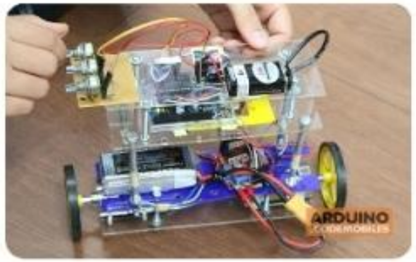
ARDUINO : IoT

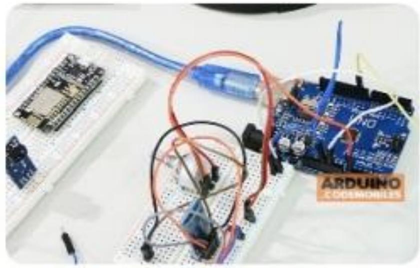
(Arduino + NodeMCU)

SHREEJI CHARAN electronics



Core Arduino

IoT Trend



Core Arduino

Arduino
+NodeMCU

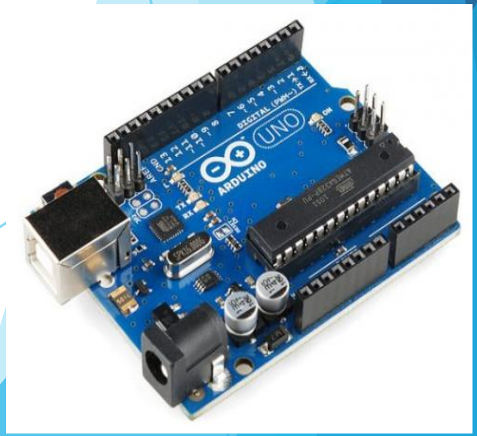
python™

Presented By:

Keyur Chauhan
Shreeji Charan Electronics Pvt. Ltd,
F16, Pooja Estate, Near Savera hotel,
Vallabh Vidhyanagar, Anand
Contact Us : 9978844178

ARDUINO
Genuino
AN OPEN PROJECT WRITTEN, DEBUGGED, AND SUPPORTED BY ARDUINO.CC AND THE ARDUINO COMMUNITY WORLDWIDE
LEARN MORE ABOUT THE CONTRIBUTORS OF ARDUINO.CC on arduino.cc/credits

SHREEJICHARAN ELECTRONICS 9978844178



IMPORTANT FOR IOT USING NODEMCU

1. INSTALL ARDUINO IDE 1.6.5 or Latest version of Arduino IDE
2. DOWNLOAD NODEMCU BOARD FORM BOARD MANAGER
3. DOWNLOAD THE ESP8166WIFI LIBRARY
4. CRATE THINGSPEAK LOGIN ID TO PLOT SENSOR DATA GRAPH
5. CRATE MIT APPS LOGIN ID TO CREAT OWN WIFI APPLICATION

BASIC Requirements

- INTERNET - PC + ROUTER
- SOFTWARE – ARDUINO UNO WITH NODEMCU AND ESP8266WIFI LIB + XAMPP
- PROTEUS SIMULATION SOFTWARE

DOWNLOAD NODEMCU BOARD FORM BOARD MANAGER -STEP-1

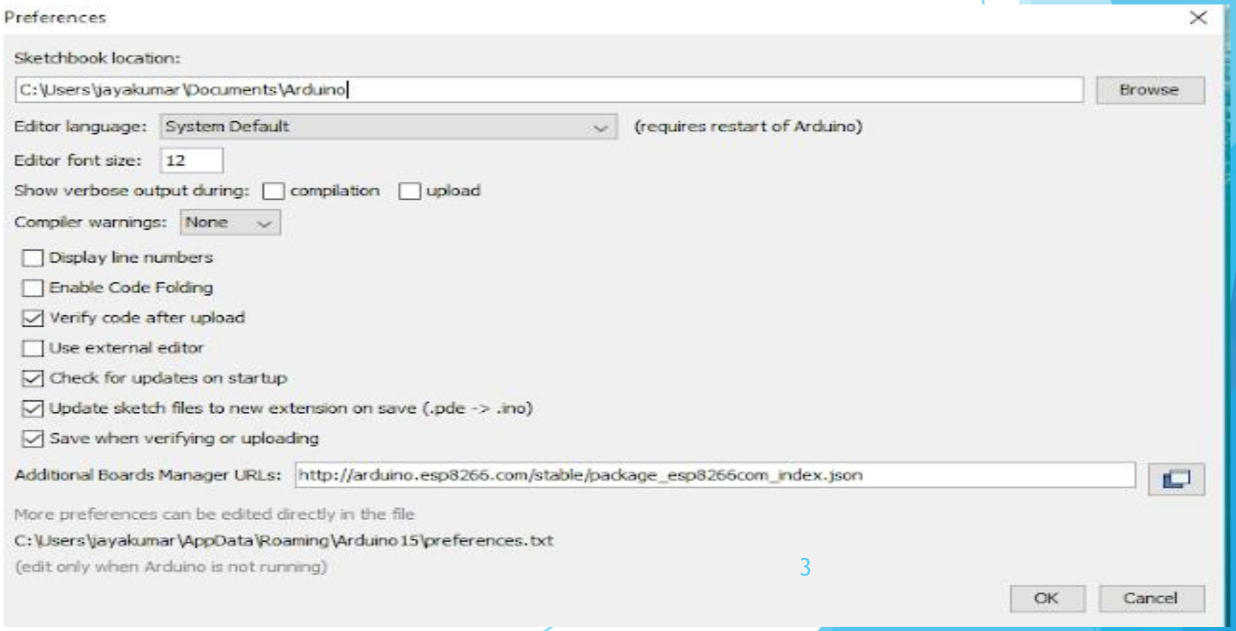
Firstly open the Arduino IDE

Go to files and click on the preference in the Arduino IDE

copy the below code in the Additional boards Manager

http://arduino.esp8266.com/stable/package_esp8266com_index.json

click OK to close the preference Tab.



Preferences

Sketchbook location:

C:\Users\jayakumar\Documents\Arduino

Editor language: System Default (requires restart of Arduino)

Editor font size: 12

Show verbose output during: □ compilation □ upload

Compiler warnings: None

□ Display line numbers

□ Enable Code Folding

☑ Verify code after upload

□ Use external editor

☑ Check for updates on startup

☑ Update sketch files to new extension on save (.pde -> .ino)

☑ Save when verifying or uploading

Additional Boards Manager URLs: http://arduino.esp8266.com/stable/package_esp8266com_index.json

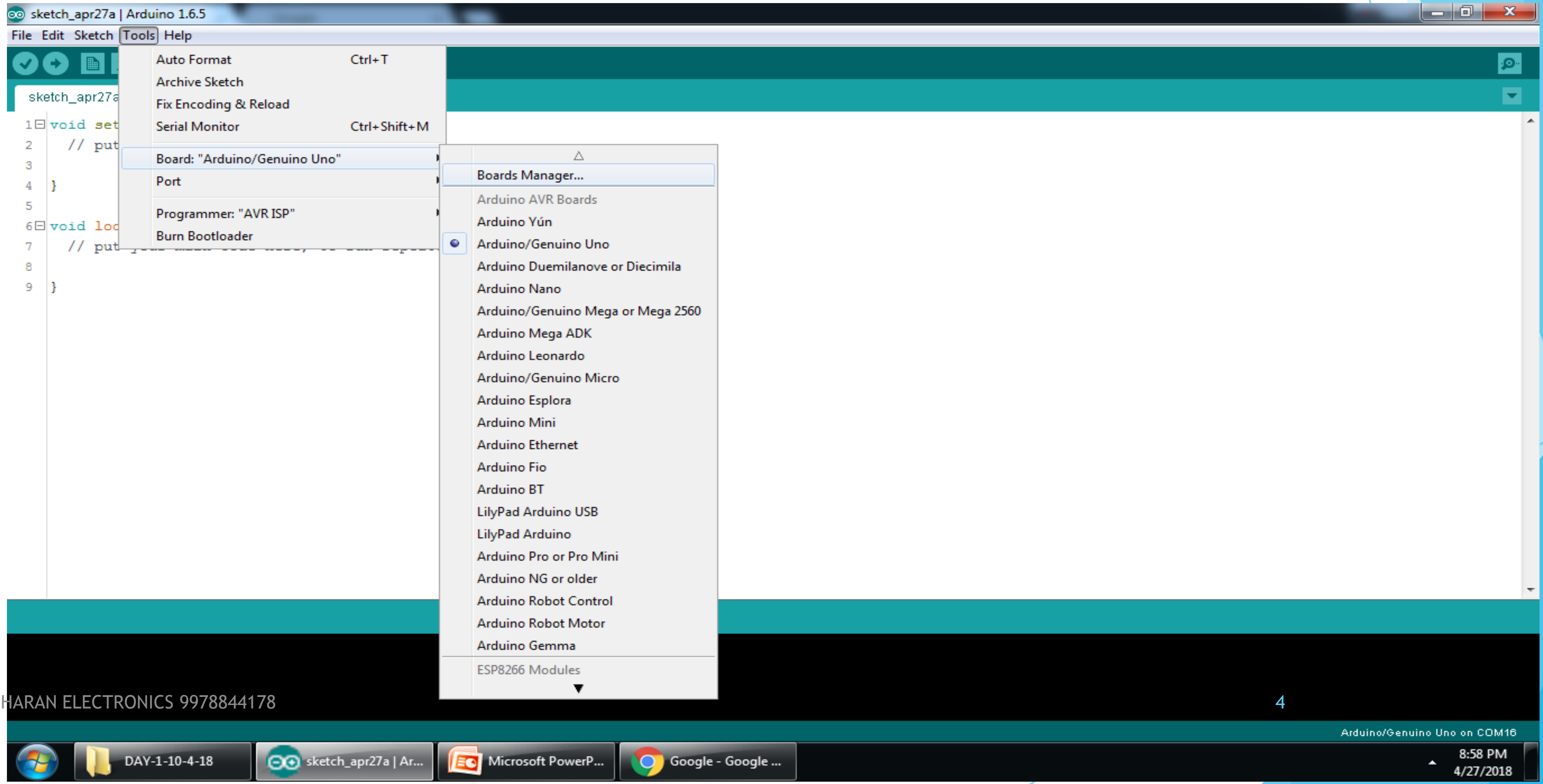
More preferences can be edited directly in the file

C:\Users\jayakumar\AppData\Roaming\Arduino15\preferences.txt

(edit only when Arduino is not running)

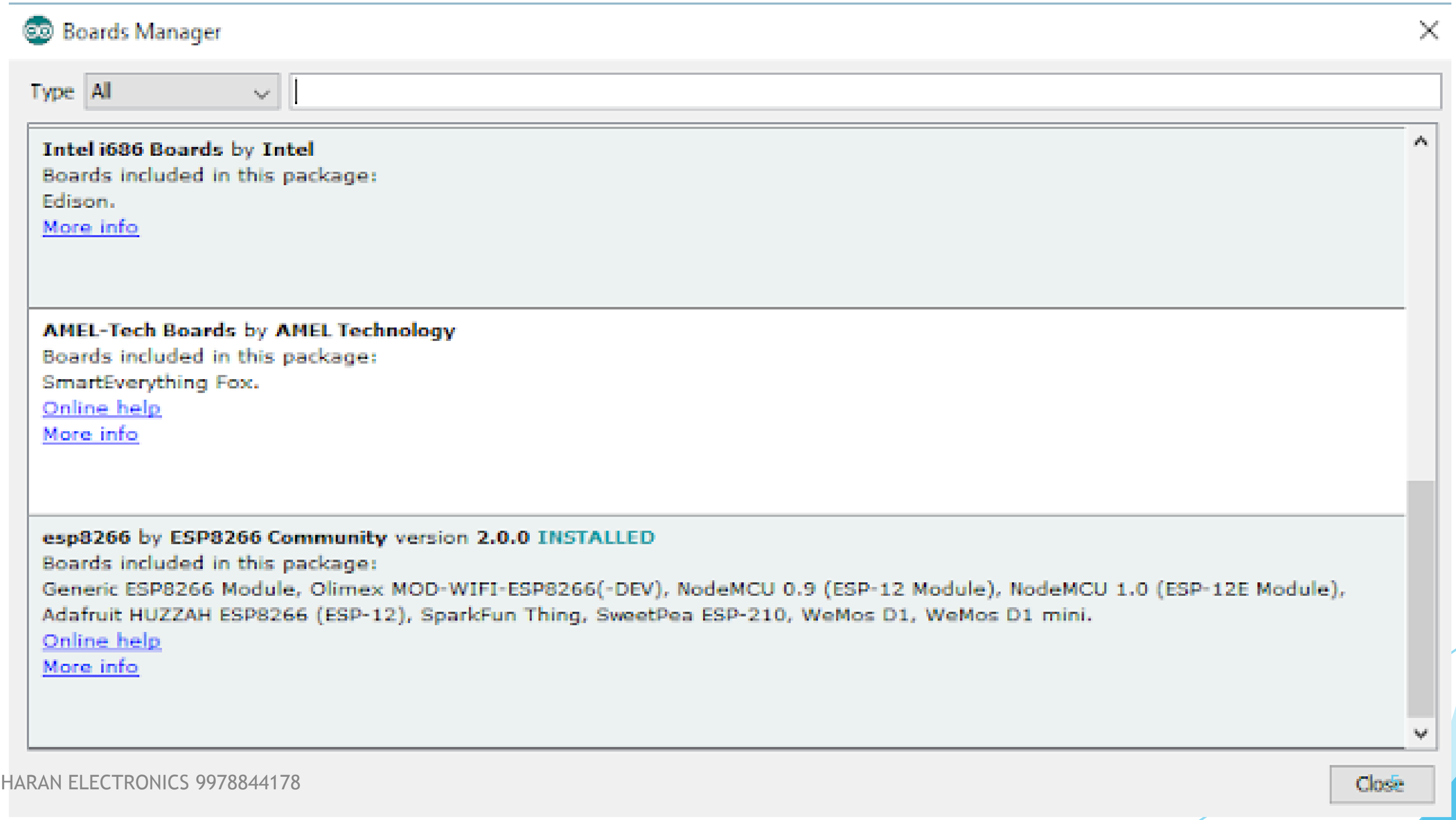
DOWNLOAD NODEMCU BOARD FORM BOARD MANAGER -STEP-2

go to Tools and board, and then select board Manager



DOWNLOAD NODEMCU BOARD FORM BOARD MANAGER -STEP-3

Navigate to esp8266 by esp8266 community and install the software for Arduino.



Intel i686 Boards by **Intel**
Boards included in this package:
Edison.
More info

AMEL-Tech Boards by **AMEL Technology**
Boards included in this package:
SmartEverything Fox.
Online help
More info

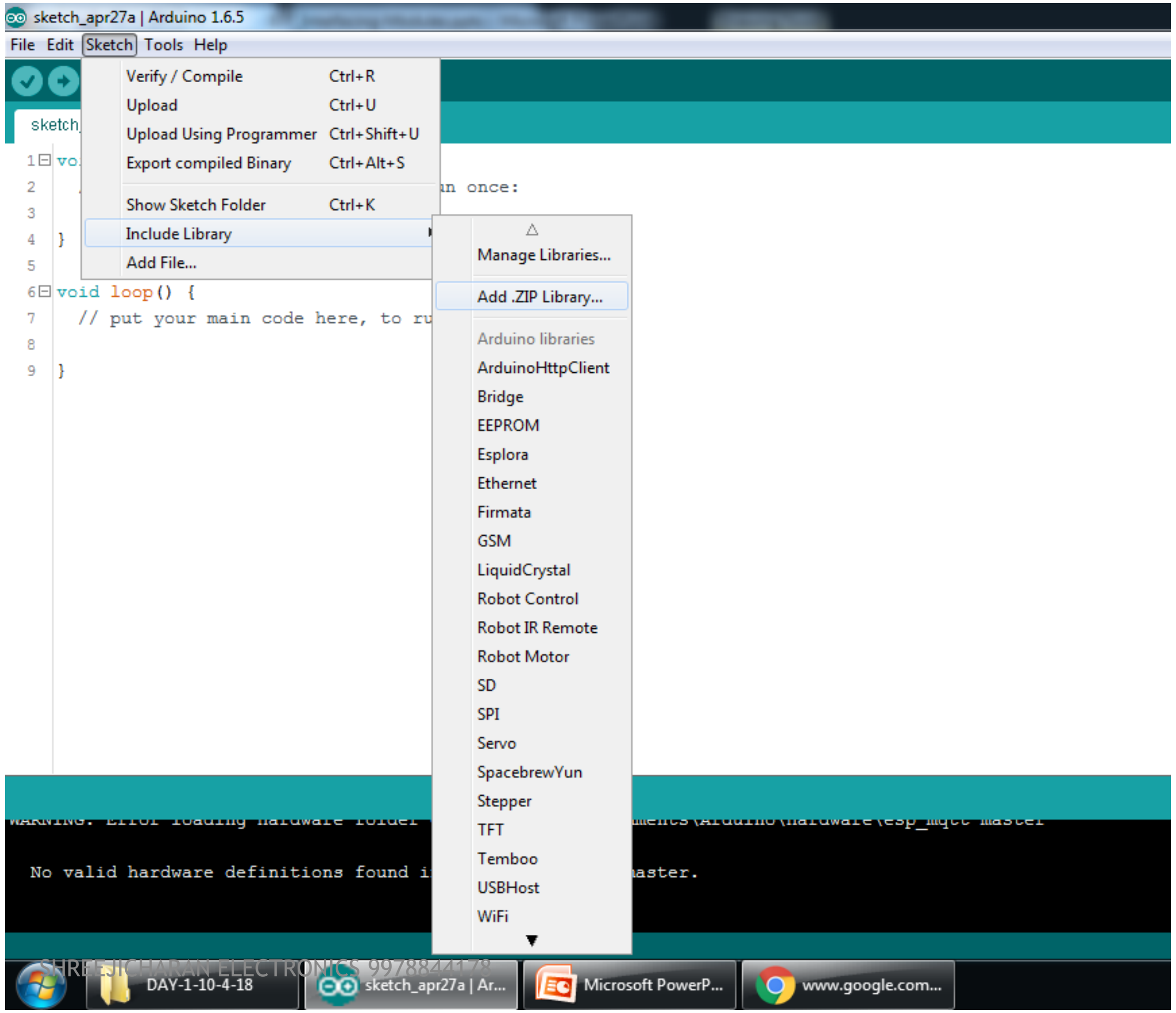
esp8266 by **ESP8266 Community** version **2.0.0** **INSTALLED**
Boards included in this package:
Generic ESP8266 Module, Olimex MOD-WIFI-ESP8266(-DEV), NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Adafruit HUZZAH ESP8266 (ESP-12), SparkFun Thing, SweetPea ESP-210, WeMos D1, WeMos D1 mini.
Online help
More info

