For Arduino UNODescription

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32KB(ATmega328) of which 0.5KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16MHz

Length	68.6 mm
Width	53.4 mm
Weight	53gm

Table 1. Specifications of Arduino

2.2 HC-SR04 ULTRASONIC SENSOR.

HC-SR04 is an ultrasonic sensor which is used for measuring the distance between the top of the lid to the top of the garbage.

PIN NUMBER	PIN NAME	DESCRIPTION
1.	VCC	The Vcc pin powers the sensor, typically with +5V
2.	Trigger	Trigger pin is an Input pin. This pin has to be kept high for 10us to initialize measurement by sending US wave.
3.	Echo	Echo pin is an Output pin. This pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor.
4.	GND	This pin is connected to the Ground of the system.

Table 2: Pin Number and Function of Ultrasonic sensor

2.2.2 HC-SR04 SENSOR FEATURES.

Operating voltage: +5V

Theoretical Measuring Distance: 2cm to 450cm

Accuracy: 3mm

Measuring angle covered: <15°

Operating Current: <15ma

Operating Frequency: 40Hz

2.2.3 ULTRASONIC SENSOR WORKING

The **HC-SR04 Ultrasonic (US) sensor** is a 4 pin module, whose pin names are Vcc, Trigger, Echo and Ground respectively. This sensor is a very popular sensor used in many applications where measuring distance or sensing objects are required. The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver. The sensor works with the simple high school formula that

Distance = Speed × Time

The Ultrasonic transmitter transmits an ultrasonic wave, this wave travels in air and when it gets objected by any material it gets reflected back toward the sensor this reflected wave is observed by the Ultrasonic receiver module.Now, to calculate the distance using the above formulae, we should know the Speed and time. Since we are using the Ultrasonic wave we know the universal speed of US wave at room conditions which is 330m/s. The circuitry inbuilt on the module will calculate the time taken for the US wave to come back and turns on the echo pin high for that same particular amount of time, this way we can also know the time taken. Now simply calculate the distance using a microcontroller or microprocessor.

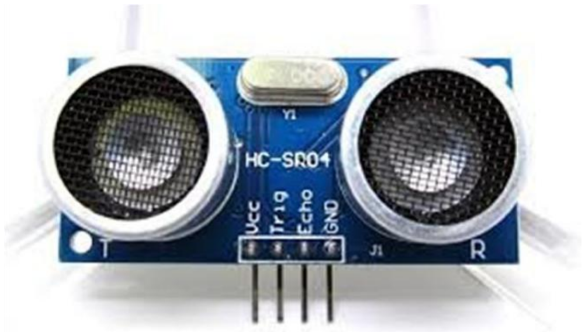


Figure 3: HCSR04 ULTRASONIC SENSOR

2.3 GSM MODULE.

GSM/GPRS module is used to establish communication between a computer and a **GSM-GPRS system**. **Global System for Mobile communication (GSM)** is an architecture used for mobile communication in most of the countries. **Global Packet Radio Service (GPRS)** is an extension of GSM that enables higher data transmission rate. **GSM/GPRS module consists of a**

GSM/GPRS modem assembled together with power supply circuit and communication interfaces (like RS-232, USB, etc) for computer. GSM/GPRS MODEM is a class of wireless MODEM devices that are designed for communication of a computer with the GSM and GPRS network. It requires a **SIM (Subscriber Identity Module)** card just like mobile phones to activate communication with the network. Also they have **IMEI** (International Mobile Equipment Identity) number similar to mobile phones for their identification. A GSM/GPRS MODEM can perform the following operations:

1. Receive, send or delete SMS messages in a SIM.
2. Read, add, search phonebook entries of the SIM.
3. Make, Receive, or reject a voice call.

The MODEM needs **AT commands**, for interacting with processor or controller, which are communicated through serial communication. These commands are sent by the controller/processor. The MODEM sends back a result after it receives a command. Different AT commands supported by the MODEM can be sent by the processor/controller/computer to interact with the **GSM and GPRS cellular network**.

