In this [IOT Project](http://circuitdigest.com/internet-of-things-iot-projects), 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 webpage and if the distance will be more than this threshold value, then we will print the message “Basket is Empty”. Here we have set the Threshold value of 5cm in the Program code. We will use **ESP8266 Wi-Fi module** for connecting the Arduino to the webserver. Here we have used Local webserver to demonstrate the working of this Garbage Monitoring System

In effort to reduce waste and increase recycling, industry,countries need some platform through which they could manage waste and recycleand increase their knowledge about where item should go to avoid landfill buildup.It will make easy for user to find nearby recycling locations for their materials are grouped into categories.This can be done by our android appGreen Chennai where there will be certain features in the app.This app will contain certain features like automatic reminder for next turn for recycling of waste. This application will be a reminder depending on the time one chooses.

Another feature is entering your location through GPS so that people can complain about the garbage near their locality on which no one pays attention. After entering the location one can get the contact number in the near area of the private garbage collectors.

Another section will be there about processes whether it was carried out effectively or not, like people can actually comment over the section about their condition.

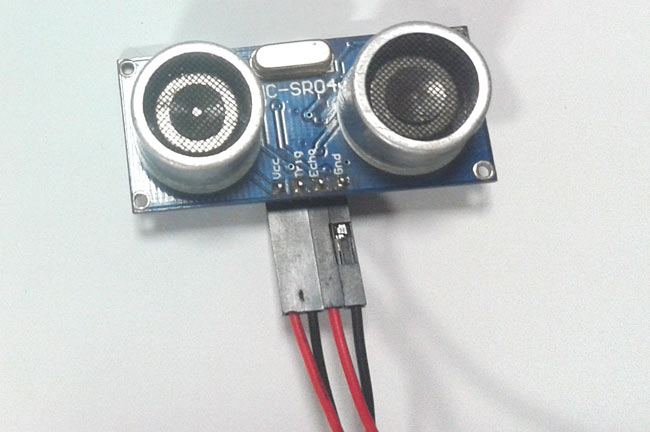
In this way, all can come together to participate for creating a hygienic place.

**Components Required:**

* Arduino Uno (you can use any other)
* ESP8266 Wi-Fi module
* HC-SR04 Ultrasonic sensor
* 1K Resistors
* Breadboard
* Connecting wires

**HC-SR04 Ultrasonic Sensor:**

The [Ultrasonic Sensor](http://circuitdigest.com/tags/ultrasonic-sensor) is used to measure the distance with high accuracy and stable readings. It can measure distance from 2cm to 400cm or from 1 inch to 13 feet.  It emits an ultrasound wave at the frequency of 40KHz in the air and if the object will come in its way then it will bounce back to the sensor. By using that time which it takes to strike the object and comes back, you can calculate the distance.



The ultrasonic sensor has four pins. Two are VCC and GND which will be connected to the 5V and the GND of the Arduino while the other two pins are Trig and Echo pins which will be connected to any digital pins of the Arduino. The trig pin will send the signal and the Echo pin will be used to receive the signal. To generate an ultrasound signal, you will have to make the Trig pin high for about 10us which will send a 8 cycle sonic burst at the speed of sound and after striking the object, it will be received by the Echo pin.

Further check below projects to properly understand the working of Ultrasonic sensor and to measure the distance of any object using it:

* [Arduino Based Distance Measurement using Ultrasonic Sensor](http://circuitdigest.com/microcontroller-projects/arduino-ultrasonic-sensor-based-distance-measurement)
* [Distance Measurement using HC-SR04 and AVR Microcontroller](http://circuitdigest.com/microcontroller-projects/distance-measurement-using-hc-sr04-avr)