How to add ESP8266WiFi library in Arduino IDE

- **#include <ESP8266WiFi.h>**
- Download from

  https://github.com/esp8266/Arduino/tree/master/libraries/ESP8266WiFi

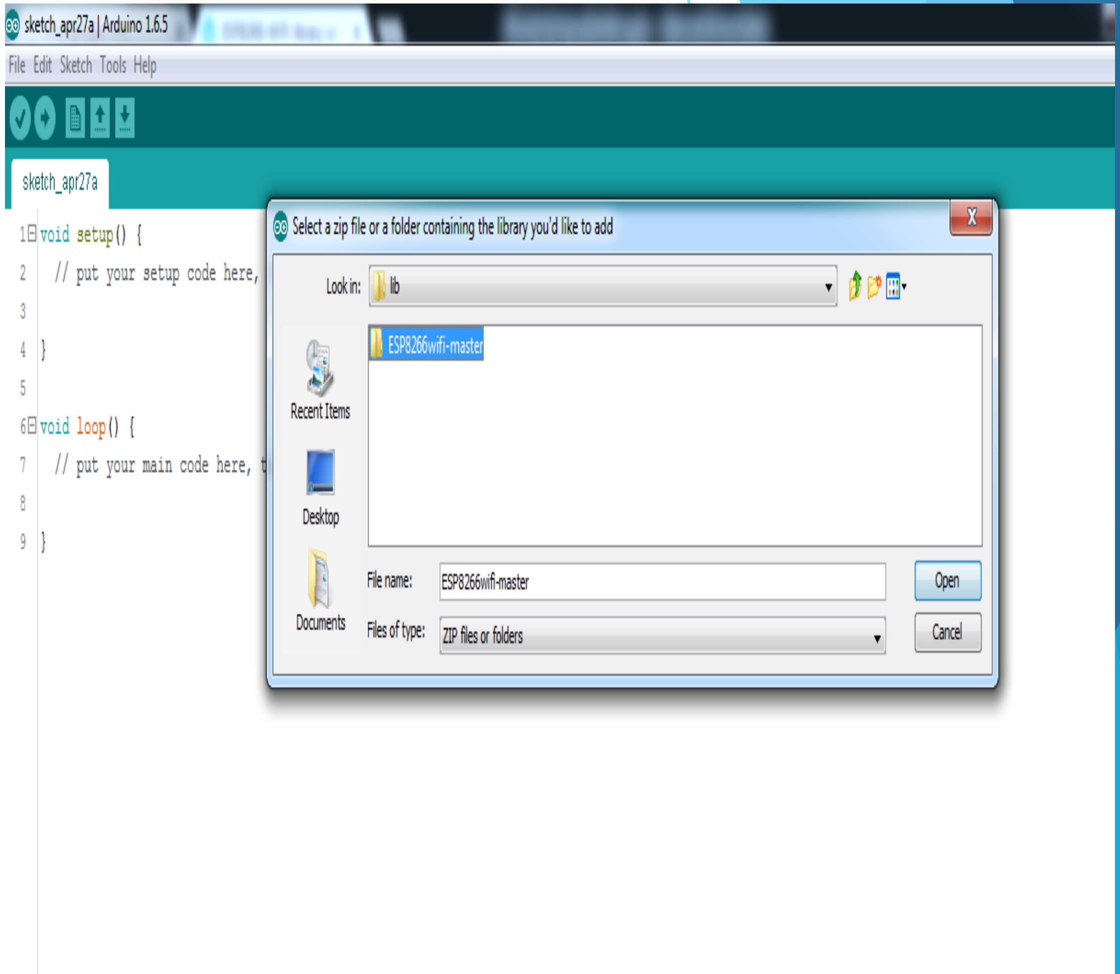
How to add ESP8266WiFi library in Arduino IDE



```
vo
                    n once:

}

void loop() {
  // put your main code here, to ru

}
```

```
WARNING: Error loading hardware folder                    ments\Arduino\hardware\esp_mqtt-master

  No valid hardware definitions found i                    aster.
```

SHREEJI CHARAN ELECTRONICS 9978844178


sketch_apr27a | Arduino 1.6.5
File Edit Sketch Tools Help
sketch_apr27a
Select a zip file or a folder containing the library you'd like to add
ESP8266wifi-master
Recent Items
Desktop
Documents
Open
Cancel

```
void setup() {
  // put your setup code here,

}

void loop() {
  // put your main code here, t

}
```

Look in: lib

File name: ESP8266wifi-master

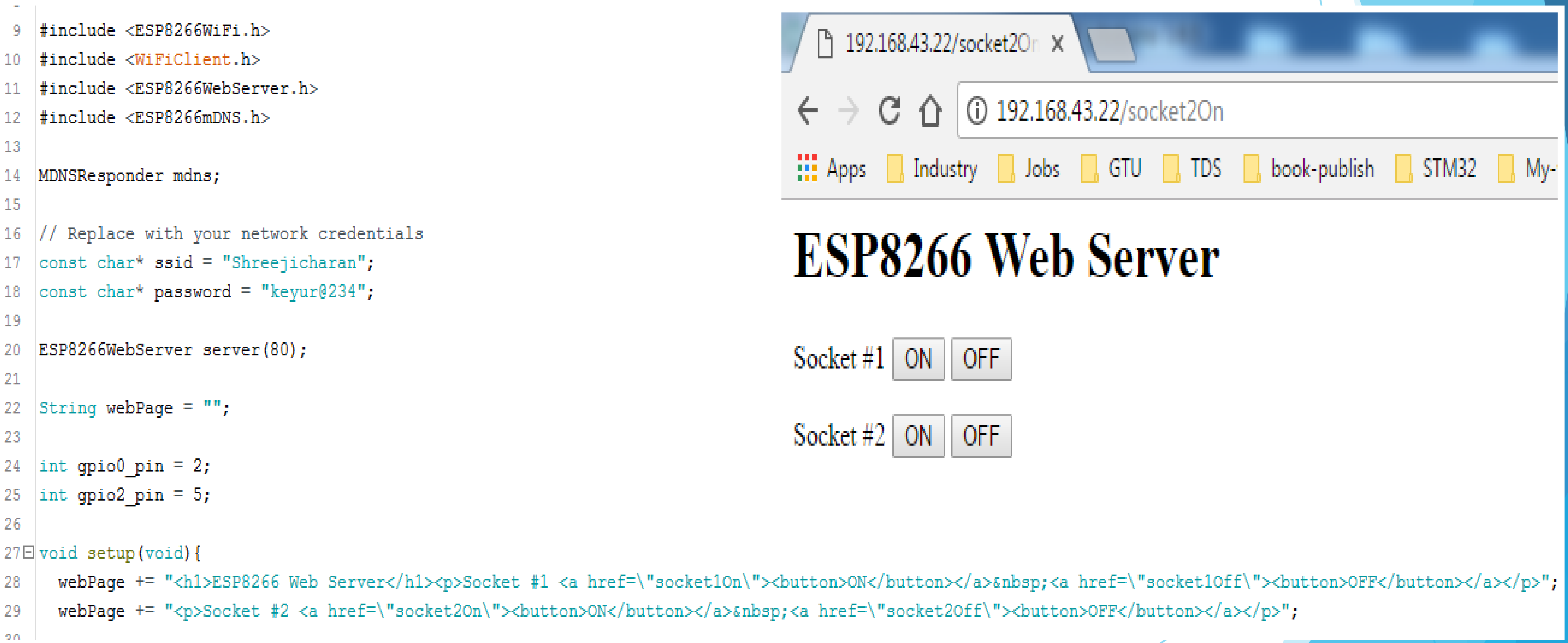
Files of type: ZIP files or folders

OUTLINE

- **INTRODUCTION – NODEMCU – IOT ESP01/ESP8266**
  - **WEBSERVER**
    1. **ARDUINO UNO IDE+ HTML WITH IP ADDRESS**
    2. **XAMPP – LOCAL HOST – LED**
    3. **XAMPP – LOCAL HOST – SENSOR**
  - **WIFI ANDRIOD APPLICATION - LED**
  - **THINGSPEAK WEBSITE SENSOR DATA PLOT**
  - **ONLINE**
    1. **WEBSERVER – LED**
    2. **WEBSERVER – SENSOR**
  - **MQTT PROTOCOL - LED**
  - **BLYNK**

WEBSERVER -1

ARDUINO UNO IDE+ HTML WITH IP ADDRESS



```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <ESP8266mDNS.h>

MDNSResponder mdns;

// Replace with your network credentials
const char* ssid = "Shreejicharan";
const char* password = "keyur@234";

ESP8266WebServer server(80);

String webPage = "";

int gpio0_pin = 2;
int gpio2_pin = 5;

void setup(void){
  webPage += "<h1>ESP8266 Web Server</h1><p>Socket #1 <a href=\"socket1On\"><button>ON</button></a>&nbsp;<a href=\"socket1Off\"><button>OFF</button></a></p>";
  webPage += "<p>Socket #2 <a href=\"socket2On\"><button>ON</button></a>&nbsp;<a href=\"socket2Off\"><button>OFF</button></a></p>";
```

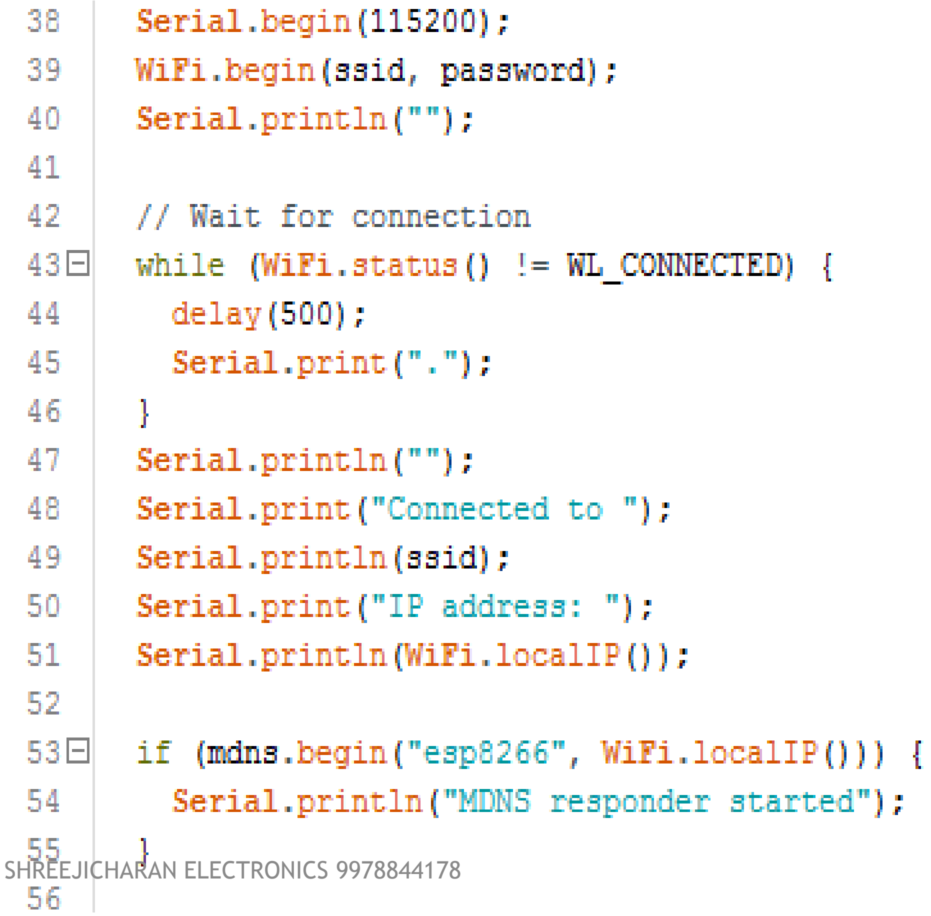
ESP8266 Web Server

Socket #1 ON OFF

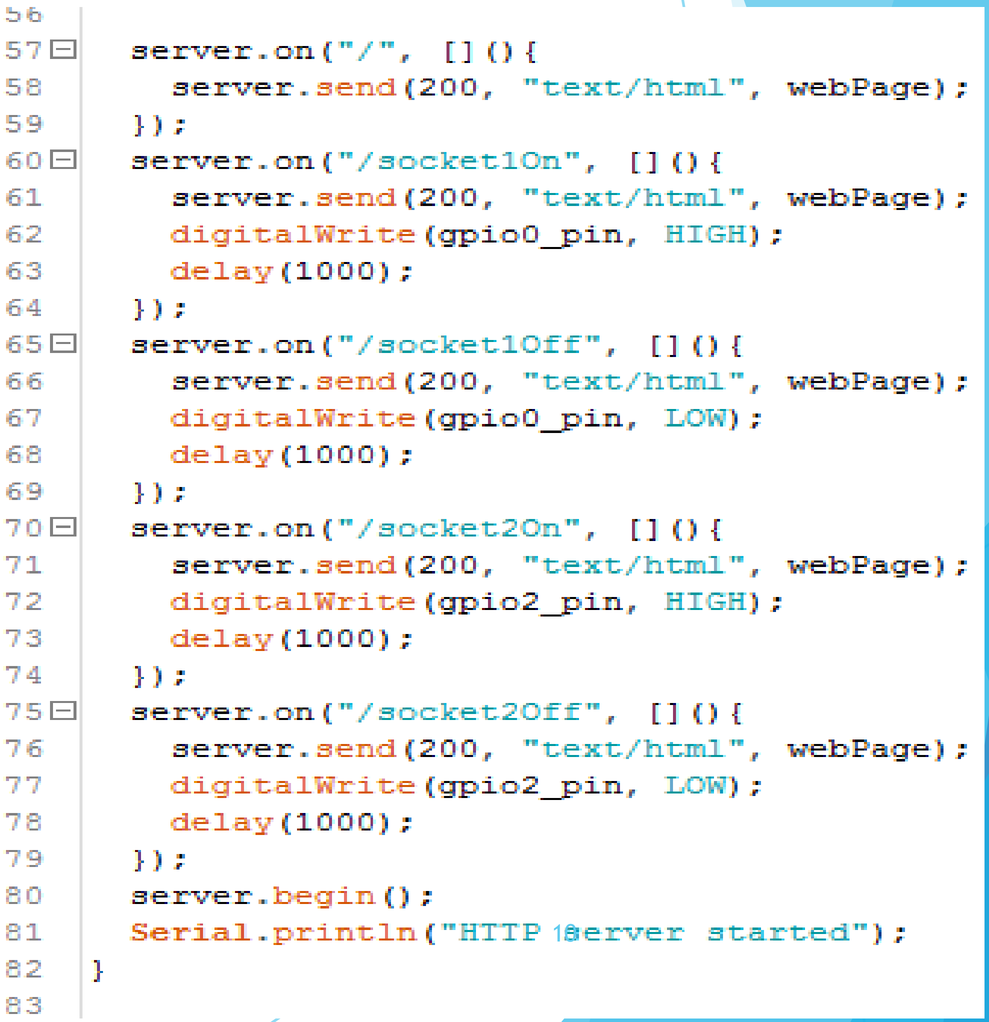
Socket #2 ON OFF

WEBSERVER -1

ARDUINO UNO IDE+ HTML WITH IP ADDRESS



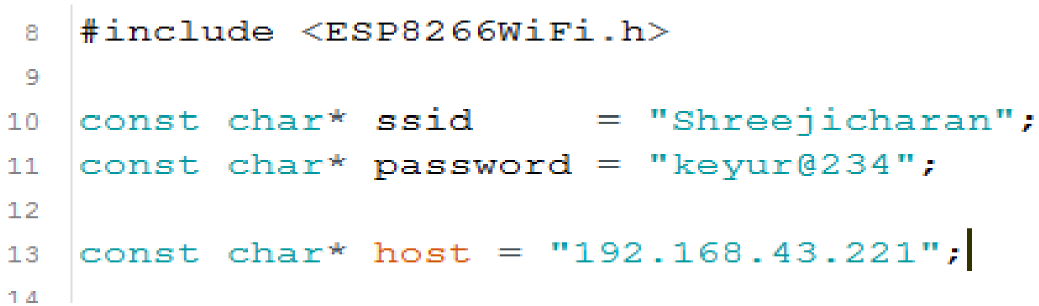
```
Serial.begin(115200);
WiFi.begin(ssid, password);
Serial.println("");

// Wait for connection
while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
}
Serial.println("");
Serial.print("Connected to ");
Serial.println(ssid);
Serial.print("IP address: ");
Serial.println(WiFi.localIP());

if (mdns.begin("esp8266", WiFi.localIP())) {
  Serial.println("MDNS responder started");
}

```



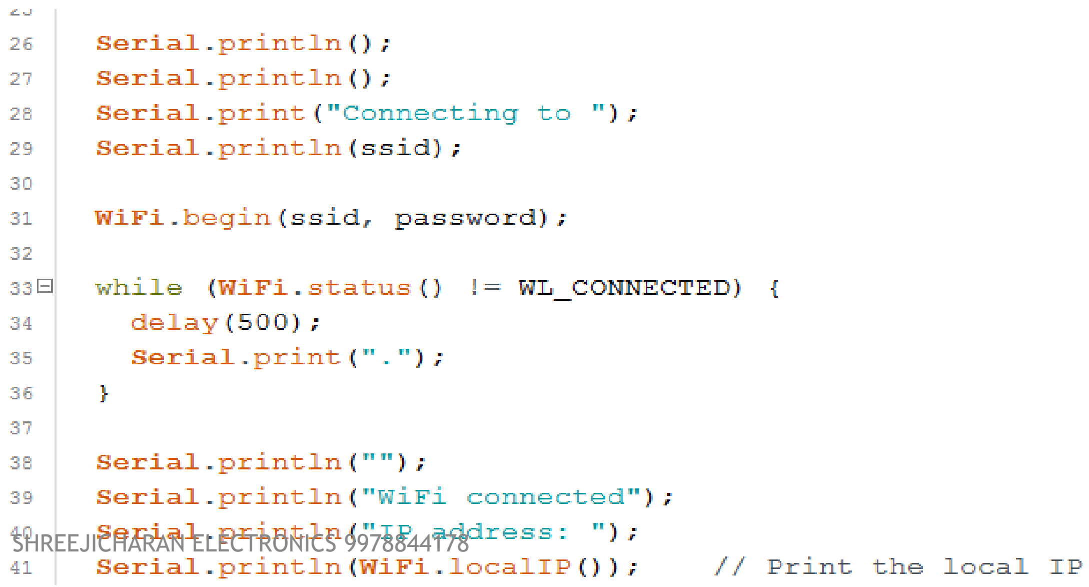
```

  server.on("/", [](){
    server.send(200, "text/html", webPage);
  });
  server.on("/socket1On", [](){
    server.send(200, "text/html", webPage);
    digitalWrite(gpio0_pin, HIGH);
    delay(1000);
  });
  server.on("/socket1Off", [](){
    server.send(200, "text/html", webPage);
    digitalWrite(gpio0_pin, LOW);
    delay(1000);
  });
  server.on("/socket2On", [](){
    server.send(200, "text/html", webPage);
    digitalWrite(gpio2_pin, HIGH);
    delay(1000);
  });
  server.on("/socket2Off", [](){
    server.send(200, "text/html", webPage);
    digitalWrite(gpio2_pin, LOW);
    delay(1000);
  });
  server.begin();
  Serial.println("HTTP server started");
}

```

WEBSERVER -2

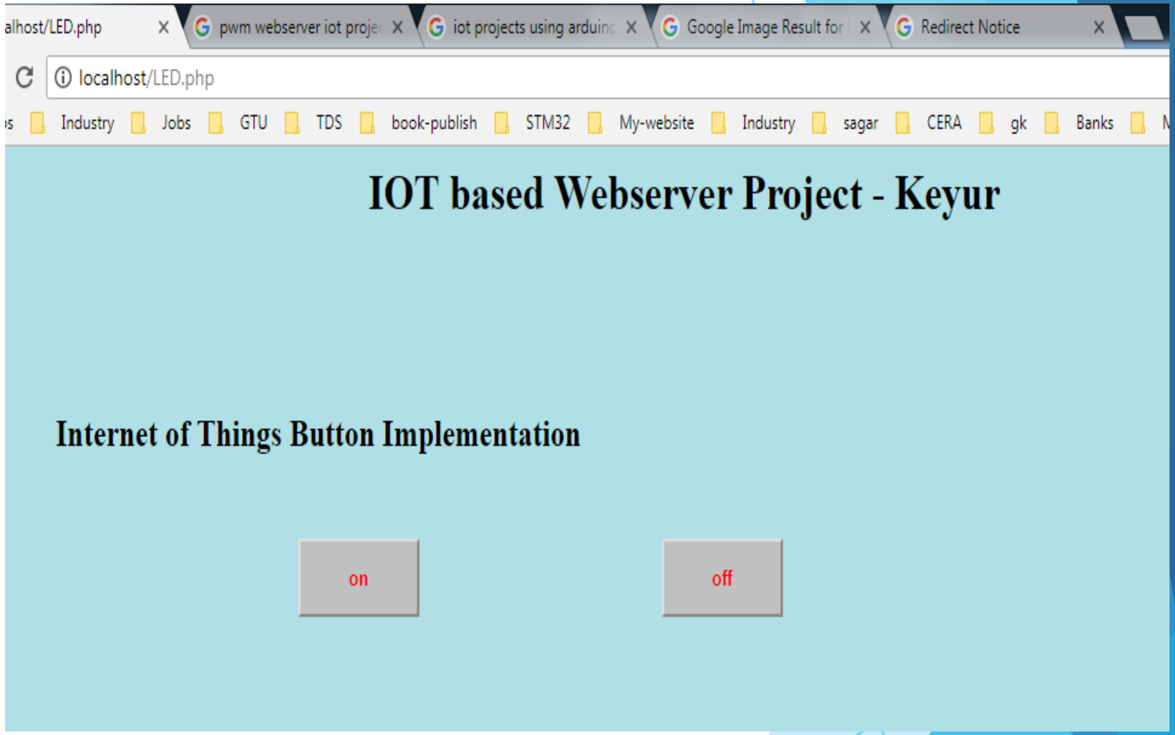
XAMPP – LOCAL HOST – LED



```
#include <ESP8266WiFi.h>

const char* ssid     = "Shreejicharan";
const char* password = "keyur@234";

const char* host = "192.168.43.221";
```



```
 Serial.println();
 Serial.println();
 Serial.print("Connecting to ");
 Serial.println(ssid);

 WiFi.begin(ssid, password);

 while (WiFi.status() != WL_CONNECTED) {
   delay(500);
   Serial.print(".");
 }

 Serial.println("");
 Serial.println("WiFi connected");
 Serial.println("IP address: ");
 Serial.println(WiFi.localIP());    // Print the local IP
```