A GSM modem is a wireless modem that works with a GSM wireless network. A wireless modem behaves like a dial-up modem. The main difference between them is that a dial-up modem sends and receives data through a fixed telephone line while a wireless modem sends and receives data through radio waves.

A GSM modem can be an external device or a PC Card / PCMCIA Card. Typically, an external PC Card / PCMCIA Card is designed for use with a laptop computer. It should be inserted into one of the PC Card / PCMCIA Card slots of a laptop computer. Like a GSM mobile phone, a GSM modem requires a SIM card from a wireless carrier in order to operate.

A SIM card contains the following information:

Subscriber telephone number (MSISDN)

International subscriber number **(IMSI, International Mobile Subscriber Identity)**

State of the SIM card

Service code (operator)

Authentication key

PIN (*Personal Identification Code*)

PUK (*Personal Unlock Code*)

Computers use AT commands to control modems. Both GSM modems and dial-up modems support a common set of standard AT commands. In addition to the standard AT commands, GSM modems support an extended set of AT commands. These extended AT commands are defined in the GSM standards. With the extended AT commands, the following operations can be performed:

Reading, writing and deleting SMS messages.

Sending SMS messages.

Monitoring the signal strength.

Monitoring the charging status and charge level of the battery.

Reading, writing and searching phone book entries.

Figure 4: GSM MODULE

2.4.DHT11 sensor:

DHT11 is a Humidity and Temperature Sensor, which generates calibrated digital output. DHT11 can be interface with any microcontroller like Arduino, Raspberry Pi, etc. and get instantaneous results. DHT11 is a low cost humidity and temperature sensor which provides high reliability and long term stability. The DHT11 Humidity and Temperature Sensor consists of 3 main components. A resistive type humidity sensor, an NTC (negative temperature coefficient) thermistor (to measure the temperature) and an 8-bit microcontroller, which converts the analog signals from both the sensors and sends out single digital signal. This digital signal can be read by any microcontroller or microprocessor for further analysis.DHT11 Humidity Sensor consists of 4 pins: VCC, Data Out, Not Connected (NC) and GND. The range of voltage for VCC pin is 3.5V to 5.5V. A 5V supply would do fine. The data from the Data Out pin is a serial digital data. DHT11 Sensor can measure a humidity value in the range of 20 – 90% of Relative Humidity (RH) and a temperature in the range of 0 – 50^{0}C. The sampling period of the sensor is 1 second. All the DHT11 Sensors are accurately calibrated in the laboratory and the results are stored in the memory. A single wire communication can be established between any microcontroller like Arduino and the DHT11 Sensor. Also, the length of the cable can be as long as 20 meters. The data from the sensor consists of integral and decimal parts for both Relative Humidity (RH) and temperature.

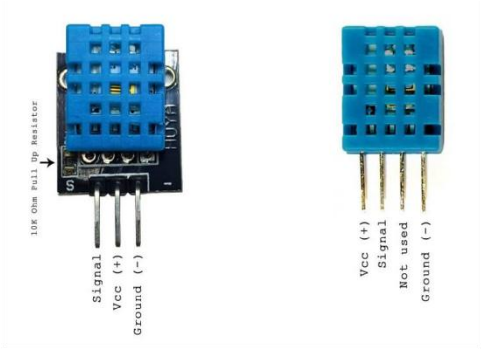
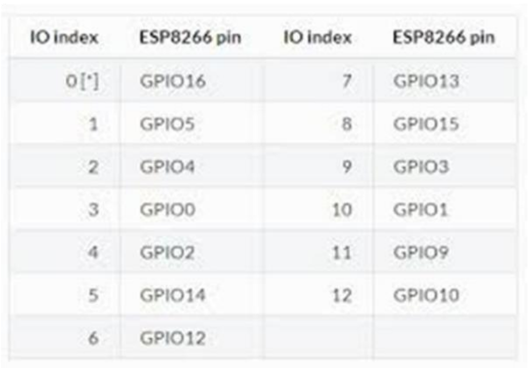


Figure 5: DHT 11 pin out

2.3.5 Node MCU:

NodeMCU is an open source IoT platform. It includes firmware which runs on the ESP8266Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the development kits. The firmware uses the Lua scripting language. It is based on the eLua project, and built on the Espress-if Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson, and spiffs.