**ESP8266 Wi-Fi Module:**

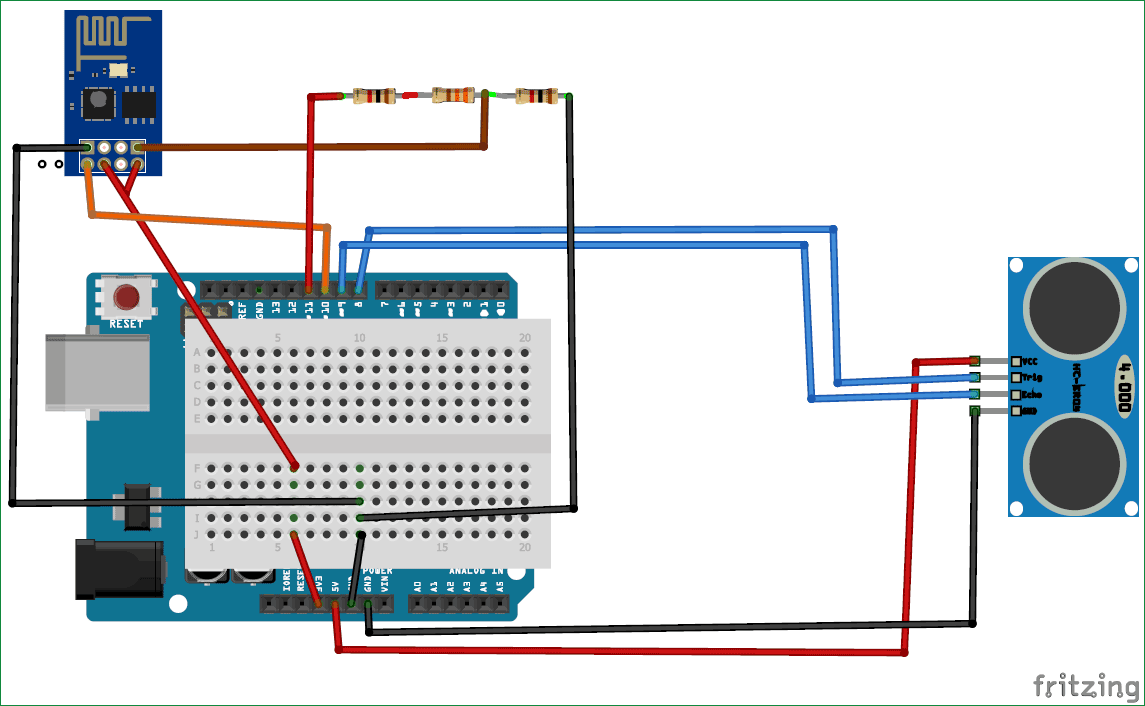
ESP8266 is a Wi-Fi module which will give your projects **access to Wi-Fi or internet**. It is a very cheap device but it will make your projects very powerful. It can communicate with any microcontroller and make the projects wireless. It is in the list of most leading devices in the IOT platform. It runs on 3.3V and if you will give it 5V then it will get damage

The ESP8266 has 8 pins; the VCC and CH-PD will be connected to the 3.3V to enable the wifi. The TX and RX pins will be responsible for the [communication of ESP8266 with the Arduino](http://circuitdigest.com/microcontroller-projects/sending-arduino-data-to-webpage). The RX pin works on 3.3V so you will have to make a voltage divider for it as we made in our project.

**Circuit Diagram and Explanation:**

First of all we will connect the ESP8266 with the Arduino. ESP8266 runs on 3.3V and if you will give it 5V from the Arduino then it won’t work properly and it may get damage. Connect the VCC and the CH\_PD to the 3.3V pin of Arduino. The RX pin of ESP8266 works on 3.3V and it will not communicate with the Arduino when we will connect it directly to the Arduino. So, we will have to make a voltage divider for it. Three 1k resistors connected in series will do the work for us. Connect the RX to the pin 11 of the Arduino through the resistors as shown in the figure below and also the TX of the Arduino to the pin 10 of the Arduino.

Now it’s time to connect the HC-SR04 ultrasonic sensor with the Arduino. 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 8 and 9 of the Arduino respectively.

[](https://circuitdigest.com/fullimage?i=circuitdiagram_mic/IoT-Arduino-Garbage-Monitor-circuit.png)

**Code Explanation:**

Before uploading the code, make sure that you are connected to the Wi-Fi of your ESP8266 device. You can check the full code in Code section below, code has been well explained by the comments, further we have also explained some important functions below.

The Arduino will first read the Ultrasonic Sensor. It will send a ultrasonic signal at the speed of sound when we will make the TRIG pin high for 10us. The signal will comeback after striking the object and we will store the travel time duration in the variable named *duration*. Then we will calculate the distance of object (garbage in our case) by applying a formula and will store it in the variable named *distance*.

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

distance= duration\*0.034/2;

For [printing the output on the webpage](http://circuitdigest.com/microcontroller-projects/sending-arduino-data-to-webpage) in web browser, we will have to use **HTML programming**. So, we have created a string named *webpage* and stored the output in it. To tell whether the trash can is empty or not, we have applied a condition there. **If the distance will be less than 5cm then it will show “Basket is Full” on the webpage** and if the distance will be greater than 5cm then it will show the message “Basket is Empty” on webpage.

if(esp8266.available())

{

if(esp8266.find("+IPD,"))

{

delay(1000);

int connectionId = esp8266.read()-48;

String webpage = "<h1>IOT Garbage Monitoring System</h1>";

webpage += "<p><h2>";

if (distance<5)

{

webpage+= " Trash can is Full";

}

else{

webpage+= " Trash can is Empty";

}

webpage += "</h2></p></body>";

The following code will send and show the data on the webpage. The data, we stored in string named *‘webpage’,* will be saved in string named *‘command’*. The ESP8266 will then read the character one by one from the *‘command’* and will print it on the webpage.