IOT based Webserver Project - Keyur

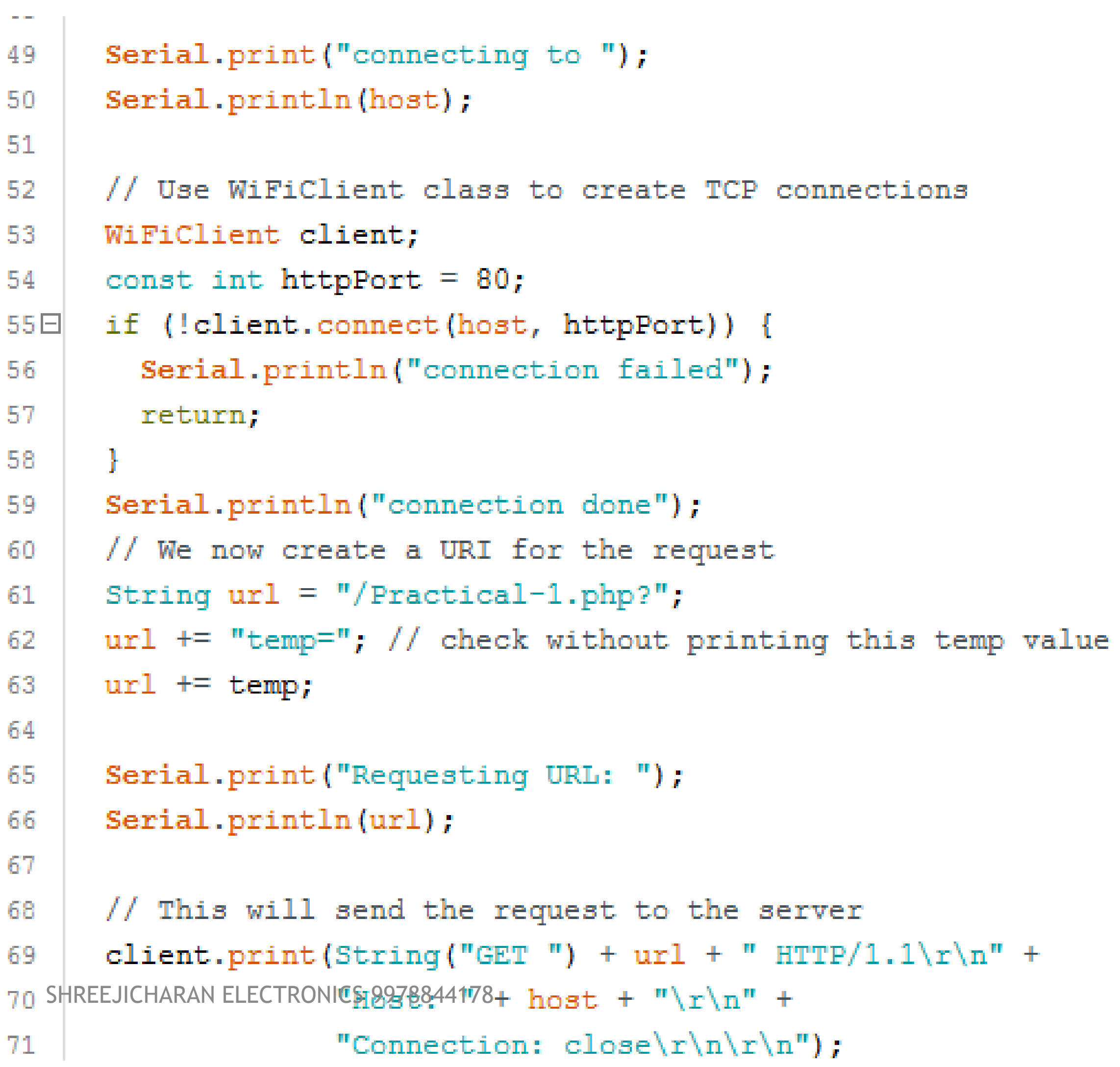
Internet of Things Button Implementation

on

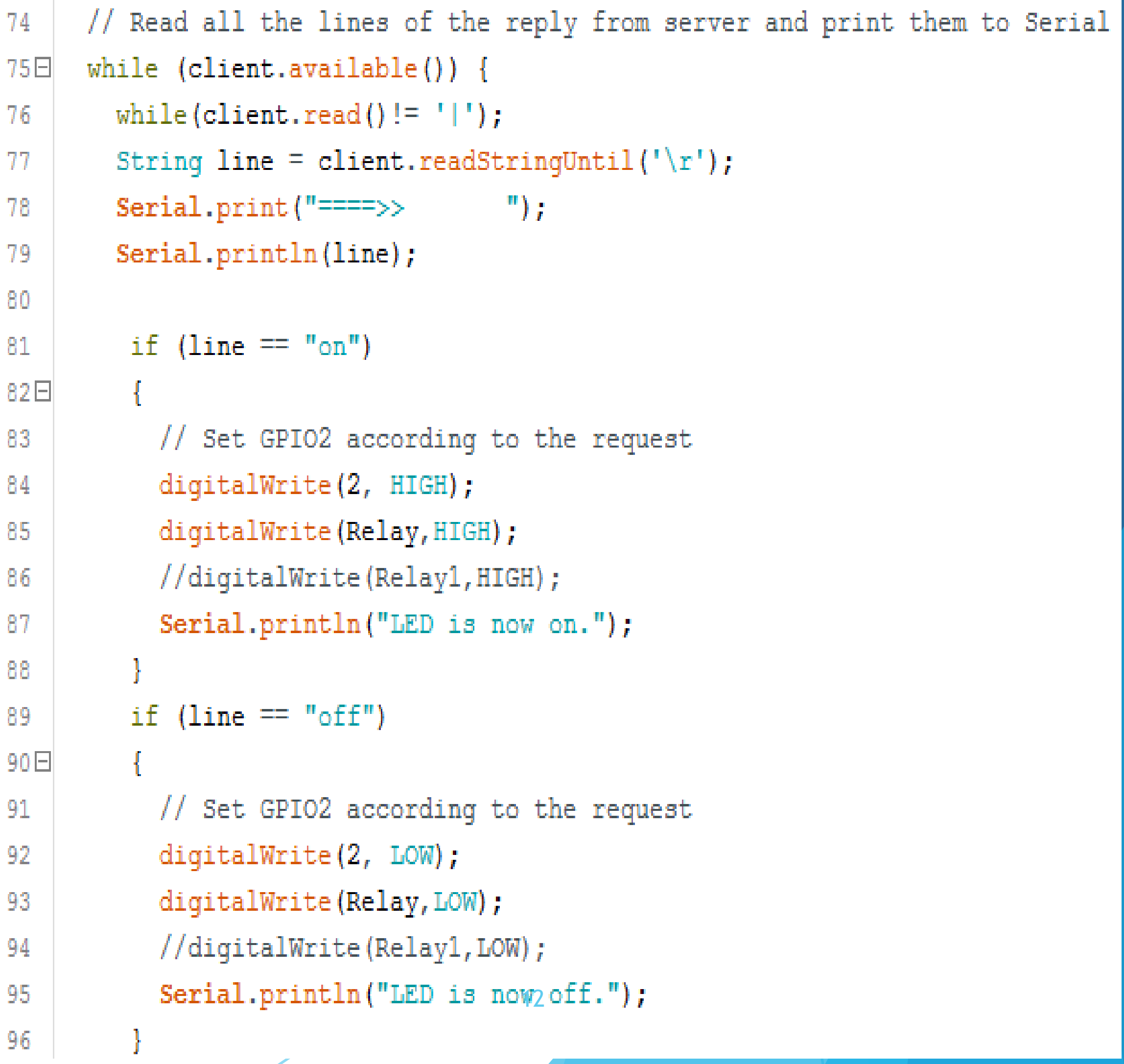
off

WEBSERVER -2

XAMPP – LOCAL HOST – LED



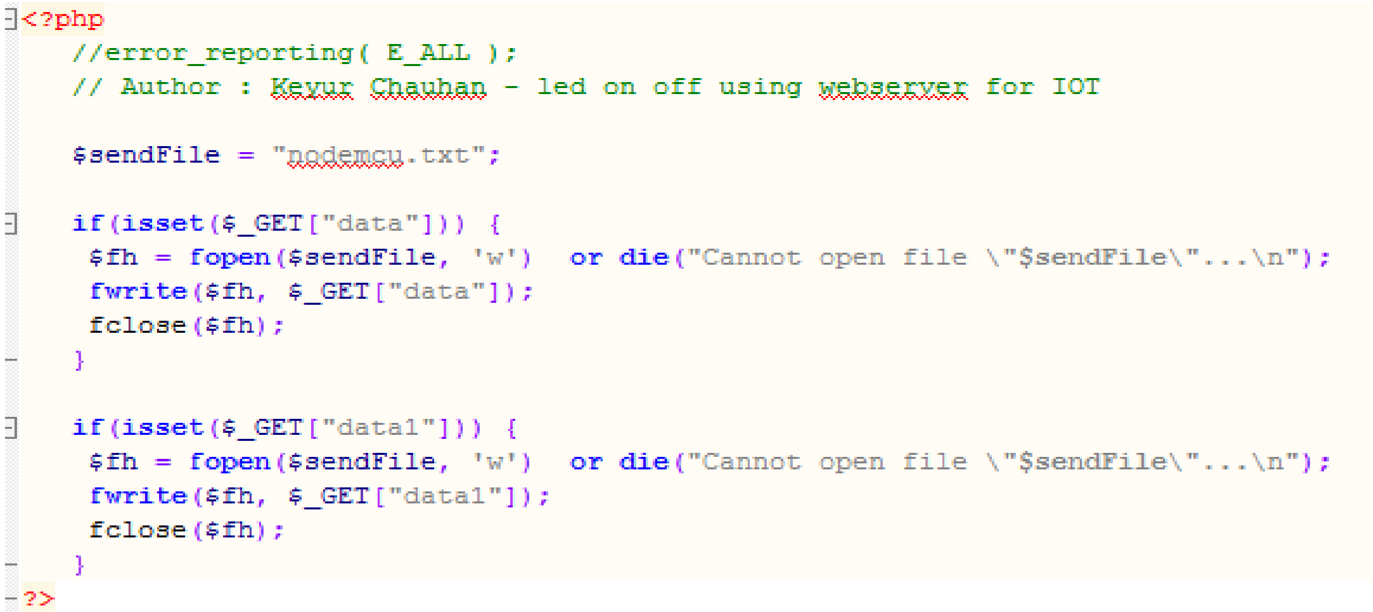
```
Serial.print("connecting to ");
Serial.println(host);

// Use WiFiClient class to create TCP connections
WiFiClient client;
const int httpPort = 80;
if (!client.connect(host, httpPort)) {
  Serial.println("connection failed");
  return;
}
Serial.println("connection done");
// We now create a URI for the request
String url = "/Practical-1.php?";
url += "temp="; // check without printing this temp value
url += temp;

Serial.print("Requesting URL: ");
Serial.println(url);

// This will send the request to the server
client.print(String("GET ") + url + " HTTP/1.1\r\n" +
             "Host: " + host + "\r\n" +
             "Connection: close\r\n\r\n");
```



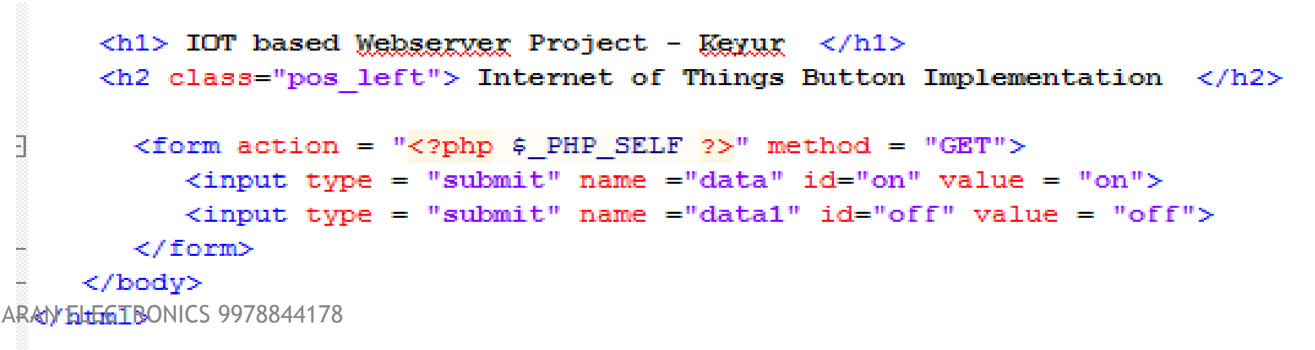
```
// Read all the lines of the reply from server and print them to Serial
while (client.available()) {
  while(client.read()!= '|');
  String line = client.readStringUntil('\r');
  Serial.print("====>>       ");
  Serial.println(line);

   if (line == "on")
   {
     // Set GPIO2 according to the request
     digitalWrite(2, HIGH);
     digitalWrite(Relay,HIGH);
     //digitalWrite(Relay1,HIGH);
     Serial.println("LED is now on.");
   }
   if (line == "off")
   {
     // Set GPIO2 according to the request
     digitalWrite(2, LOW);
     digitalWrite(Relay,LOW);
     //digitalWrite(Relay1,LOW);
     Serial.println("LED is now off.");
   }
```

WEBSERVER -2

XAMPP – LOCAL HOST – LED



```
<?php
    //error_reporting( E_ALL );
    // Author : Keyur Chauhan - led on off using webserver for IOT

    $sendFile = "nodemcu.txt";

    if(isset($_GET["data"])) {
     $fh = fopen($sendFile, 'w')  or die("Cannot open file \"$sendFile\"...\n");
     fwrite($fh, $_GET["data"]);
     fclose($fh);
    }

    if(isset($_GET["data1"])) {
     $fh = fopen($sendFile, 'w')  or die("Cannot open file \"$sendFile\"...\n");
     fwrite($fh, $_GET["data1"]);
     fclose($fh);
    }
?>
```



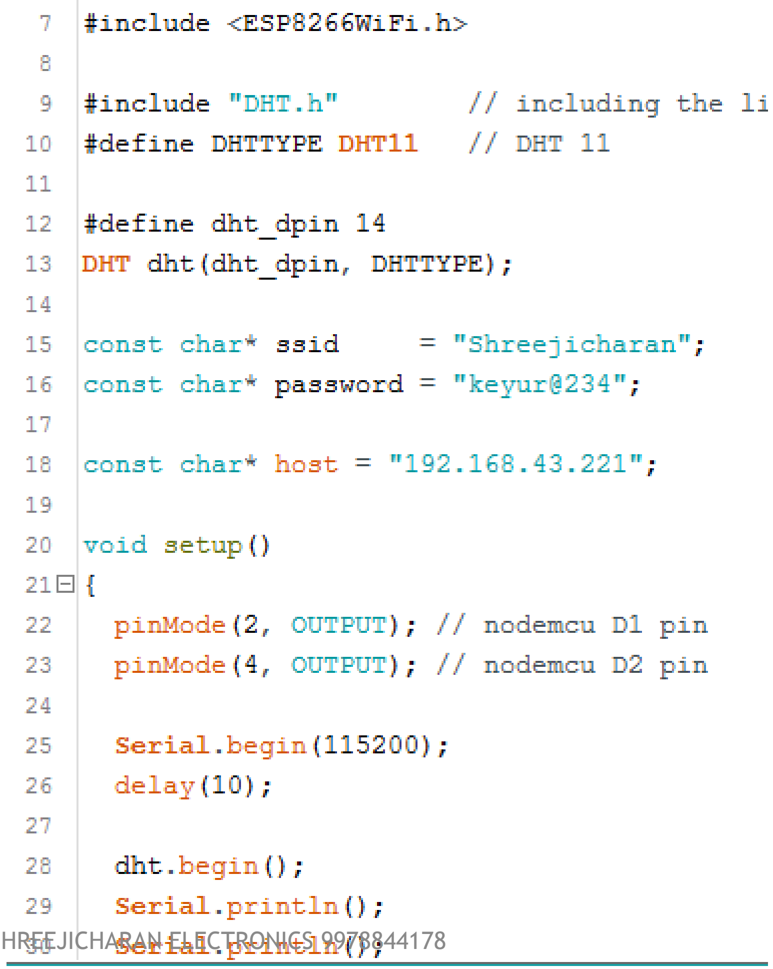
```
     <h1> IOT based Webserver Project - Keyur  </h1>
     <h2 class="pos_left"> Internet of Things Button Implementation  </h2>

       <form action = "<?php $_PHP_SELF ?>" method = "GET">
          <input type = "submit" name ="data" id="on" value = "on">
          <input type = "submit" name ="data1" id="off" value = "off">
       </form>
    </body>
</html>
```

WEBSERVER -2

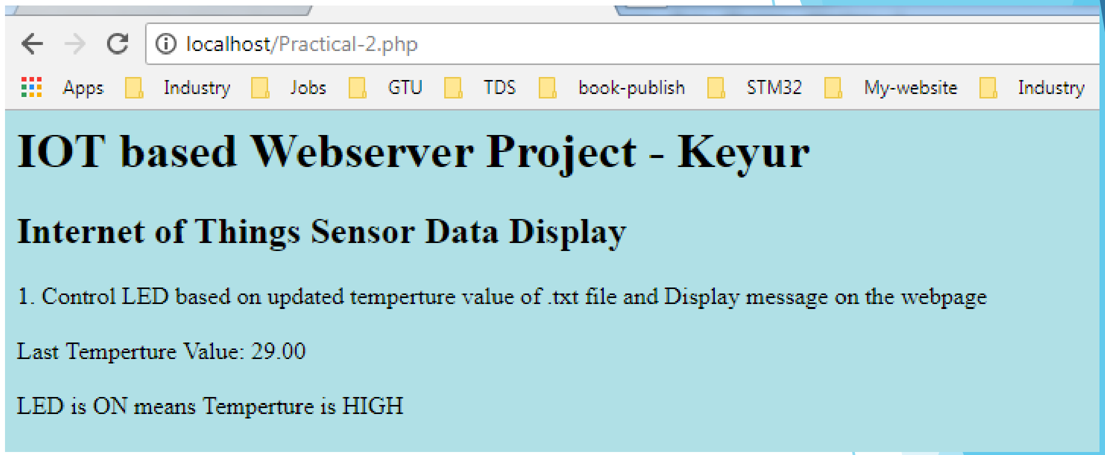
XAMPP – LOCAL HOST – LED - HARDWARE

WEBSERVER -3

XAMPP – LOCAL HOST – SENSOR



```
#include <ESP8266WiFi.h>

#include "DHT.h"        // including the li
#define DHTTYPE DHT11   // DHT 11

#define dht_dpin 14
DHT dht(dht_dpin, DHTTYPE);

const char* ssid     = "Shreejicharan";
const char* password = "keyur@234";

const char* host = "192.168.43.221";

void setup()
{
  pinMode(2, OUTPUT); // nodemcu D1 pin
  pinMode(4, OUTPUT); // nodemcu D2 pin

  Serial.begin(115200);
  delay(10);

  dht.begin();
  Serial.println();
  Serial.println();
```