IO index	ESP8266 pin	IO index	ESP8266 pin
0 [*]	GPIO16	7	GPIO13
1	GPIO5	8	GPIO15
2	GPIO4	9	GPIO3
3	GPIO0	10	GPIO1
4	GPIO2	11	GPIO9
5	GPIO14	12	GPIO10
6	GPIO12		

Figure 6: NodeMCU PINOUT

Node MCU provides a way to connect different sensors to their controllers wirelessly via wifi. Since, it is an improved version of the ESP8266 it has better and easier programming, with better voltage stability and more reliability.

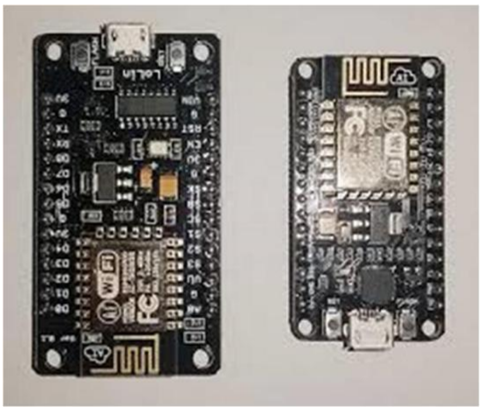


Figure 7: Node MCU

3. DESIGN AND MODELLING

In this section we design structure of the system before implementation of circuit. we use advanced microcontroller called Arduino (ATmega8). It has in built with many components like analog to digital converter, clock of 16 MHz, shift registers.

In this project we put the ultrasonic sensor on top of the garbage bin/ dump. The output of the ultrasonic sensor is processed by the Arduino and the output is then sent to the GSM module which sends a text message to the concerned person. We have a threshold value of 5cm.Which means that if the distance of the sensor from the top of the garbage is less than 5cm, the output will come with a message that the basket is full. Also, a buzzer will ring if output is less than 5cm. The DHT11 sensor will show the temperature and the humidity.

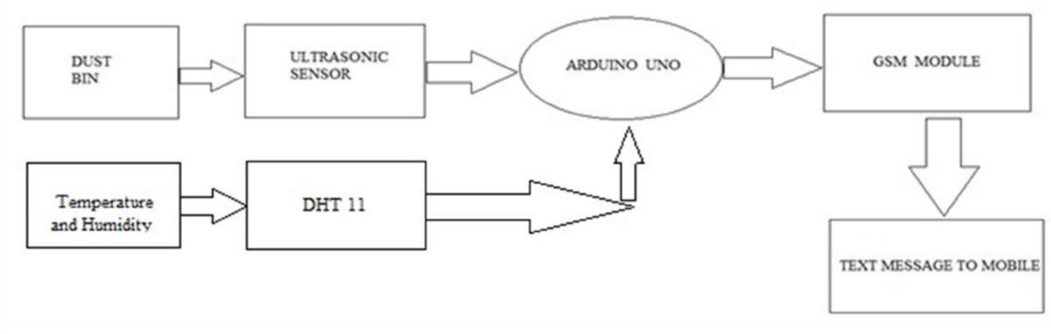


Figure 8: project diagram

4. IMPLEMENTATION

4.1 Hardware implementation

Connections of the ultrasonic sensor with the Arduino are very simple. Connect the VCC and the ground of the ultrasonic sensor to the 5V and the ground of the Arduino. Then connect the TRIG and ECHO pin of ultrasonic sensor to the pin 11 and 12 of the Arduino respectively(you can use any other pin as well).Connect the RX pin of the arduino with the TX pin of the GSM module and the TX pin of the arduino with the RX pin of the GSM module. Connect the GND of the arduino to the ground of the module. Also,the GSM module needs an external 12v supply.