String sendData(String command, const int timeout, boolean debug)

{

String response = "";

esp8266.print(command);

long int time = millis();

while( (time+timeout) > millis())

{

while(esp8266.available())

{

char c = esp8266.read();

response+=c;

}

}

if(debug)

{

Serial.print(response);

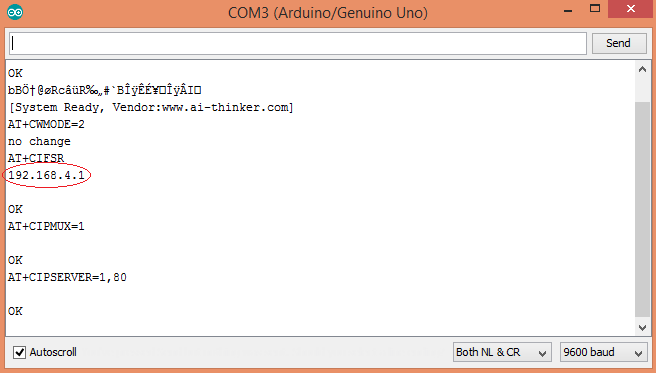
}

return response;

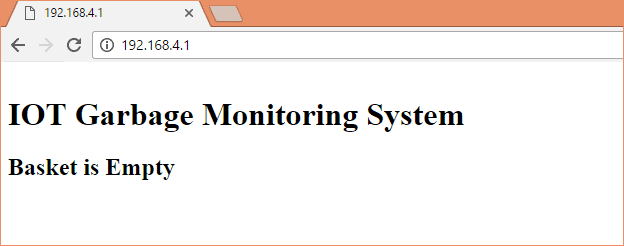
}

**Testing and Output of the Project:**

After uploading the code, open the Serial Monitor and it will show you an IP address as shown below.



Type this IP address in your browser, it will show you the output as shown below. You will have to refresh the page again if you want to see again that the trash can is empty or not.



So this how this **Garbage Monitoring System** works, this project can be further enhanced by adding few more features in it like we can set one more message when the Trash Can is half filled or we can trigger a Email/SMS to alert the user when Trash Basket is full.

Code

#include <SoftwareSerial.h>         // including the library for the software serial  
#define DEBUG true  
SoftwareSerial esp8266(10,11);      /\* This will make the pin 10 of arduino as RX pin and  
pin 11 of arduino as the TX pin Which means that you have to connect the TX from the esp8266  
to the pin 10 of arduino and the Rx from the esp to the pin 11 of the arduino\*/  
                                     
const int trigPin = 8;            // Making the arduino's pin 8 as the trig pin of ultrasonic sensor  
const int echoPin = 9;            // Making the arduino's pin 9 as the echo pin of the ultrasonic sensor  
// defining two variable for measuring the distance  
long duration;  
int distance;

void setup()  
{  
  Serial.begin(9600);         // Setting the baudrate at 9600  
  esp8266.begin(9600);        // Set the baudrate according to you esp's baudrate. your esp's baudrate might be different from mine  
  pinMode(trigPin, OUTPUT);   // Setting the trigPin as Output pin  
  pinMode(echoPin, INPUT);    // Setting the echoPin as Input pin  
    
  sendData("AT+RST\r\n",2000,DEBUG);            // command to reset the module  
  sendData("AT+CWMODE=2\r\n",1000,DEBUG);       // This will configure the mode as access point  
  sendData("AT+CIFSR\r\n",1000,DEBUG);          // This command will get the ip address  
  sendData("AT+CIPMUX=1\r\n",1000,DEBUG);       // This will configure the esp for multiple connections  
  sendData("AT+CIPSERVER=1,80\r\n",1000,DEBUG); // This command will turn on the server on port 80  
}

void loop()  
{  
digitalWrite(trigPin, LOW);   // Making the trigpin as low  
delayMicroseconds(2);         // delay of 2us  
digitalWrite(trigPin, HIGH); // making the trigpin high for 10us to send the signal   
delayMicroseconds(10);  
digitalWrite(trigPin, LOW);     
duration = pulseIn(echoPin, HIGH);  // reading the echopin which will tell us that how much time the signal takes to come back

distance= duration\*0.034/2;         // Calculating the distance and storing in the distance variable  
    
  if(esp8266.available())         // This command will that check if the esp is sending a message   
  {      
    if(esp8266.find("+IPD,"))  
    {  
     delay(1000);  
     int connectionId = esp8266.read()-48; /\* We are subtracting 48 from the output because the read() function returns   
                                            the ASCII decimal value and the first decimal number which is 0 starts at 48\*/  
     String webpage = "<h1>IOT Garbage Monitoring System</h1>";  
       webpage += "<p><h2>";     
       if (distance<5)  
       {  
        webpage+= " Trash can is Full";  
        }  
        else{  
          webpage+= " Trash can is Empty";  
          }  
       webpage += "</h2></p></body>";    
     String cipSend = "AT+CIPSEND=";  
     cipSend += connectionId;  
     cipSend += ",";  
     cipSend +=webpage.length();  
     cipSend +="\r\n";

     sendData(cipSend,1000,DEBUG);  
     sendData(webpage,1000,DEBUG);      
     String closeCommand = "AT+CIPCLOSE=";   
     closeCommand+=connectionId;   
     closeCommand+="\r\n";  
     sendData(closeCommand,3000,DEBUG);  
    }  
  }  
}  
   
String sendData(String command, const int timeout, boolean debug)  
{  
    String response = "";     
    esp8266.print(command);   
    long int time = millis();  
    while( (time+timeout) > millis())  
    {  
      while(esp8266.available())  
      {  
        char c = esp8266.read();   
        response+=c;  
      }    
    }  
    if(debug)  
    {  
      Serial.print(response);  
    }  
    return response;  
}