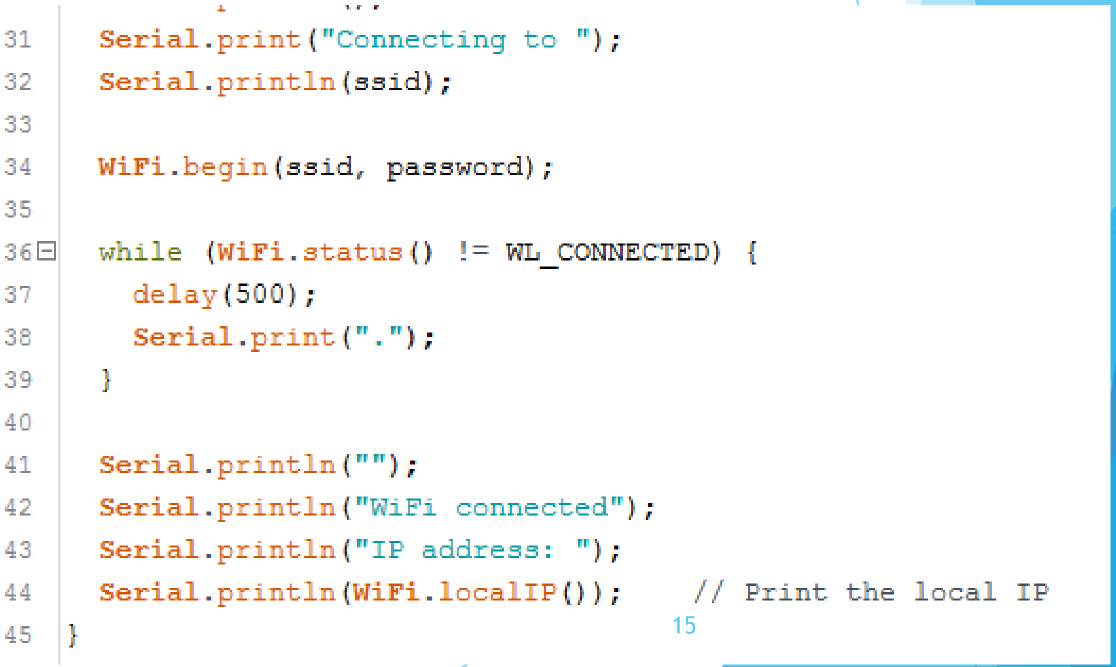
IOT based Webserver Project - Keyur

Internet of Things Sensor Data Display

1. Control LED based on updated temperture value of .txt file and Display message on the webpage

Last Temperture Value: 29.00

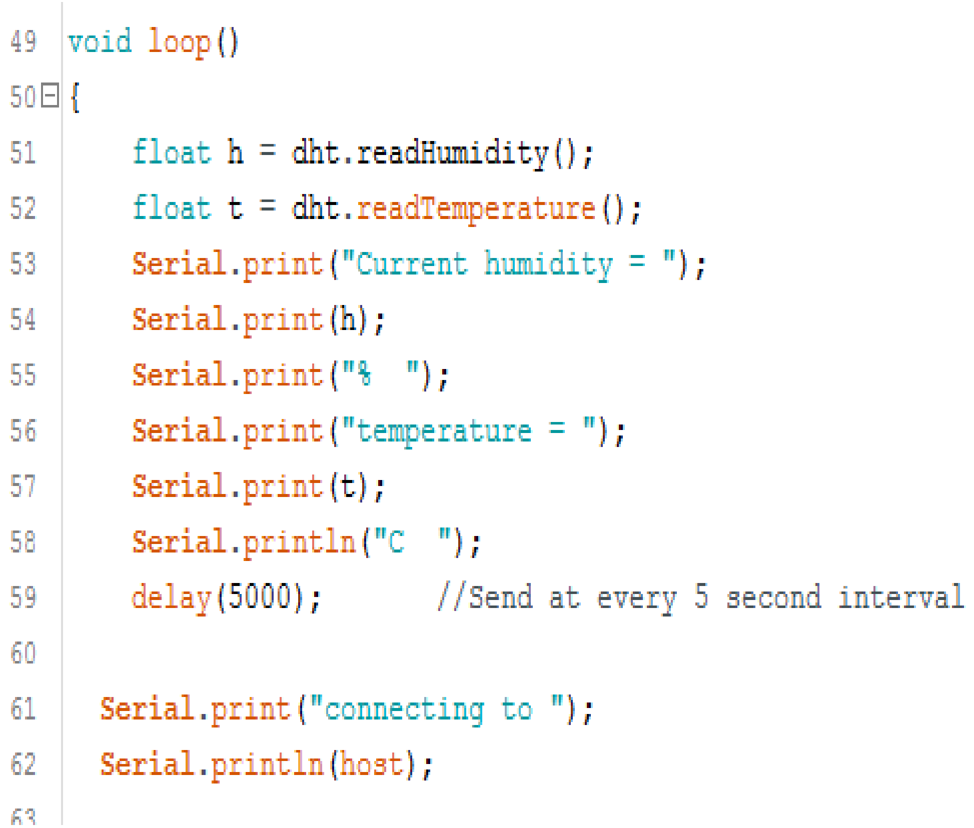
LED is ON means Temperture is HIGH



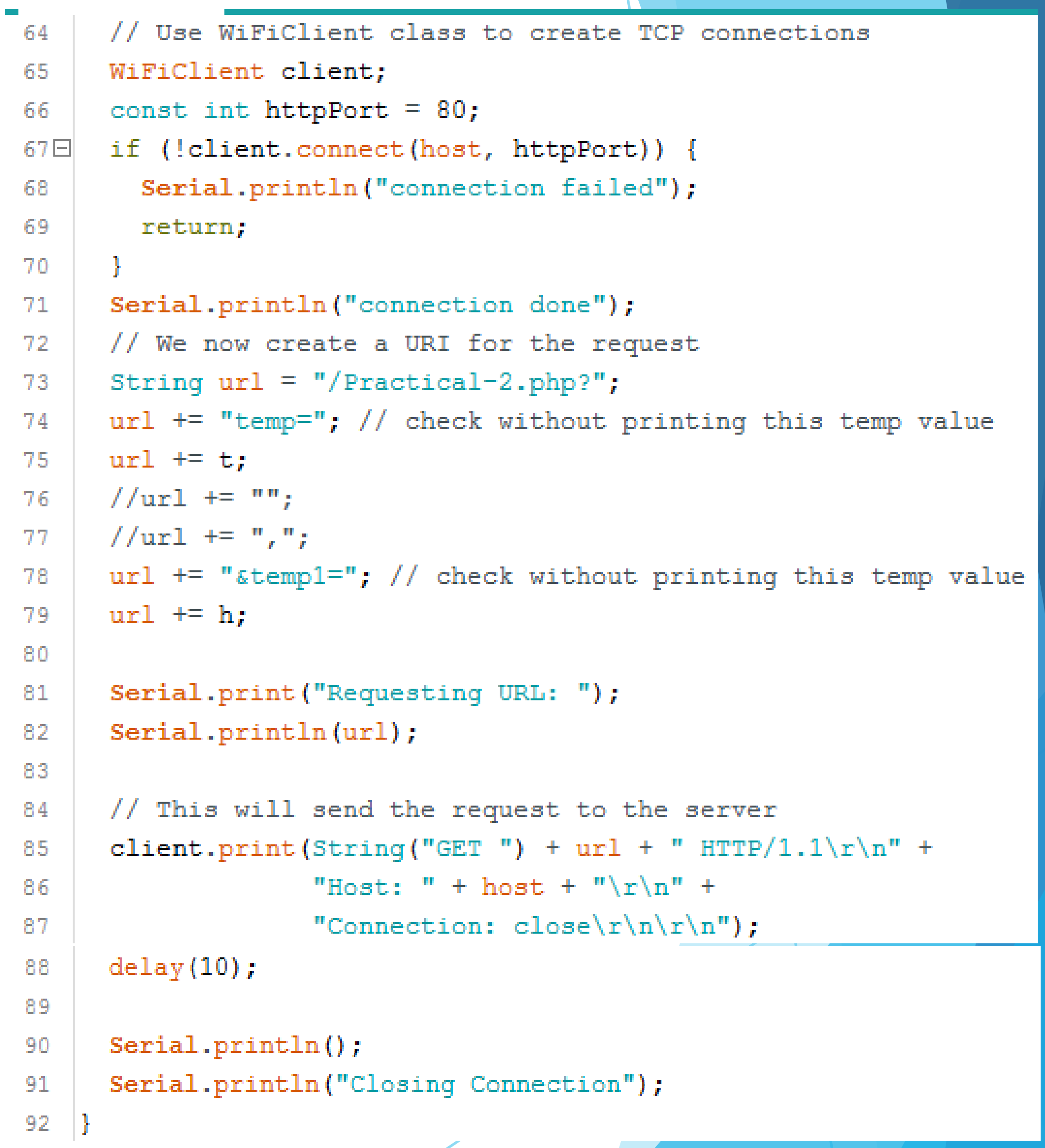
```
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());     // Print the local IP
}
```

WEBSERVER -3

XAMPP – LOCAL HOST – SENSOR



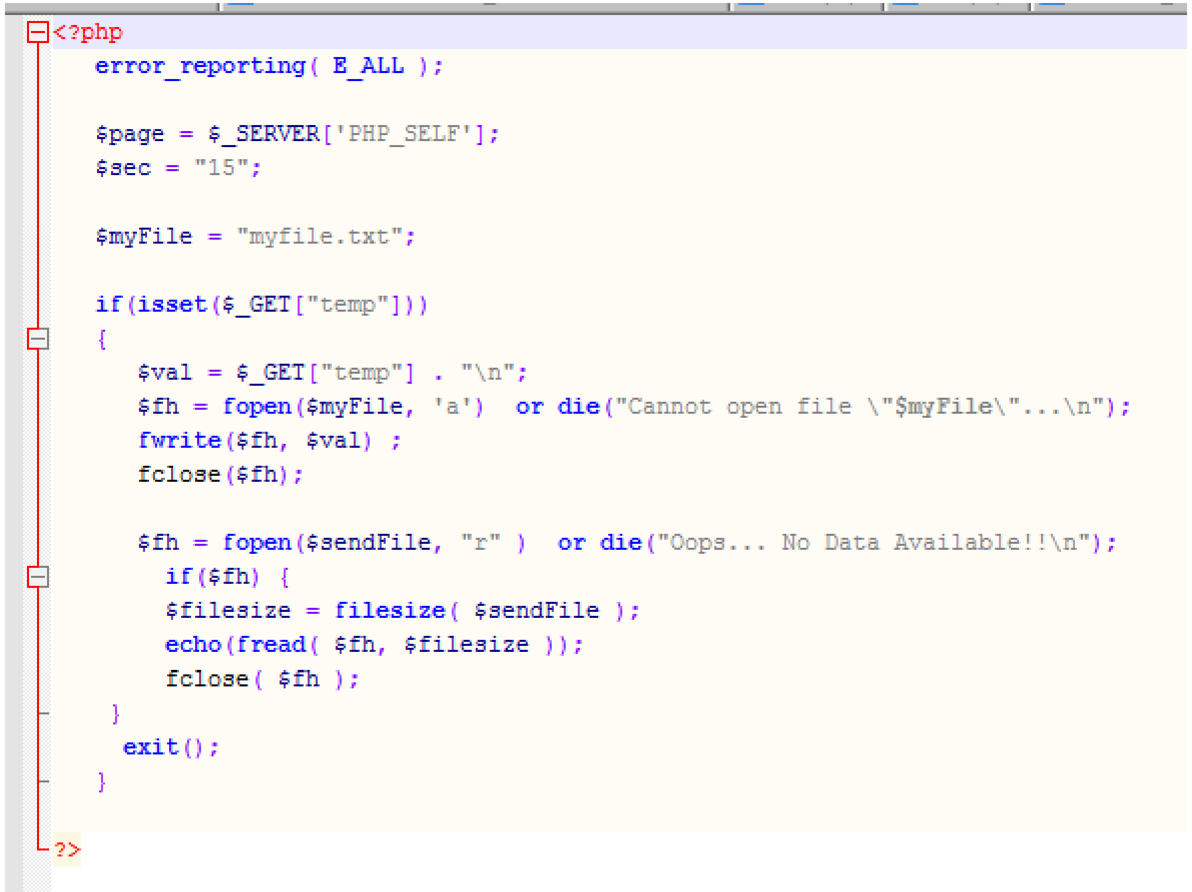
```
void loop()
{
    float h = dht.readHumidity();
    float t = dht.readTemperature();
    Serial.print("Current humidity = ");
    Serial.print(h);
    Serial.print("%  ");
    Serial.print("temperature = ");
    Serial.print(t);
    Serial.println("C  ");
    delay(5000);       //Send at every 5 second interval

  Serial.print("connecting to ");
  Serial.println(host);
```



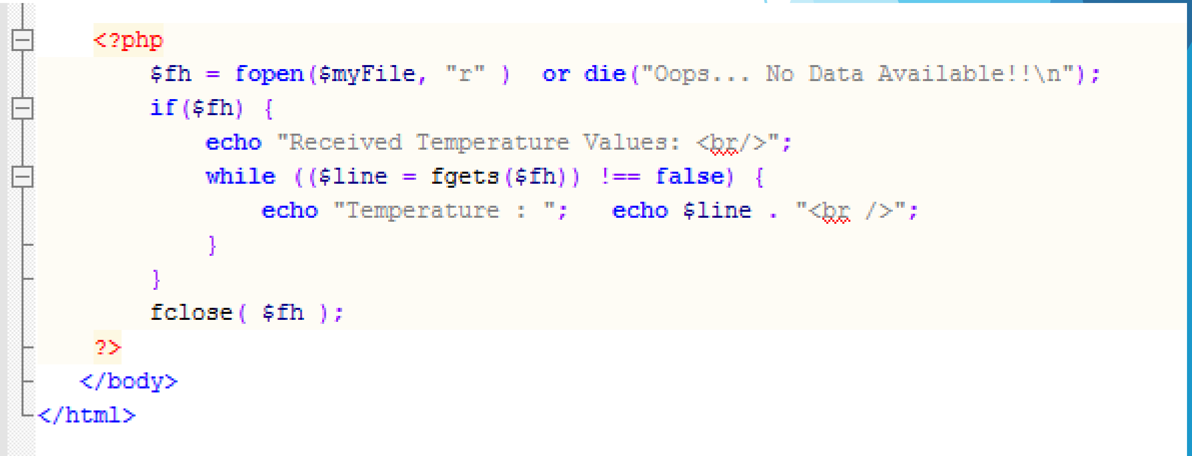
```
  // Use WiFiClient class to create TCP connections
  WiFiClient client;
  const int httpPort = 80;
  if (!client.connect(host, httpPort)) {
    Serial.println("connection failed");
    return;
  }
  Serial.println("connection done");
  // We now create a URI for the request
  String url = "/Practical-2.php?";
  url += "temp="; // check without printing this temp value
  url += t;
  //url += "";
  //url += ",";
  url += "&temp1="; // check without printing this temp value
  url += h;

  Serial.print("Requesting URL: ");
  Serial.println(url);

  // This will send the request to the server
  client.print(String("GET ") + url + " HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Connection: close\r\n\r\n");
  delay(10);

  Serial.println();
  Serial.println("Closing Connection");
}
```

WEBSERVER -3

XAMPP – LOCAL HOST – SENSOR



```
<?php
   error_reporting( E_ALL );

   $page = $_SERVER['PHP_SELF'];
   $sec = "15";

   $myFile = "myfile.txt";

   if(isset($_GET["temp"]))
   {
      $val = $_GET["temp"] . "\n";
      $fh = fopen($myFile, 'a')  or die("Cannot open file \"$myFile\"...\n");
      fwrite($fh, $val) ;
      fclose($fh);

      $fh = fopen($sendFile, "r" )  or die("Oops... No Data Available!!\n");
        if($fh) {
        $filesize = filesize( $sendFile );
        echo(fread( $fh, $filesize ));
        fclose( $fh );
    }
     exit();
   }

?>
```

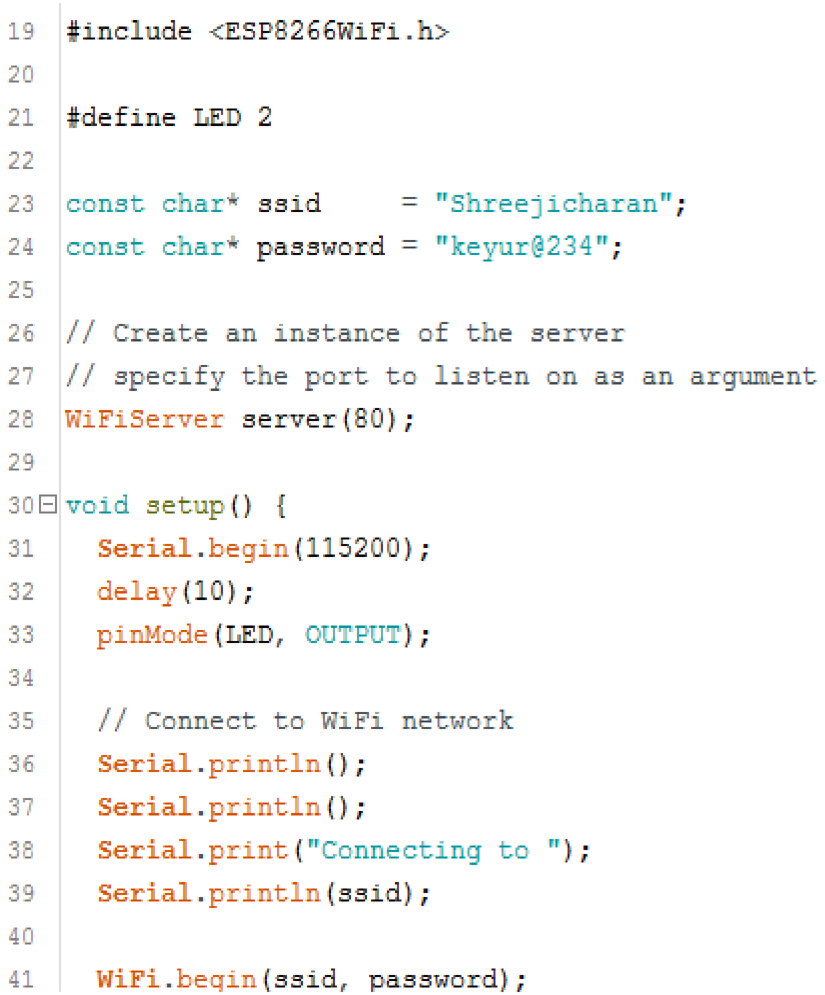


```
    <?php
      $fh = fopen($myFile, "r" )  or die("Oops... No Data Available!!\n");
      if($fh) {
          echo "Received Temperature Values: <br/>";
          while (($line = fgets($fh)) !== false) {
              echo "Temperature : ";   echo $line . "<br />";
          }
      }
      fclose( $fh );
    ?>
   </body>
</html>
```