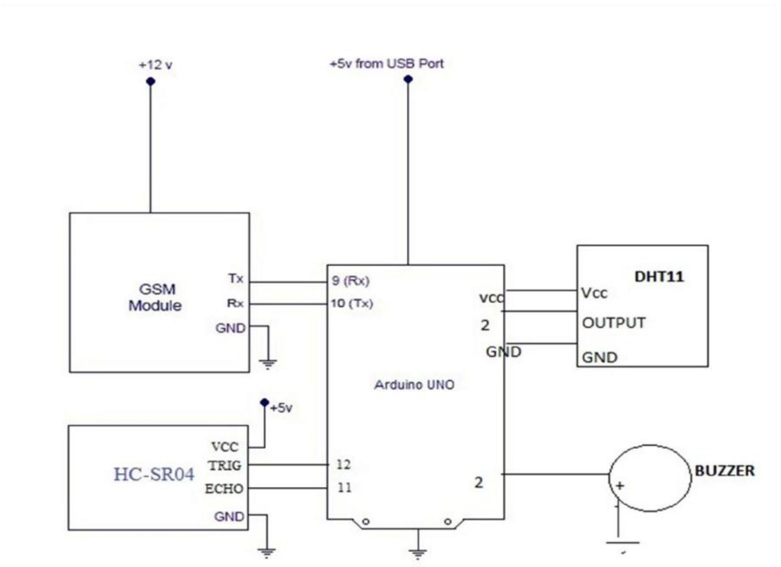


Figure 9: circuit diagram

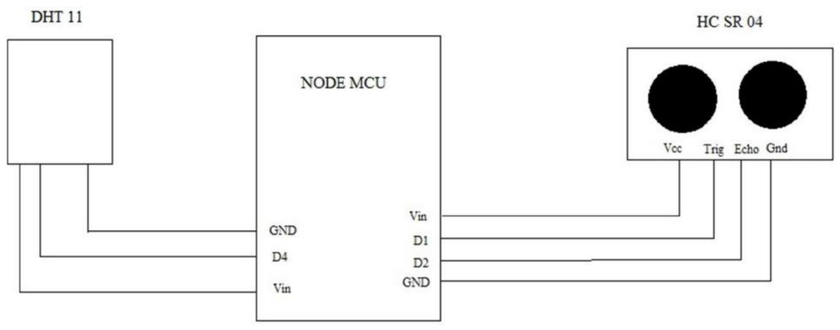


Figure 10: circuit diagram of node MCU with DHT11 and HCSR04

4.2 Software implementation.

The software required for it is the Arduino IDE.

4.2.1 ARDUINO IDE

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board. The Arduino development environment contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions, and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.Software written using Arduino are called sketches. These sketches are written in thetext editor. Sketches are saved with the file extension .ino. It has features for cutting/pastingand for searching/replacing text. The message area gives feedback while saving and exportingand also displays errors. The console displays text output by the Arduino environmentincluding complete error messages and other information. The bottom right-hand corner ofthe window displays the current board and serial port. The toolbar buttons allow you to verifyand upload programs, create, open, and save sketches, and open the serial monitor.