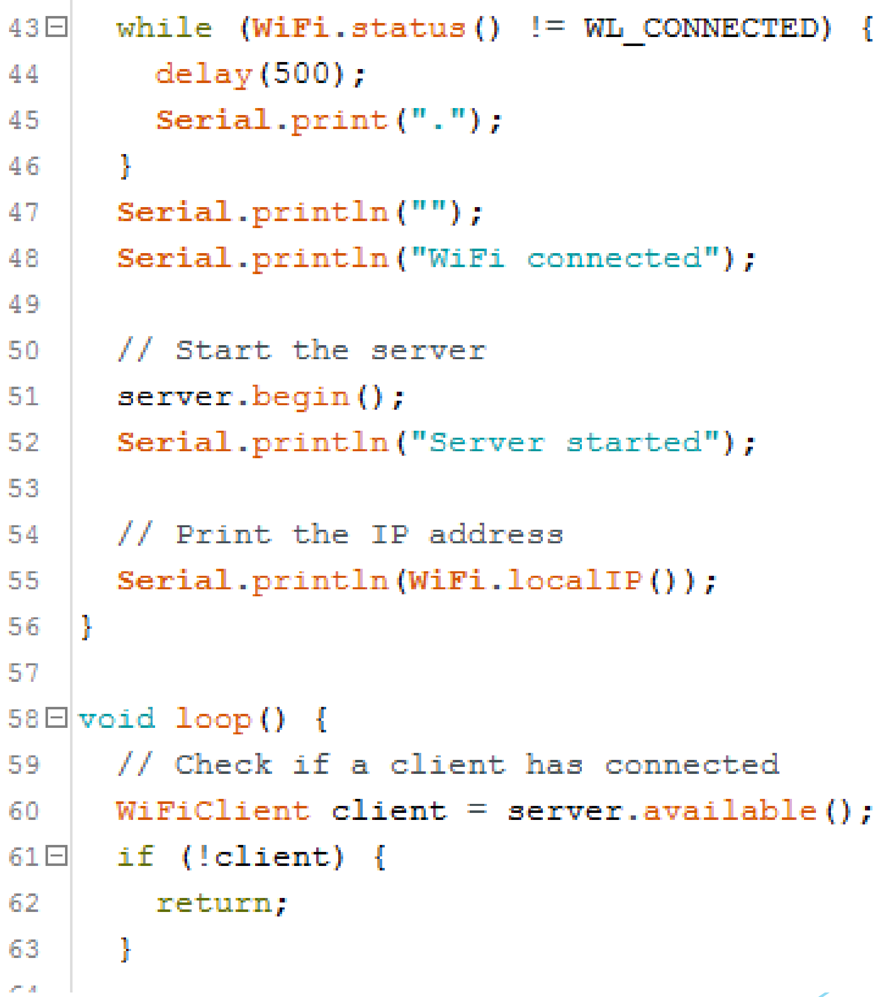
WEBSERVER -3

XAMPP – LOCAL HOST – SENSOR HARDWARE

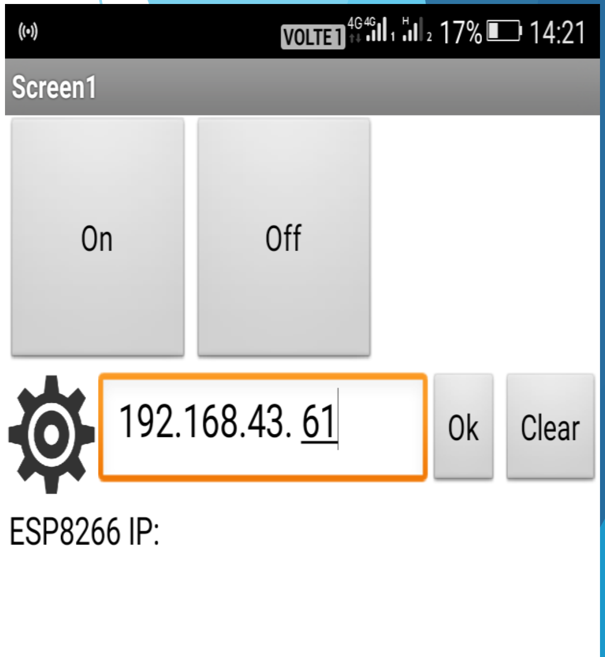
WIFI ANDRIOD APPLICATION - LED



```
#include <ESP8266WiFi.h>

#define LED 2

const char* ssid     = "Shreejicharan";
const char* password = "keyur@234";

// Create an instance of the server
// specify the port to listen on as an argument
WiFiServer server(80);

void setup() {
  Serial.begin(115200);
  delay(10);
  pinMode(LED, OUTPUT);

  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);
```



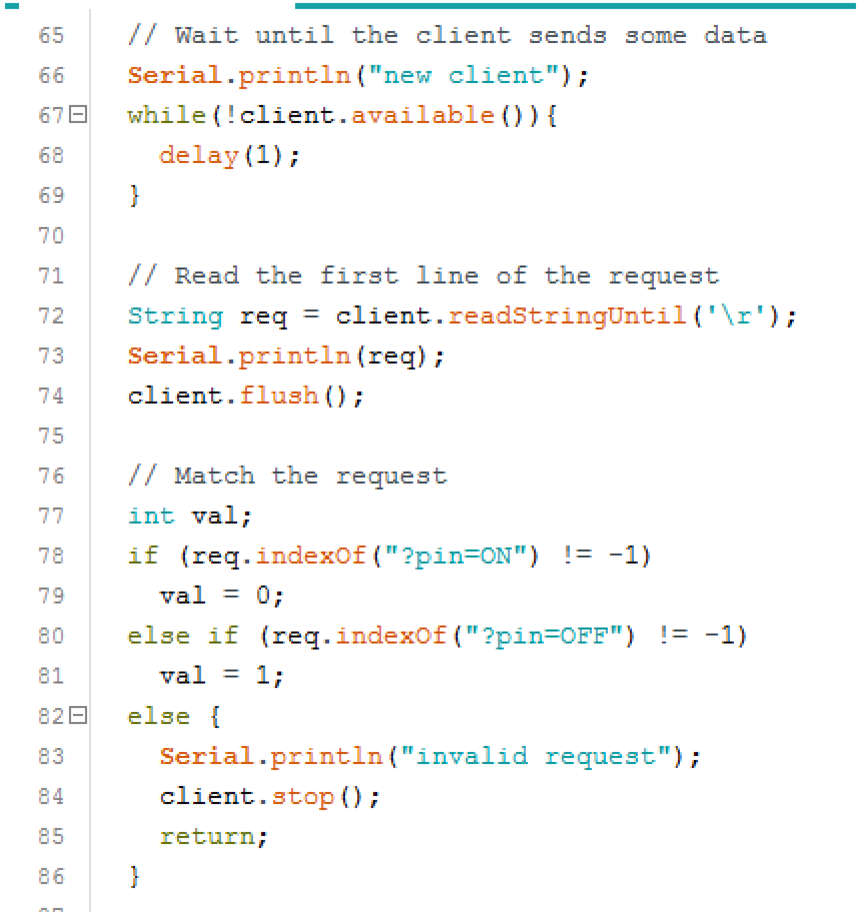
```
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  // Start the server
  server.begin();
  Serial.println("Server started");

  // Print the IP address
  Serial.println(WiFi.localIP());
}

void loop() {
  // Check if a client has connected
  WiFiClient client = server.available();
  if (!client) {
    return;
  }
```



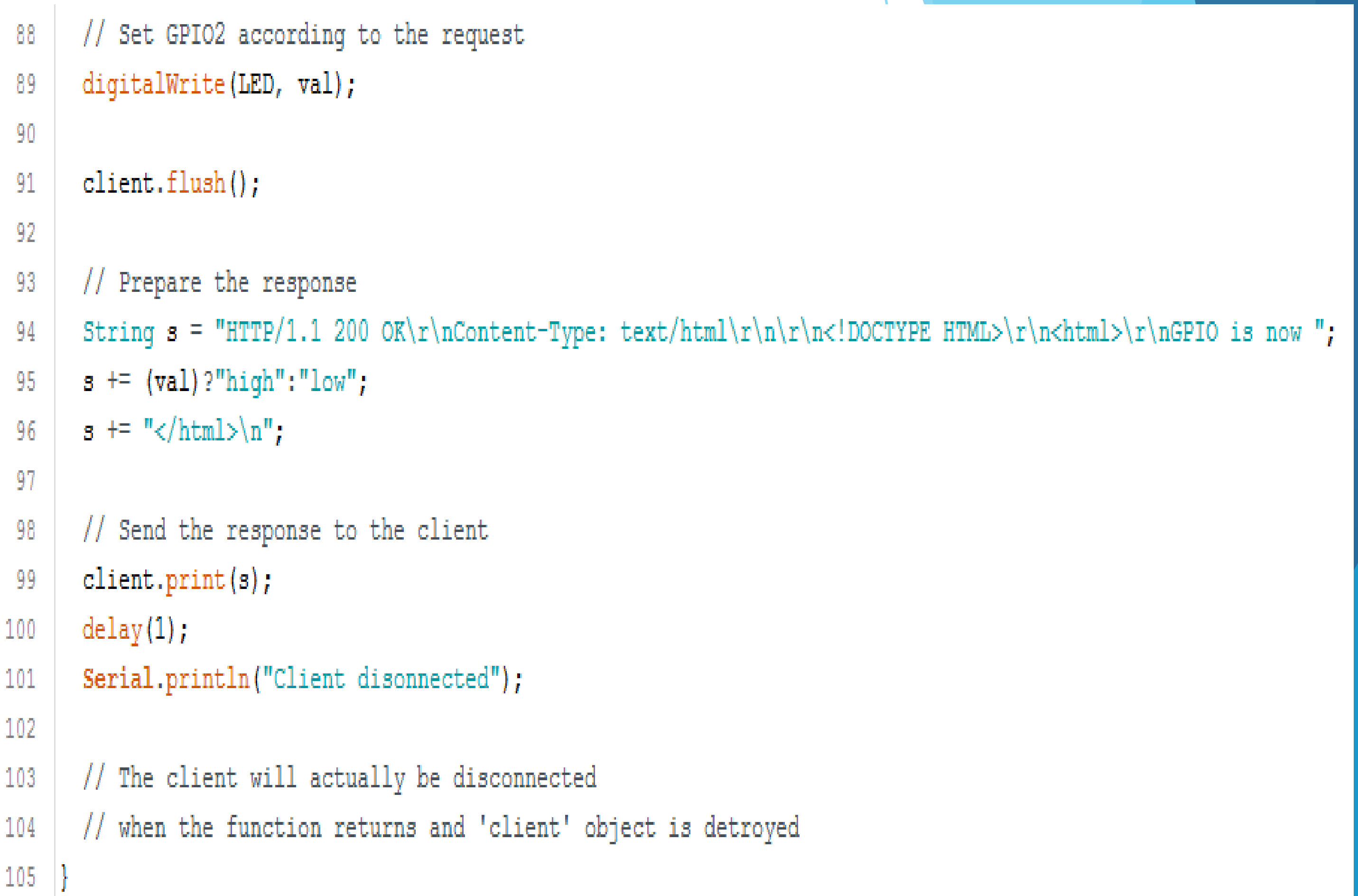
192.168.43. 61

ESP8266 IP:

WIFI ANDRIOD APPLICATION - LED

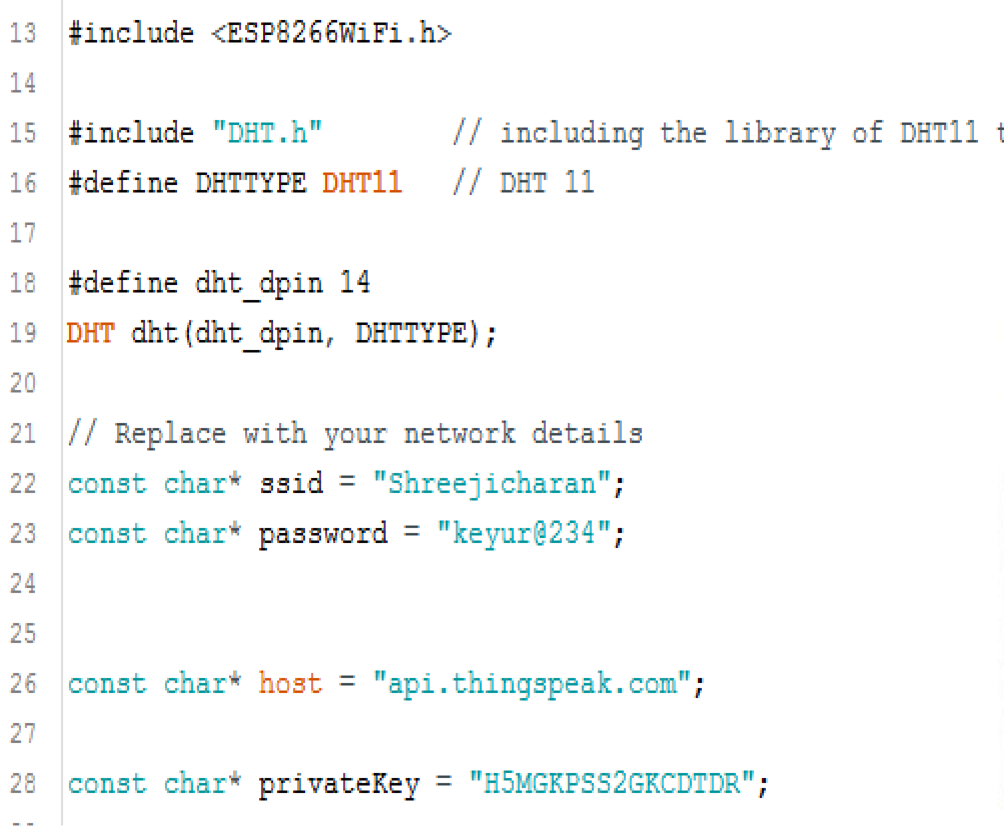


```
  // Wait until the client sends some data
  Serial.println("new client");
  while(!client.available()){
    delay(1);
  }

  // Read the first line of the request
  String req = client.readStringUntil('\r');
  Serial.println(req);
  client.flush();

  // Match the request
  int val;
  if (req.indexOf("?pin=ON") != -1)
    val = 0;
  else if (req.indexOf("?pin=OFF") != -1)
    val = 1;
  else {
    Serial.println("invalid request");
    client.stop();
    return;
  }
```

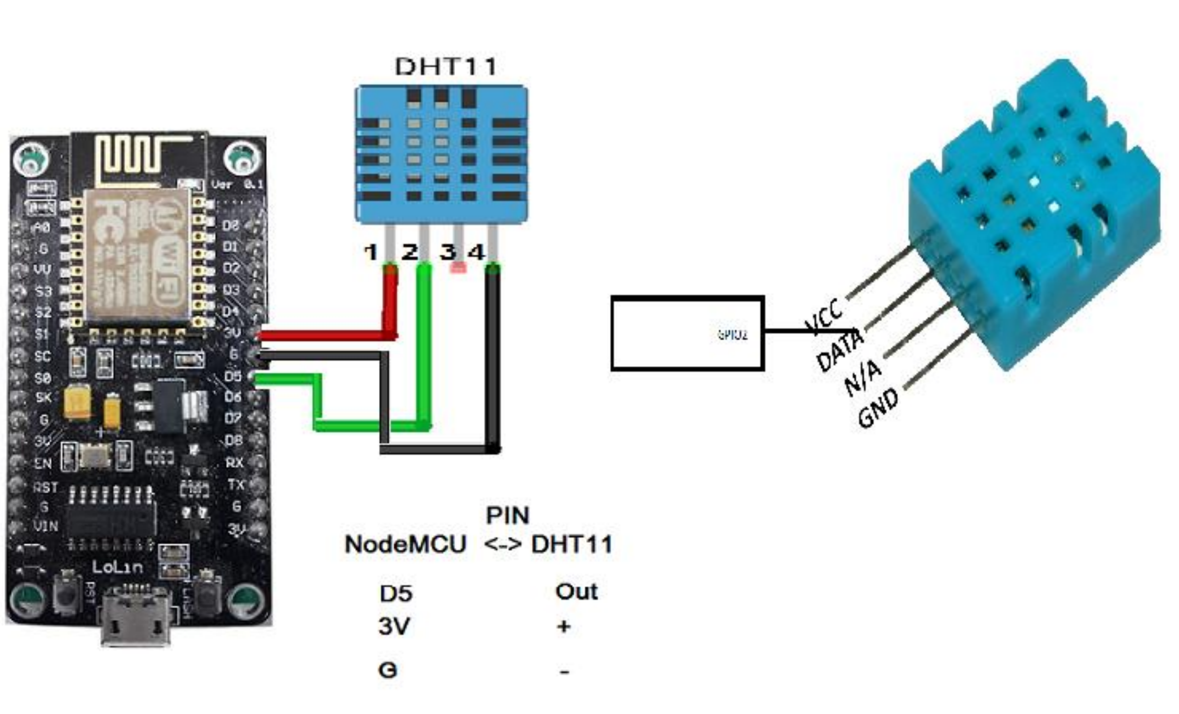


```
  // Set GPIO2 according to the request
  digitalWrite(LED, val);

  client.flush();

  // Prepare the response
  String s = "HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE HTML>\r\n<html>\r\nGPIO is now ";
  s += (val)?"high":"low";
  s += "</html>\n";

  // Send the response to the client
  client.print(s);
  delay(1);
  Serial.println("Client disonnected");

  // The client will actually be disconnected
  // when the function returns and 'client' object is detroyed
}
```

THINGSPEAK WEBSITE SENSOR DATA PLOT

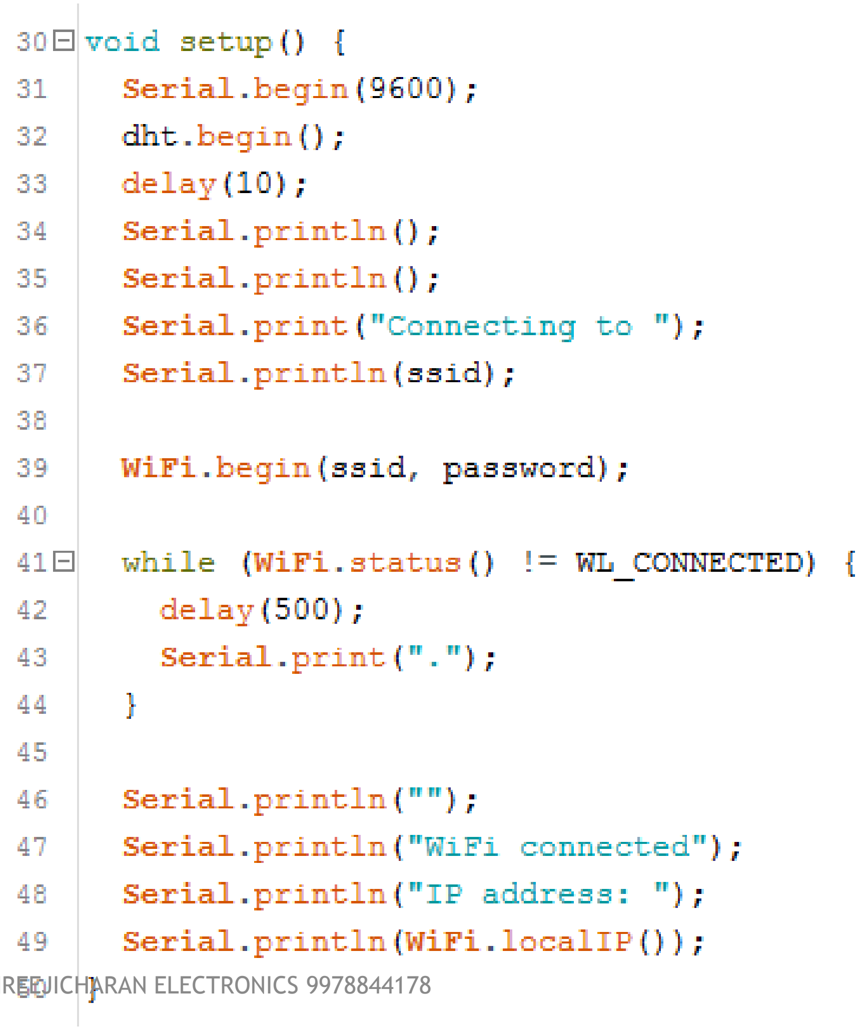


```
#include <ESP8266WiFi.h>

#include "DHT.h"         // including the library of DHT11 temperature and humidity sensor
#define DHTTYPE DHT11    // DHT 11

#define dht_dpin 14
DHT dht(dht_dpin, DHTTYPE);

// Replace with your network details
const char* ssid = "Shreejicharan";
const char* password = "keyur@234";



const char* host = "api.thingspeak.com";

const char* privateKey = "H5MGKPSS2GKCDTDR";
```

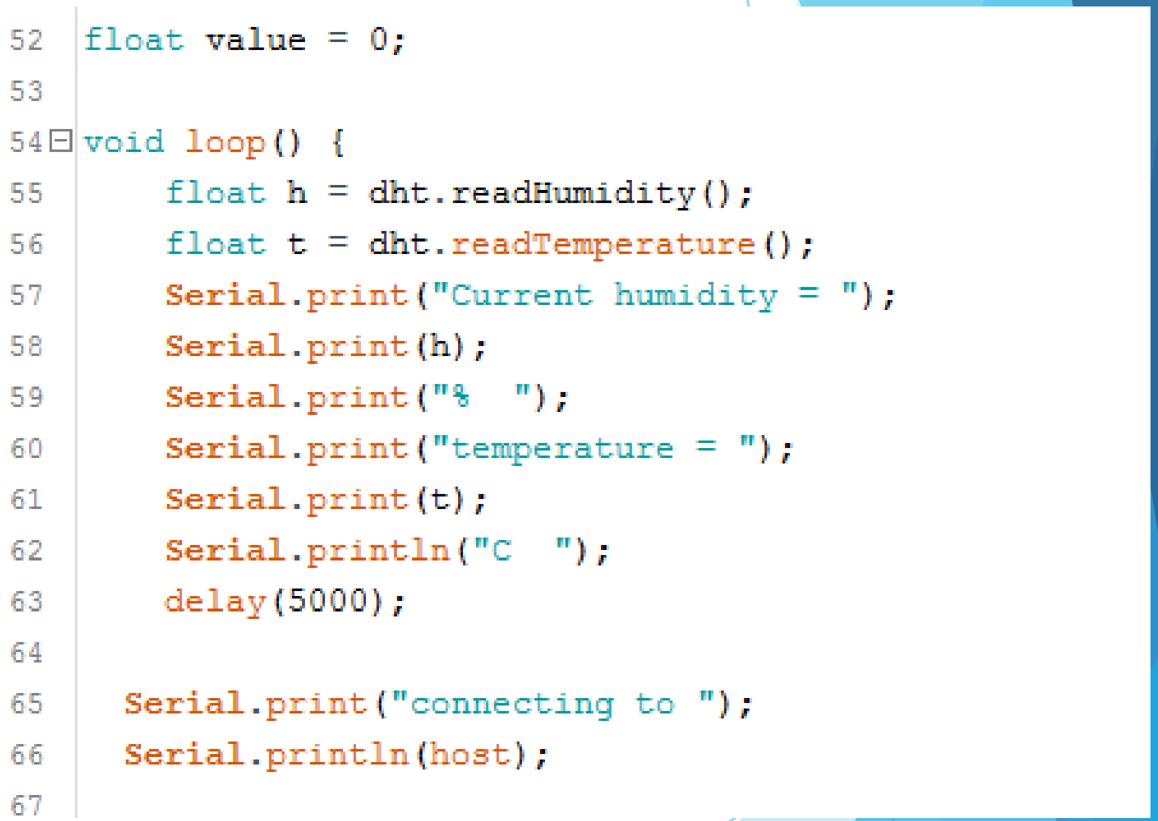


NodeMCU	PIN <-> DHT11
D5	Out
3V	+
G	-

THINGSPEAK WEBSITE SENSOR DATA PLOT

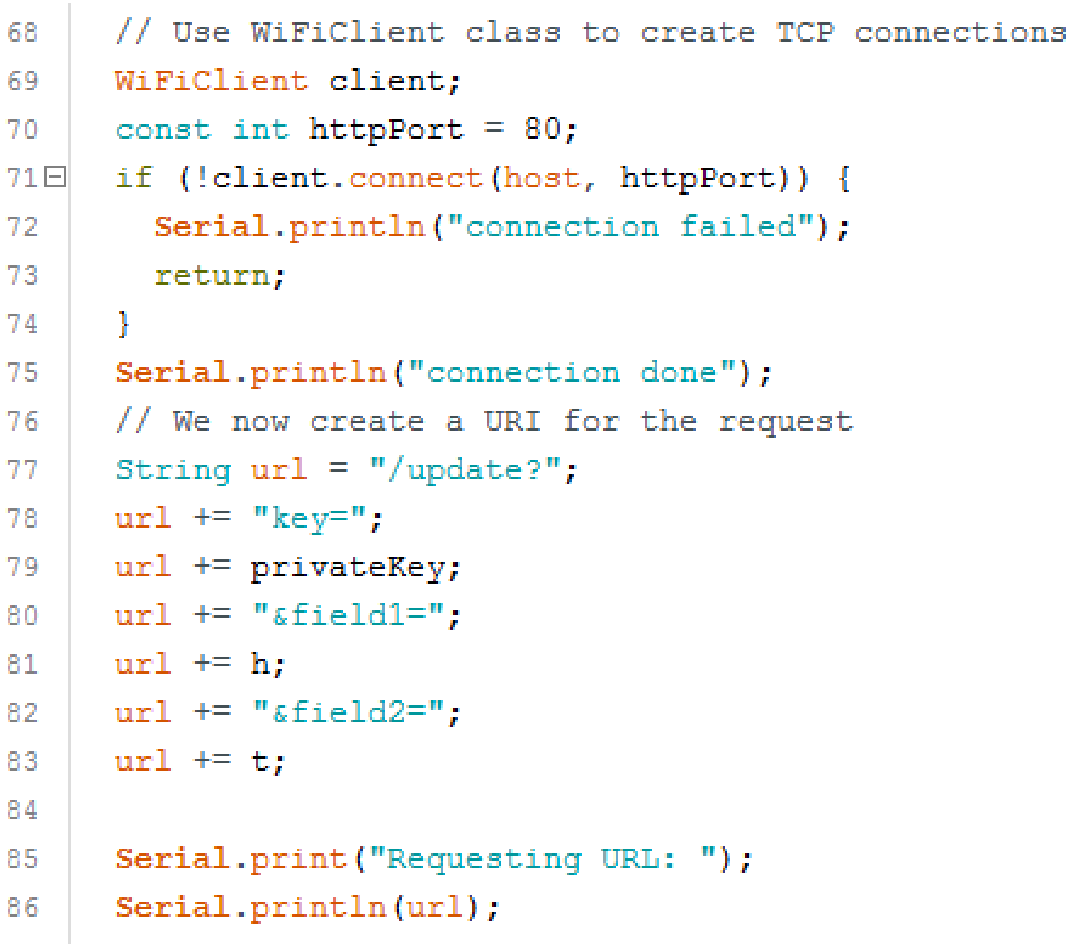


```
void setup() {
  Serial.begin(9600);
  dht.begin();
  delay(10);
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

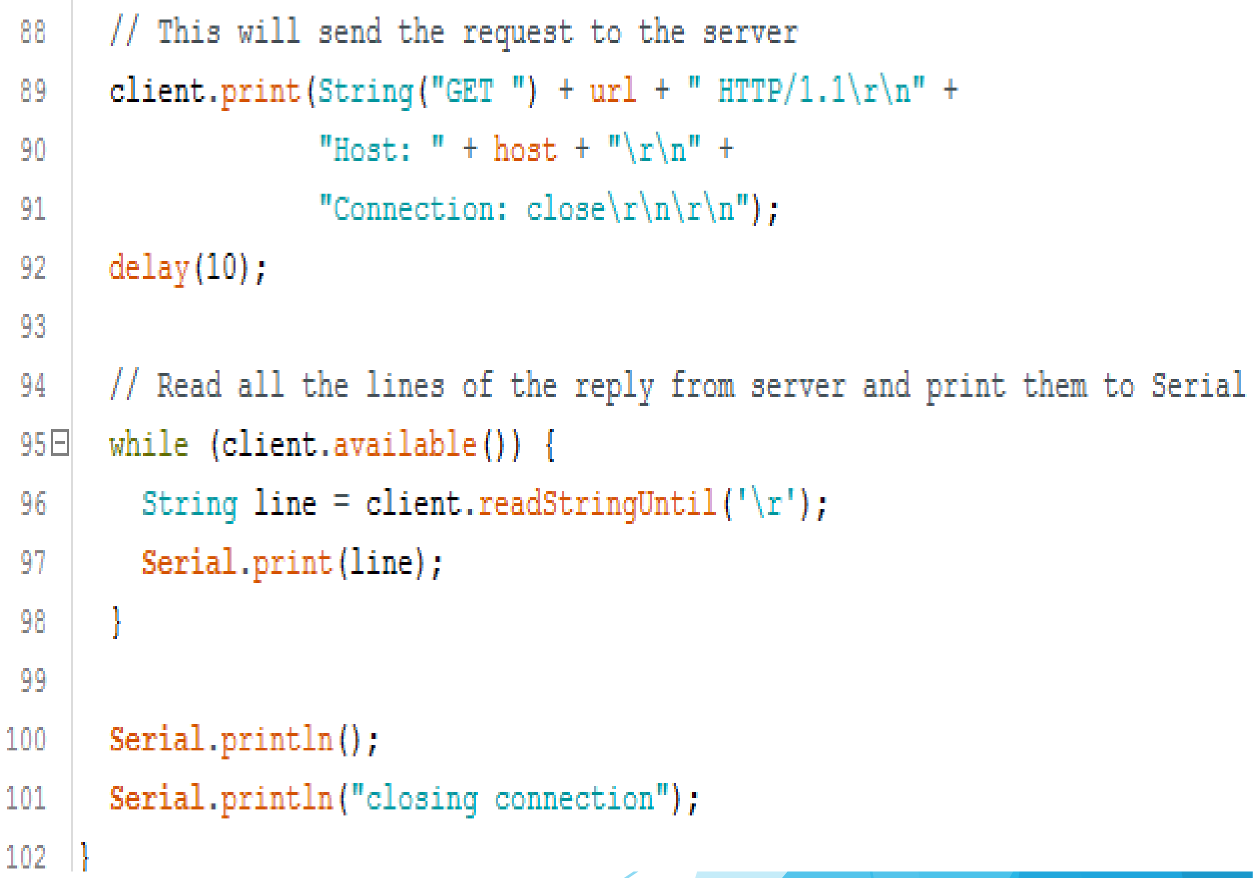


```
float value = 0;

void loop() {
    float h = dht.readHumidity();
    float t = dht.readTemperature();
    Serial.print("Current humidity = ");
    Serial.print(h);
    Serial.print("%  ");
    Serial.print("temperature = ");
    Serial.print(t);
    Serial.println("C  ");
    delay(5000);

  Serial.print("connecting to ");
  Serial.println(host);

```

THINGSPEAK WEBSITE SENSOR DATA PLOT

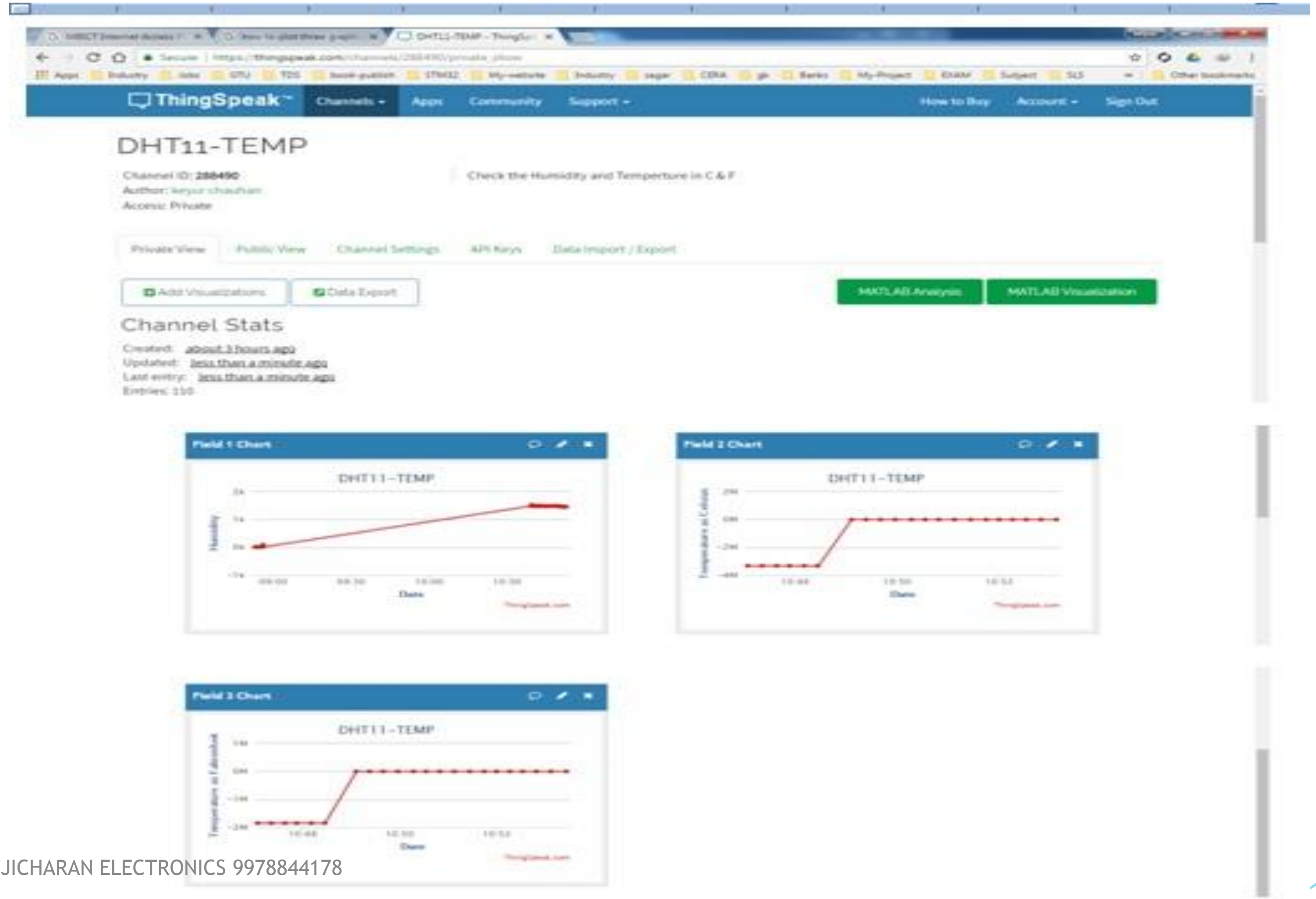


```
// Use WiFiClient class to create TCP connections
WiFiClient client;
const int httpPort = 80;
if (!client.connect(host, httpPort)) {
  Serial.println("connection failed");
  return;
}
Serial.println("connection done");
// We now create a URI for the request
String url = "/update?";
url += "key=";
url += privateKey;
url += "&field1=";
url += h;
url += "&field2=";
url += t;

Serial.print("Requesting URL: ");
Serial.println(url);
```



```
// This will send the request to the server
client.print(String("GET ") + url + " HTTP/1.1\r\n" +
             "Host: " + host + "\r\n" +
             "Connection: close\r\n\r\n");
delay(10);

// Read all the lines of the reply from server and print them to Serial
while (client.available()) {
  String line = client.readStringUntil('\r');
  Serial.print(line);
}

Serial.println();
Serial.println("closing connection");
}
```

THINGSPEAK WEBSITE SENSOR DATA PLOT



https://thingspeak.com/channels/288490/private_show

DHT11-TEMP

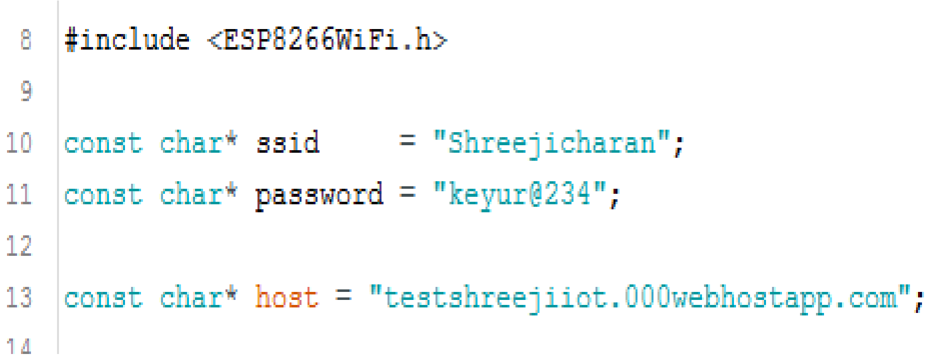
Channel ID: 288490
Author: keyur chauhan
Access: Private

Check the Humidity and Temperture in C & F

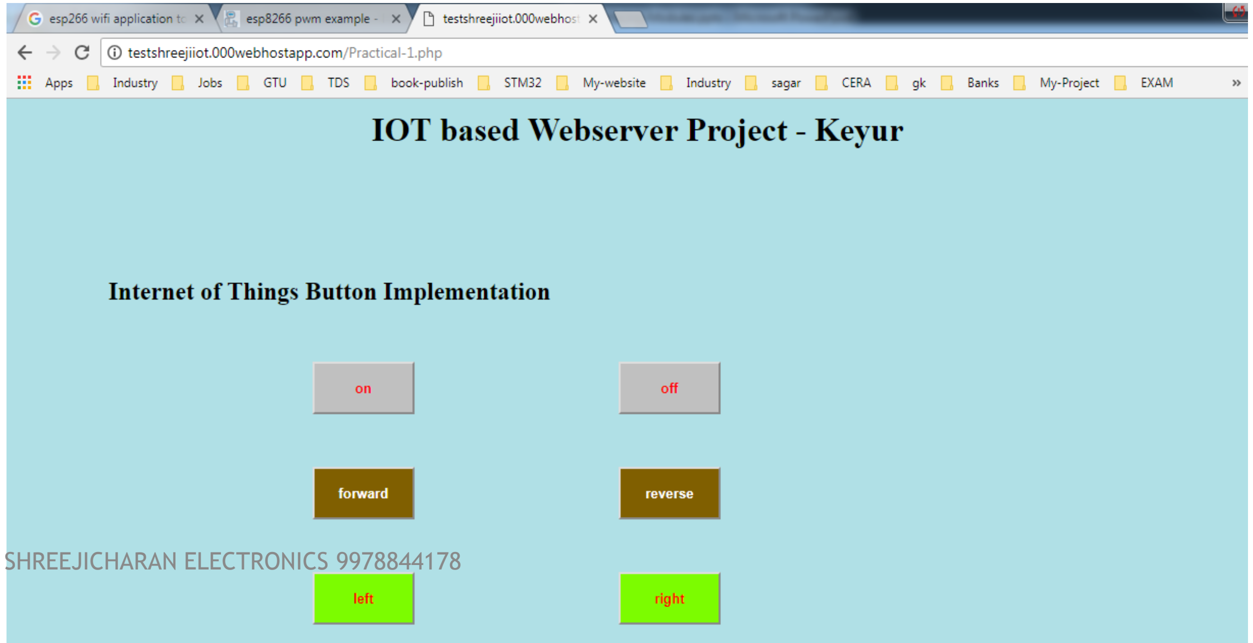
Channel Stats

Created: about 3 hours ago
Updated: less than a minute ago
Last entry: less than a minute ago
Entries: 110

ONLINE - WEBSERVER - LED

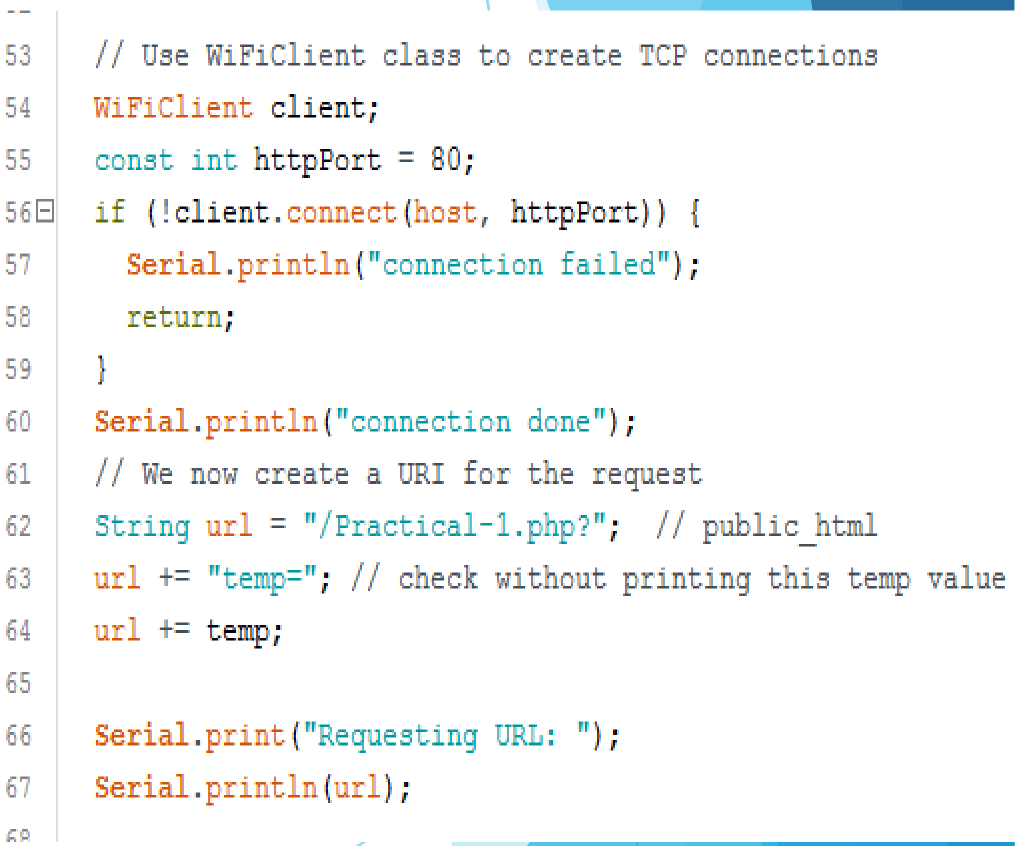