4.2.2 Source Code

```
#include <SoftwareSerial.h>
#include <DHT.h>
#define DHTPIN 2

#define DHTTYPE DHT11
constint buzzer = 6; //buzzer to arduino pin 6
DHT dht(DHTPIN, DHTTYPE);

SoftwareSerialmySerial(9, 10);
inttrigPin = 12;
constintechoPin = 11;
intchk;
float hum; //Stores humidity value

float temp;
// defines variables
long duration;
int distance;
void setup() {
pinMode(buzzer, OUTPUT);
mySerial.begin(9600); // Setting the baud rate of GSM Module
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
pinMode(echoPin, INPUT); // Sets the echoPin as an Input
Serial.begin(9600); // Starts the serial communication dht.begin();

delay(100);
}
```

```
void loop() {
// Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// Reads the echoPin, returns the sound wave travel time in microseconds
duration = pulseIn(echoPin, HIGH);
// Calculating the distance
distance= duration*0.034/2;
// Prints the distance on the Serial Monitor
if (distance==5)
{
tone(buzzer, 1000); // Send 1KHz sound signal...
delay(1000);        // ...for 1 sec
noTone(buzzer);     // Stop sound...
delay(1000); Serial.print("Distance:
"); Serial.println(distance);
Serial.print(" cm");

hum = dht.readHumidity();
temp= dht.readTemperature();

 //Print temp and humidity values to serial monitor
Serial.print("Humidity: ");
Serial.print(hum);
Serial.print(" %, Temp: ");
Serial.print(temp);
Serial.println(" Celsius");
mySerial.println("AT+CMGF=1");     //Sets the GSM Module in Text  Mode  delay(1000);
// Delay of 1000 milli seconds or 1 second
mySerial.println("AT+CMGS=\"+917059759945\"\r"); // Replace x with mobile number
delay(1000);
mySerial.println("Distance: ");// The SMS text you want to send
mySerial.print(distance);
mySerial.print(" cm ");// The SMS text you want to send
mySerial.println("Humidity: ");// The SMS text you want to send
mySerial.print(hum);
mySerial.print(" %, Temp: ");// The SMS text you want to send
mySerial.print(temp);
delay(100);
mySerial.println((char)26);// ASCII code of CTRL+Z
delay(5000);

delay(2000);
}
}
```

4.2.3 Blynk app

Blynk is a Platform with iOS and Android apps to control Arduino, Raspberry Pi and the likes over the Internet.It's a digital dashboard where you can build a graphic interface for your project by simply dragging and dropping widgets.Blynk is not tied to some specific board or shield. Instead, it's supporting hardware of your choice. Whether your Arduino or Raspberry Pi is linked to the Internet over Wi-Fi, Ethernet or this new ESP8266 chip, Blynk will get you online and ready for the Internet Of Your Things. Blynk was designed for the Internet of Things. It can control hardware remotely, it can display sensor data, it can store data, visualize it and do many other cool things.

There are three major components in the platform:

Blynk App - allows to you create amazing interfaces for your projects using various widgets we provide.

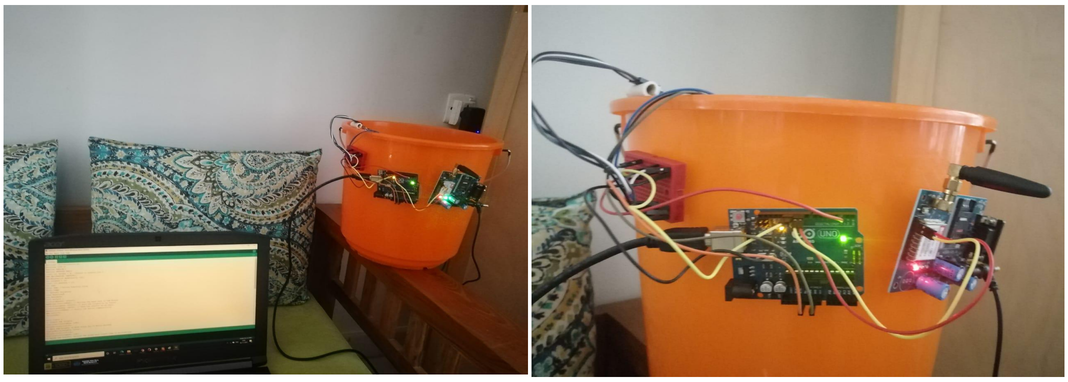
Blynk Server - responsible for all the communications between the smart phone and hardware. You can use our Blynk Cloud or run your private Blynk server locally. It's open-source, could easily handle thousands of devices and can even be launched on a Raspberry Pi.

Blynk Libraries - for all the popular hardware platforms - enable communication with the server and process all the incoming and out coming commands.

Source code for dht11 and NodeMCU shown via Blynk:

```
#define BLYNK_PRINT Serial


#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include "DHT.h"          // including the library of DHT11 temperature and humidity sensor
#include <SimpleTimer.h> //including the library of SimpleTimer
#define DHTTYPE DHT11      // DHT 11

#define dht_dpin 14
DHT dht(dht_dpin, DHTTYPE);
SimpleTimer timer;
charauth[] = "Your Auth. Key";          // You should get Auth Token in the Blynk App.
                                  // Go to the Project Settings (nut icon).

charssid[] = "Your Wifi Network name";    // Your WiFi credentials.
char pass[] = "Password of your network"; // Set password to "" for open networks.
float t;                          // Declare the variables
float h;



void setup()
{
  Serial.begin(9600);// Debug console
  Blynk.begin(auth, ssid, pass);
  dht.begin();
  timer.setInterval(2000, sendUptime);
}
```

```
voidsendUptime()
{

  float h = dht.readHumidity();
  float t = dht.readTemperature();
  Serial.println("Humidity and temperature\n\n");
  Serial.print("Current humidity = ");
  Serial.print(h);
  Serial.print("% ");
  Serial.print("temperature = ");
  Serial.print(t);
  Blynk.virtualWrite(V0, t);
  Blynk.virtualWrite(V1, h);

}

void loop()
{
  Blynk.run();
  timer.run();
}
```



5.SYSTEM TESTING

5.1 Test approach

We will test the project in two stages: software and hardware. The software part is to be tested via the Arduino IDE, whereas the hardware part has to be tested physically. It is necessary to check whether the system is working properly or not. To check whether the readings are accurate, we will check the distance pointed out by the sensor by a meter tape.

5.2 Features to be tested

After building the whole circuit we test it, testing procedure is given in 5.1. This project should satisfy some features. Features to be tested as follows:

The ultrasonic sensor should give proper output. To check whether the output is accurate or not, the output of the sensor will be checked against a meter tape.

The arduino board should show the distance in the serial monitor. So should the NodeMCU.

The GSM module should send messages after the specified delay. If the text messages are reaching the phone, that means the GSM module is working. It should make a small ringing sound, when it sends messages.

The DHT11 sensor should work properly and show its output in the serial monitor.

The blynk app should be checked.

5.2.2 Testing tools and environment

For testing of the project we require some tools, like to test Arduino program we require a software called Arduino IDE. Using this we can check the program that program is working properly or not. For hardware checking we require power supply and proper range of measurements and a meter tape. The garbage dump should have only solid waste.

The NodeMCU should connect to the Blynk app and the app should show the output. For this ,theNodeMCU must connect first to the wifi hotspot.

5.3 Test cases.

In this section we discuss about the inputs, expected output, testing procedure.

5.3.1 Inputs

This project requires three inputs:

1. Power supply:Power supply is the basic need of any electronic circuit. Here we use 5v dc battery to give power Arduino and sometimes we can give power directly from the computer. We also need a 12V power supply for the GSM module.
2. We can also power these circuits via two 9v batteries using a circuit divider.Distance, The distance will be the input of the Arduino circuit and will be gotten from the ultrasonic sensor.
3. The temperature and humidity from the DHT11 sensor.

5.3.2 Expected output

The expected output of this project should be a text message showing the distance to full. Also, it will also send the humidity and the temperature of the area. The output should also be seen on the serial monitor of the Arduino IDE. Also, the output should also be seen on the serial monitor and also on the Blynk app.

5.3.3 Testing procedure:

For testing first connect the circuit to the power supply is given to the Arduino using computer and it can be done by using battery. In this way the whole testing circuit is built. Now we give input to the HC-SR04 by changing the level of solid garbage.. Change in garbage levels should be messaged using GSM Module.

Summary of testing procedure:-

1) Connect the circuit according to the diagram

2) Give power to the system.

3) Vary garbage level for the ultrasonic sensor to give output.

4) Get the output from the DHT11 sensor.

5) Send message via the GSM module.