```
#include <ESP8266WiFi.h>

const char* ssid     = "Shreejicharan";
const char* password = "keyur@234";

const char* host = "testshreejiiot.000webhostapp.com";
```



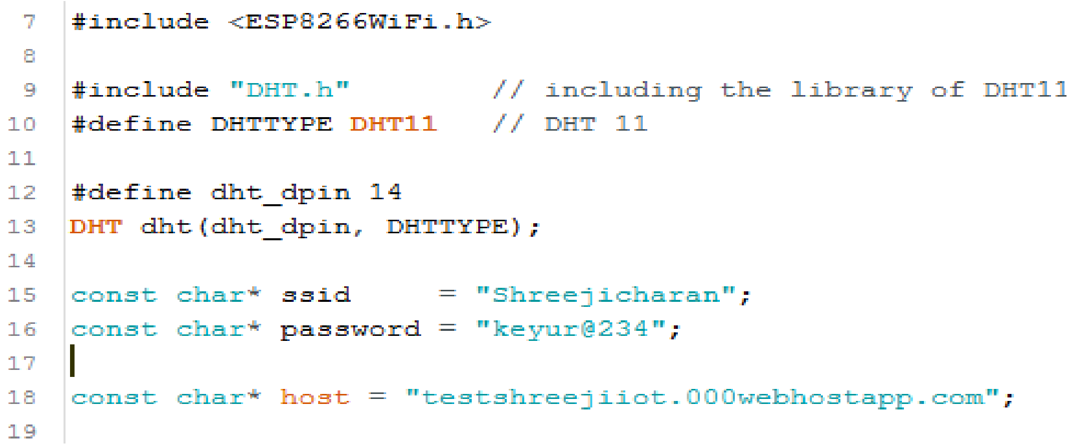
IOT based Webserver Project - Keyur

Internet of Things Button Implementation

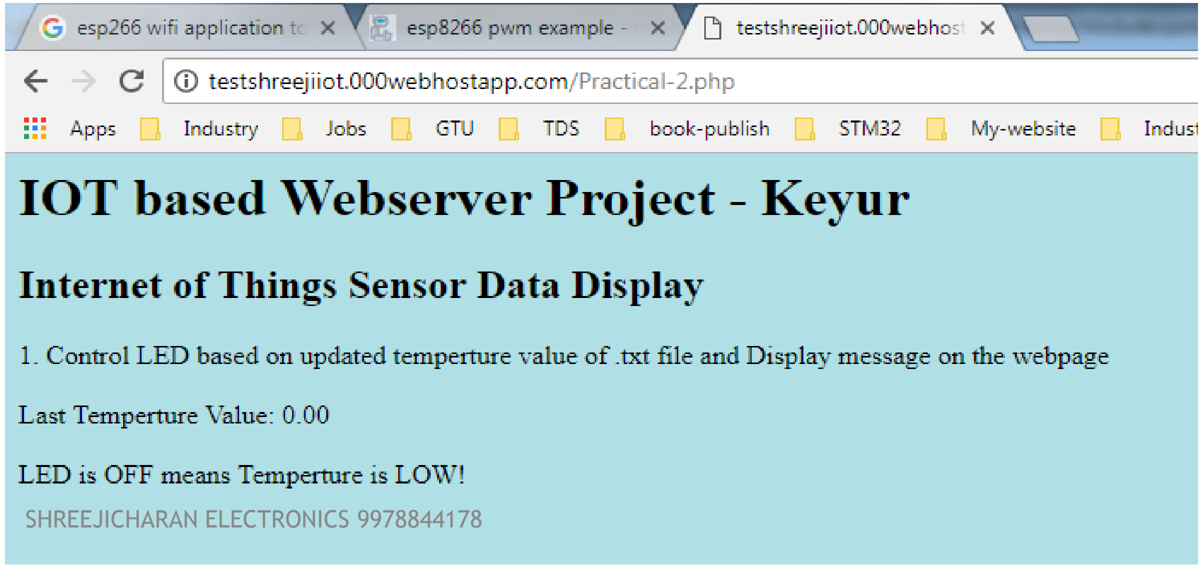


```
// Use WiFiClient class to create TCP connections
WiFiClient client;
const int httpPort = 80;
if (!client.connect(host, httpPort)) {
  Serial.println("connection failed");
  return;
}
Serial.println("connection done");
// We now create a URI for the request
String url = "/Practical-1.php?";  // public_html
url += "temp="; // check without printing this temp value
url += temp;

Serial.print("Requesting URL: ");
Serial.println(url);
```

ONLINE WEBSERVER - SENSOR



```
#include <ESP8266WiFi.h>

#include "DHT.h"         // including the library of DHT11
#define DHTTYPE DHT11    // DHT 11

#define dht_dpin 14
DHT dht(dht_dpin, DHTTYPE);

const char* ssid     = "Shreejicharan";
const char* password = "keyur@234";

const char* host = "testshreejiiot.000webhostapp.com";

```



esp266 wifi application tc
esp8266 pwm example -
testshreejiiot.000webhost
testshreejiiot.000webhostapp.com/Practical-2.php
Apps
Industry
Jobs
GTU
TDS
book-publish
STM32
My-website
Indust

IOT based Webserver Project - Keyur

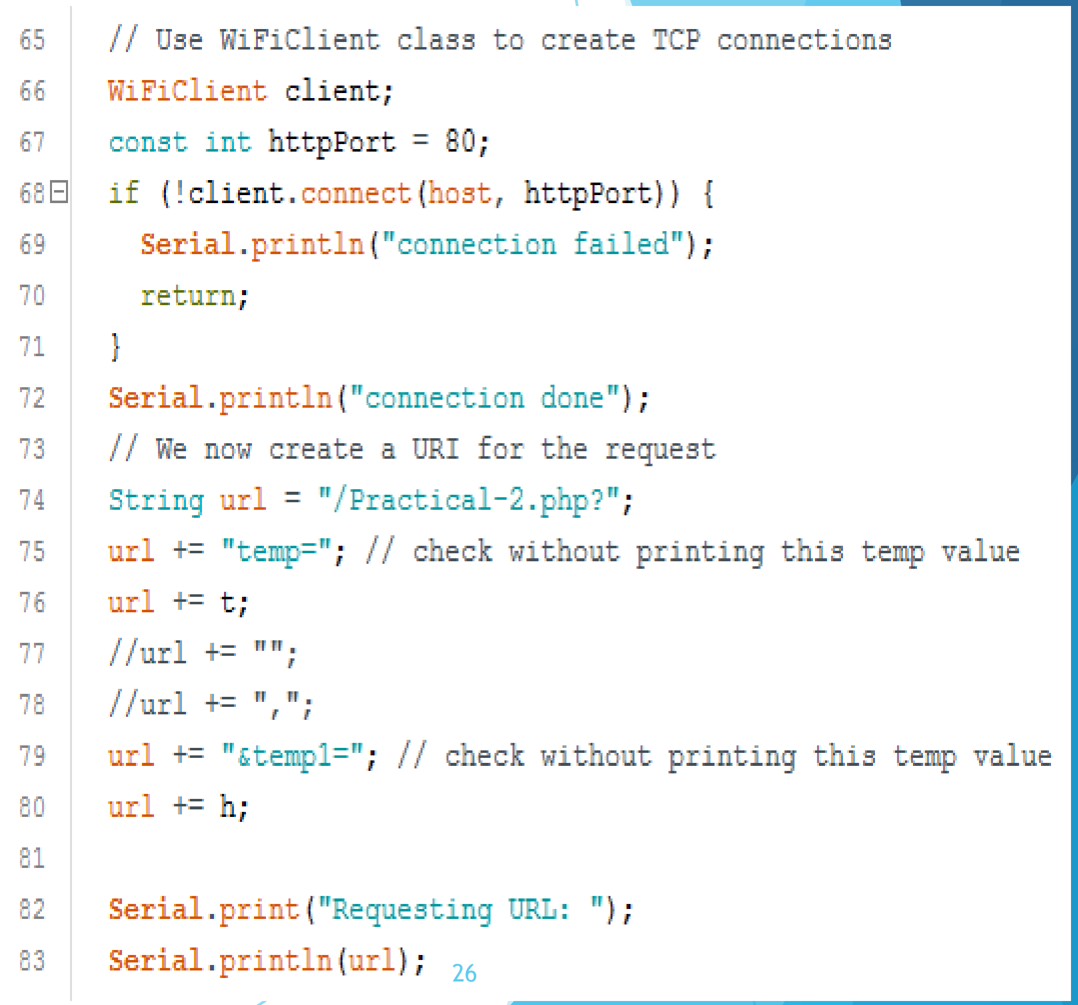
Internet of Things Sensor Data Display

1. Control LED based on updated temperture value of .txt file and Display message on the webpage

Last Temperture Value: 0.00

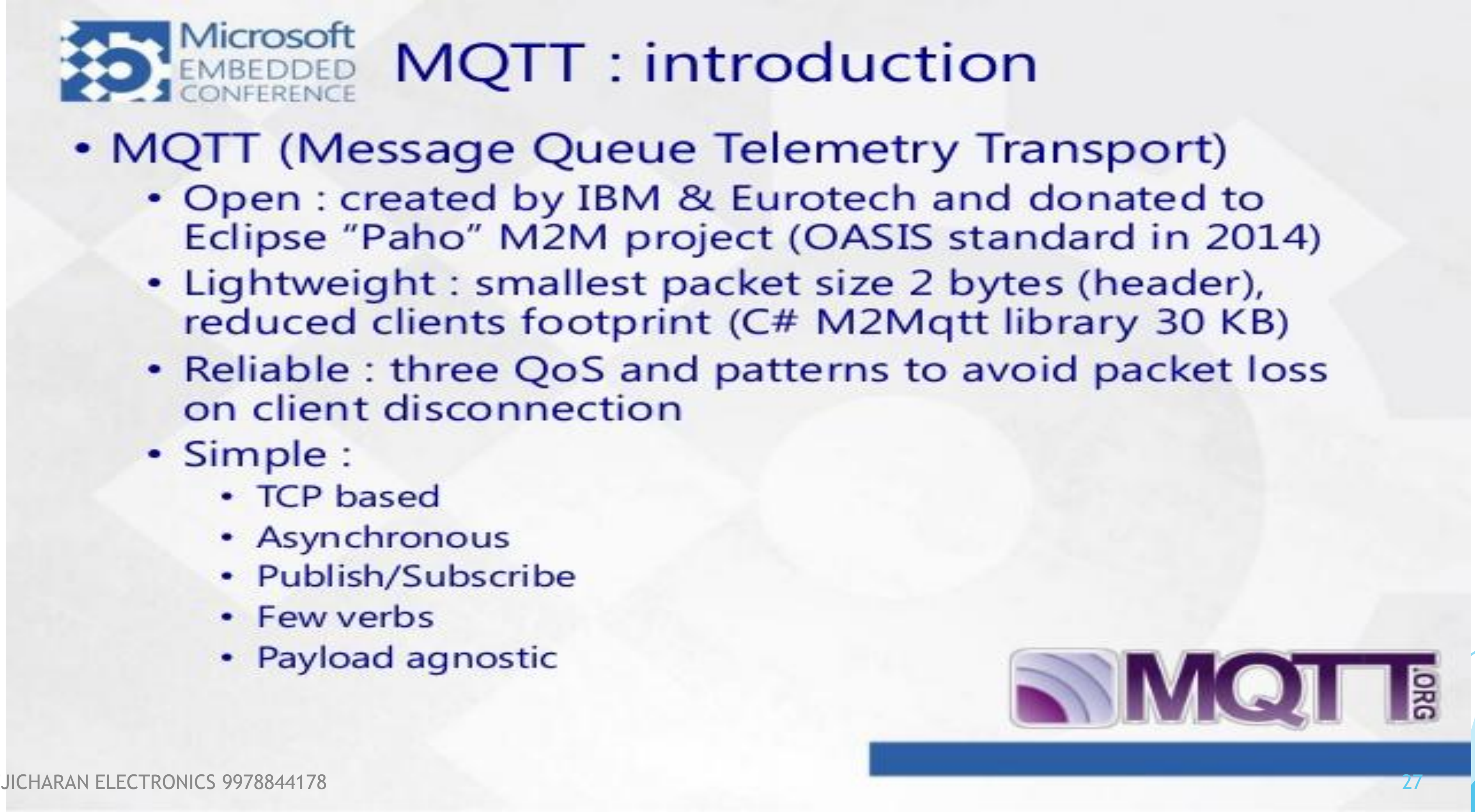
LED is OFF means Temperture is LOW!

SHREEJICHARAN ELECTRONICS 9978844178



```
// Use WiFiClient class to create TCP connections
WiFiClient client;
const int httpPort = 80;
if (!client.connect(host, httpPort)) {
  Serial.println("connection failed");
  return;
}
Serial.println("connection done");
// We now create a URI for the request
String url = "/Practical-2.php?";
url += "temp="; // check without printing this temp value
url += t;
//url += "";
//url += ",";
url += "&temp1="; // check without printing this temp value
url += h;