6. CONCLUSION.

We built an efficient garbage monitoring system which can be used to monitor the level of garbage in the dump. This data can be further used to plan garbage collection trips more efficiently, ultimately reducing overflowing bins and helping have better public sanitation.

Advantages:

Very simple circuit.

The HCSR04 sensor is very rugged.

Helps monitor garbage levels.

Uses very small amount of electricity.

Ultimately helps in better planning of garbage pickups.

Can help in reducing overflowing bins.

Reduces trips to areas where the bins still have a lot of capacity.

Disadvantages:

Cannot detect liquid waste.

Only detects the top of the garbage level. It wouldn’t realize if there is space left.

GSM module needs a 12v source.

7. Experiment Results

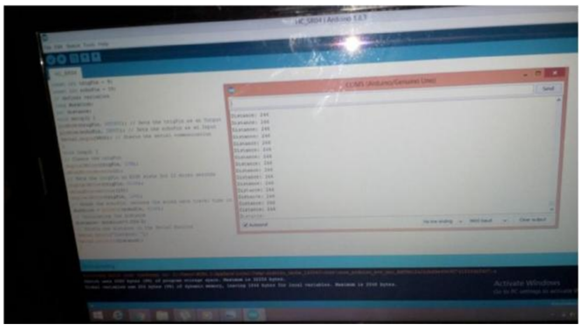
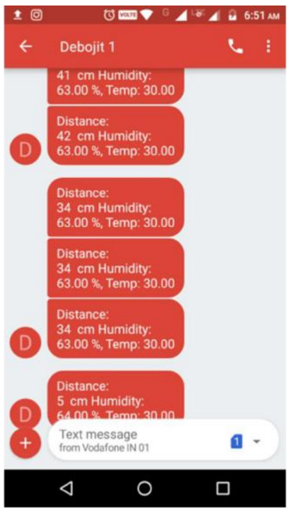


Figure 13: Output as a text message on phone.



Debojit 1

41 cm Humidity: 63.00 %, Temp: 30.00

Distance: 42 cm Humidity: 63.00 %, Temp: 30.00

Distance: 34 cm Humidity: 63.00 %, Temp: 30.00

Distance: 34 cm Humidity: 63.00 %, Temp: 30.00

Distance: 34 cm Humidity: 63.00 %, Temp: 30.00

Distance: 5 cm Humidity: 64.00 %, Temp: 30.00

Text message from Vodafone IN 01

Figure 14: Output of sensor in serial monitor

The output of this project should have been the values of the distance, temperature and humidity, which we were supposed to get via a text message. For, the GSM Module to send data, the sensors must be working perfectly which can be seen in figure 13. This brings us to figure 14 in which we see the text messages sent from the GSM Module. Also, the output in Blynk app is shown in figure 11, which shows the real time data on the app, via WIFI.

REFERENCES

Navghane S S, Killedar M S and Rohokale D V 2016 IoT Based Smart Garbage and waste collection, International Journal of Advanced Research in Electronics And Communication.

Monika K A, Rao N, Prapulla S B and Shobha G 2016 Smart Dustbin-An EfficientGarbage Monitoring System International Journal of Engineering Science andComputing 6 7113-16.

Medvedev A, Fedchenkov P, Zaslavsky A, Anagnostopoulos T and Khoruzhnikov S,2015 Waste management as an IoT-enabled service in smart cities In Conference on Smart Spaces Springer International Publishing 104-15.

www.buildofy.in/smart_home_designs

https://create.arduino.cc/projecthub/Technovation/smart-garbage-monitoring-system-using-arduino-101-3b813c

https://github.com/sourabhdeshmukh/Smart-Dustbin

http://invent.module143.com/temperature-and-humidity-using-nodemcu-blynk/

http://help.blynk.cc/getting-started-library-auth-token-code-examples/blynk-basics/what-is-virtual-pins

http://help.blynk.cc/getting-started-library-auth-token-code-examples/blynk-basics/how-to-display-any-sensor-data-in-blynk-app

http://help.blynk.cc/how-to-connect-different-hardware-with-blynk/esp8266/nodemcu