Serial.print("Requesting URL: ");
Serial.println(url);
```

MQTT PROTOCOL



Microsoft EMBEDDED CONFERENCE

MQTT : introduction

- MQTT (Message Queue Telemetry Transport)
  - Open : created by IBM & Eurotech and donated to Eclipse "Paho" M2M project (OASIS standard in 2014)
  - Lightweight : smallest packet size 2 bytes (header), reduced clients footprint (C# M2Mqtt library 30 KB)
  - Reliable : three QoS and patterns to avoid packet loss on client disconnection
  - Simple :
    - TCP based
    - Asynchronous
    - Publish/Subscribe
    - Few verbs
    - Payload agnostic

MQTT.ORG

MQTT PROTOCOL

- MQTT (originally termed Message Queuing Telemetry Transport) is a simple "publish and subscribe" messaging protocol for use over TCP/IP.

- It was designed to connect restricted devices in remote locations for sporadic messaging over low bandwidth, high-latency or unreliable networks, with minimal code size needed.

- Its original purpose was to collect data from multiple devices while using limited bandwidth and provide the information to several subscribers.

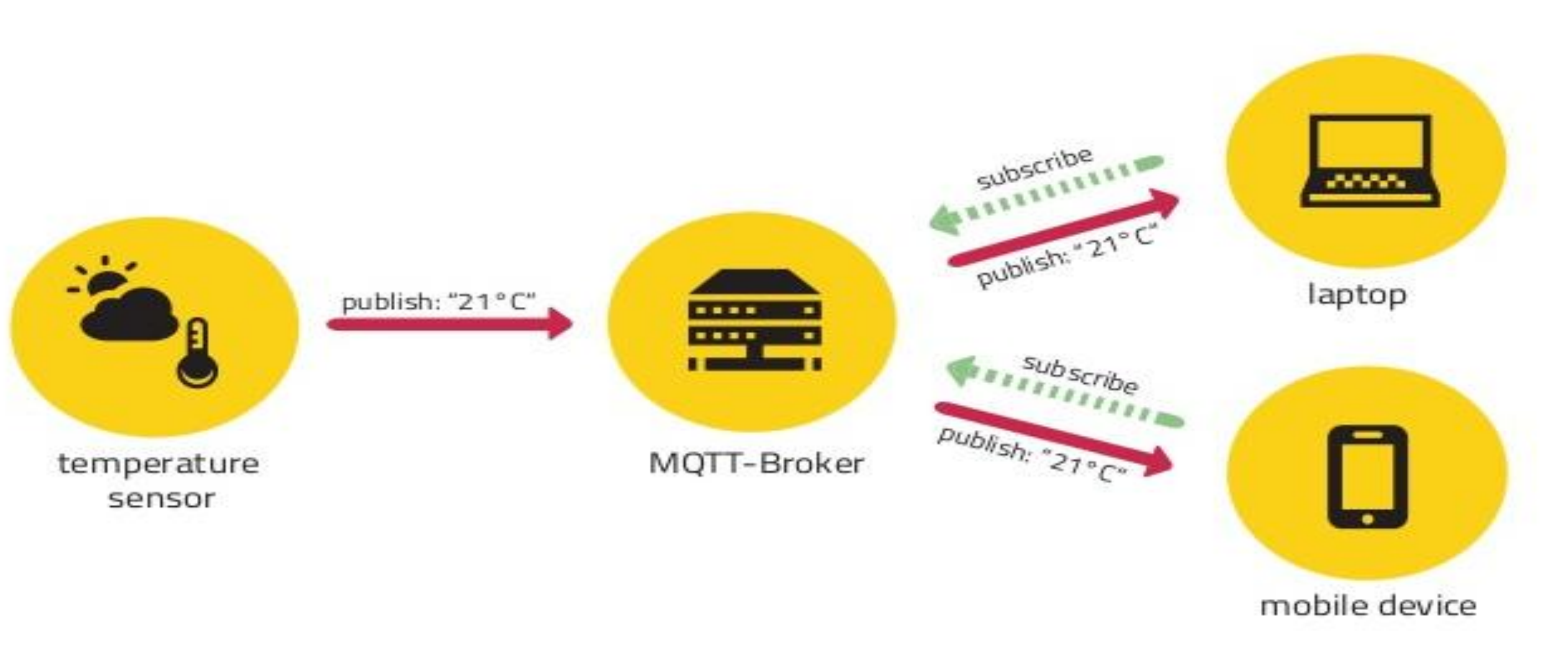
- It tries to ensure reliability and some degree of assurance of delivery. MQTT is now mainly used as a Machine-to-Machine (M2M) Internet of Things (IoT) connectivity protocol.

MQTT PROTOCOL

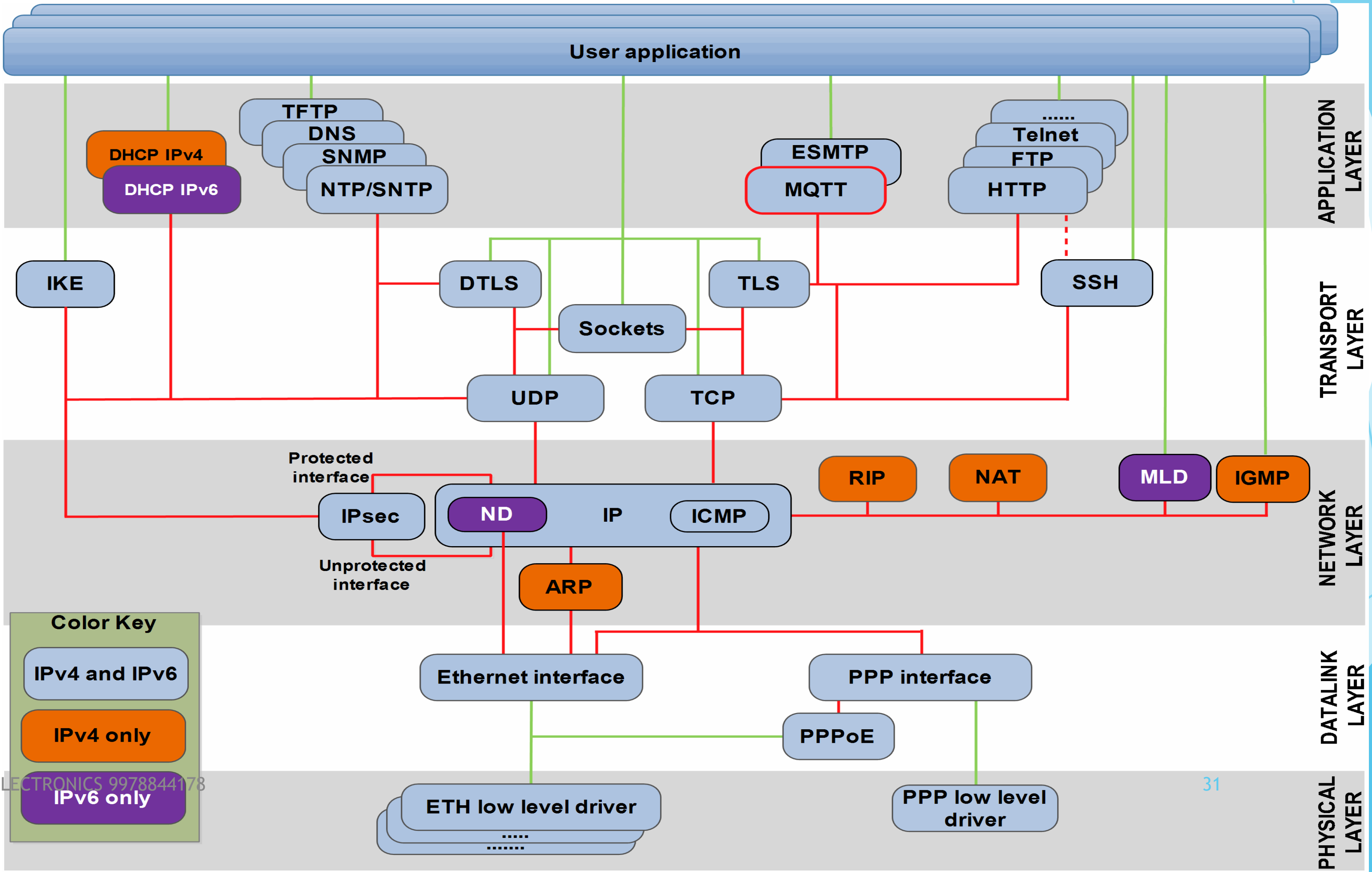
- MQTT is an Application Layer protocol that operates over TCP, normally using one of two ports: 1883 for clear data and 8883 for connections over Transport Layer Security (TLS).

- The protocol provides many useful capabilities, including different levels of "Quality of Service" (QoS), "client down" notification, automatic topic re-registration, and the ability to receive data from clients that have gone offline.

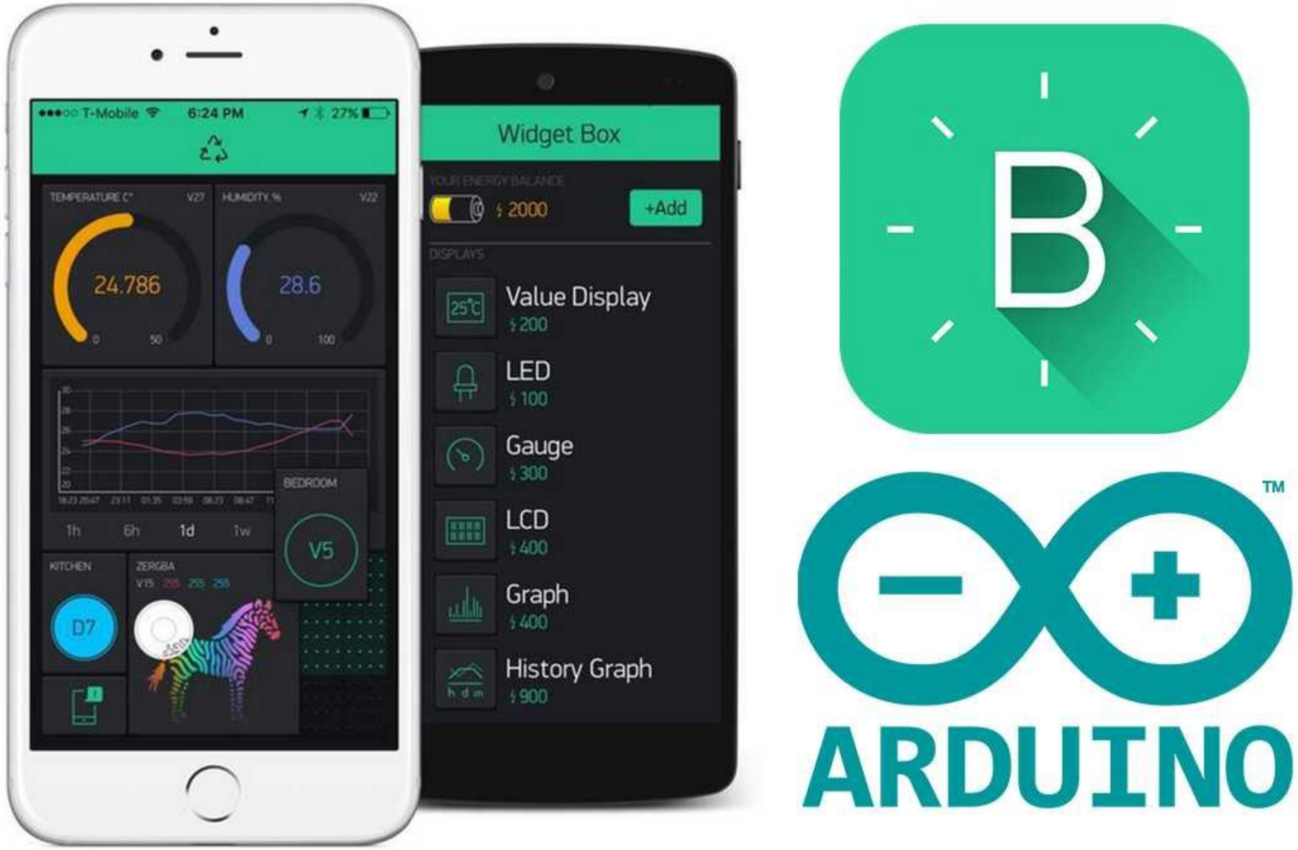
MQTT PROTOCOL



MQTT PROTOCOL

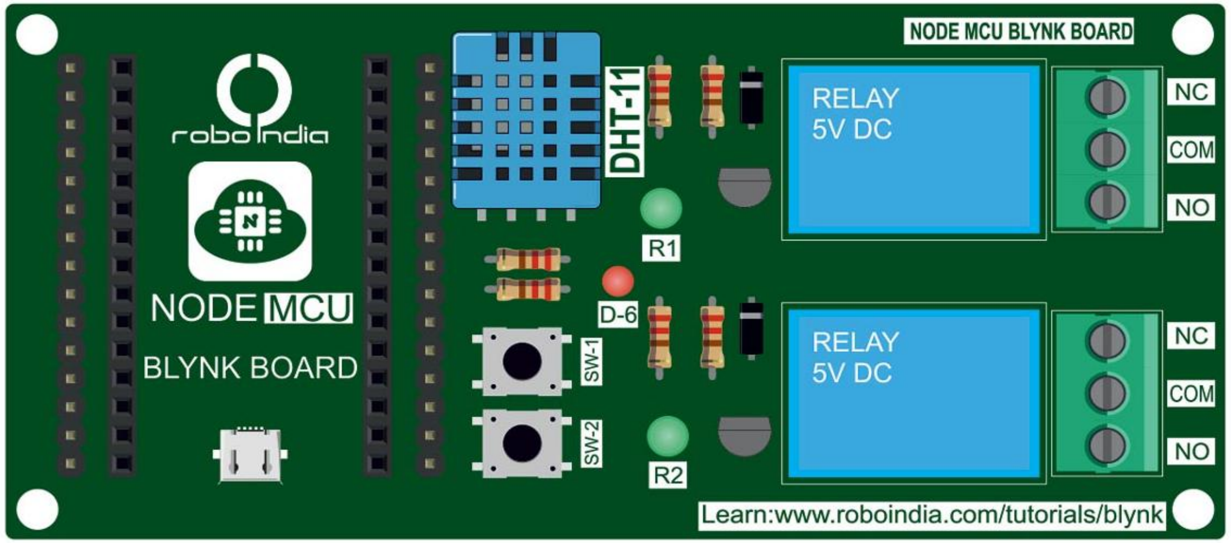


BLYNK BOARD WITH ANDRIOD APPLICATION

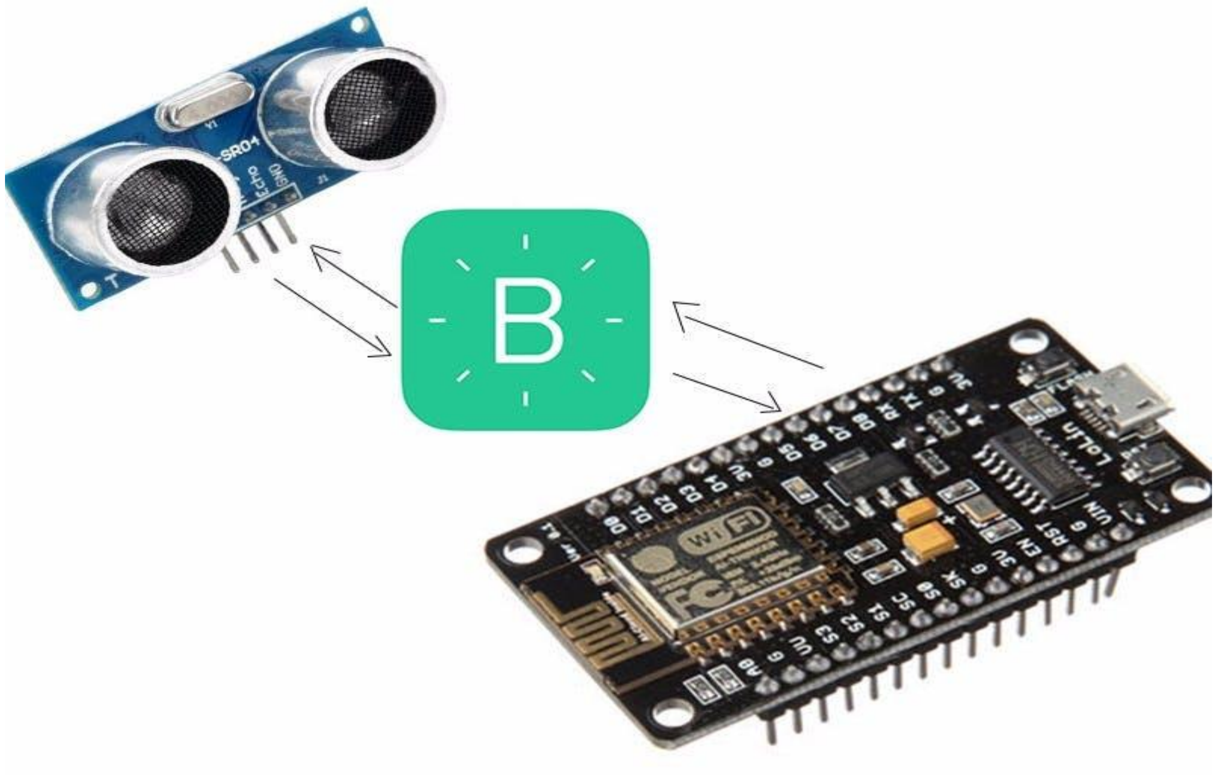


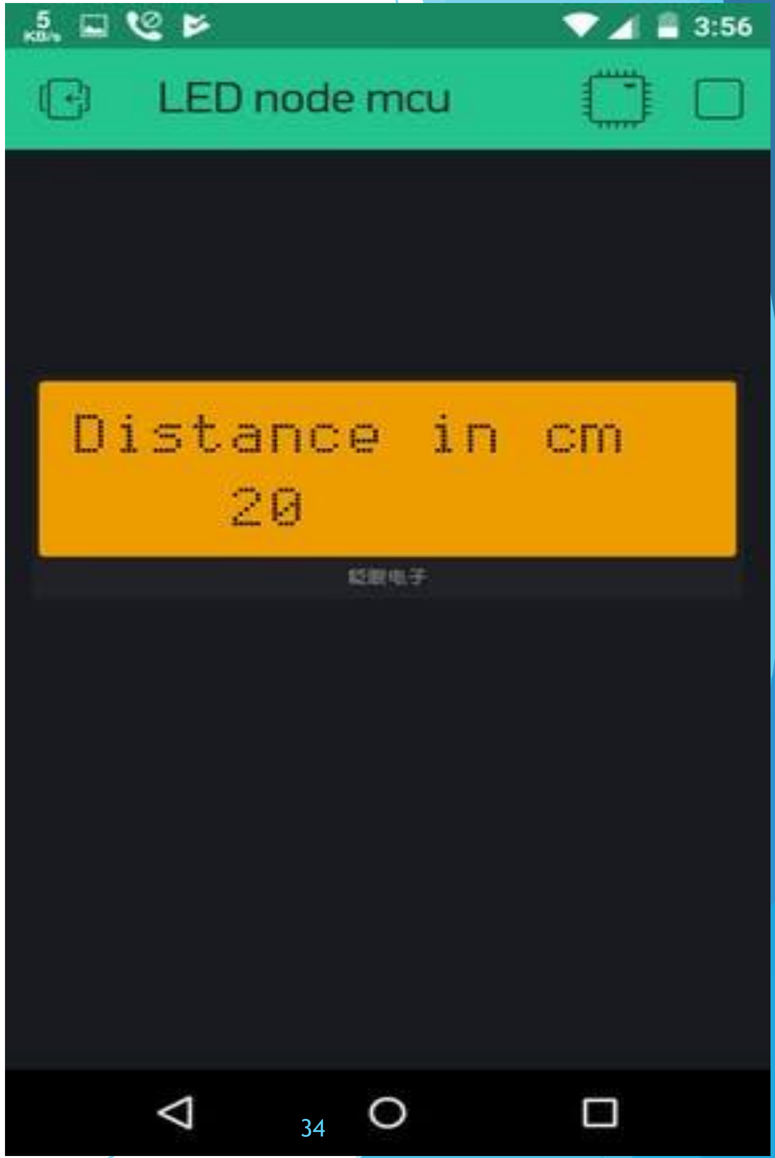
HARDWARE

BLYNK BOARD



Ultrasonic Sensor with Blynk and NodeMCU

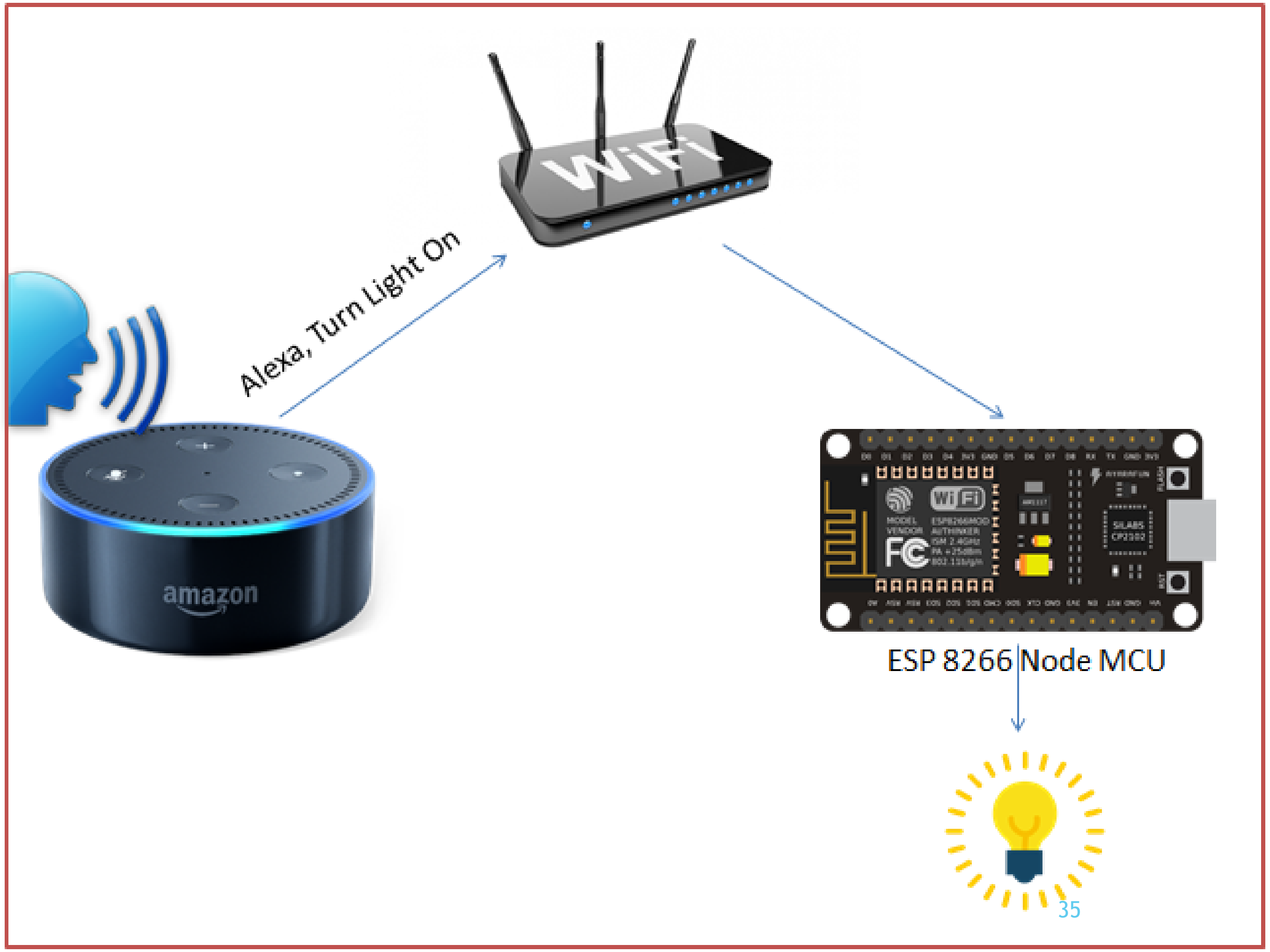


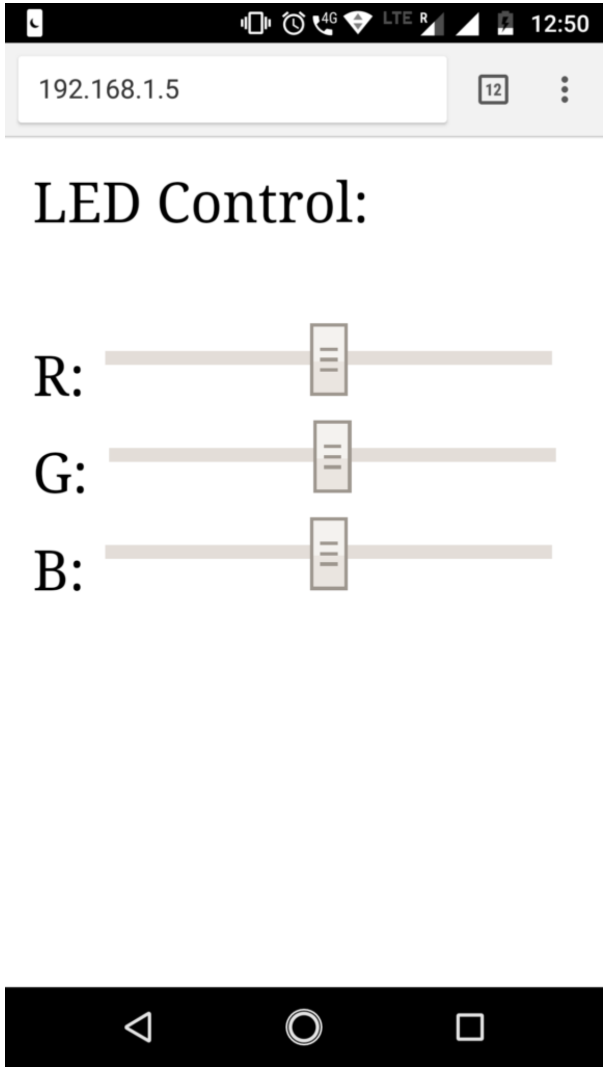


LED node mcu

```
Distance in cm
      20
```

Alexa -Amazon interface with NodeMCU



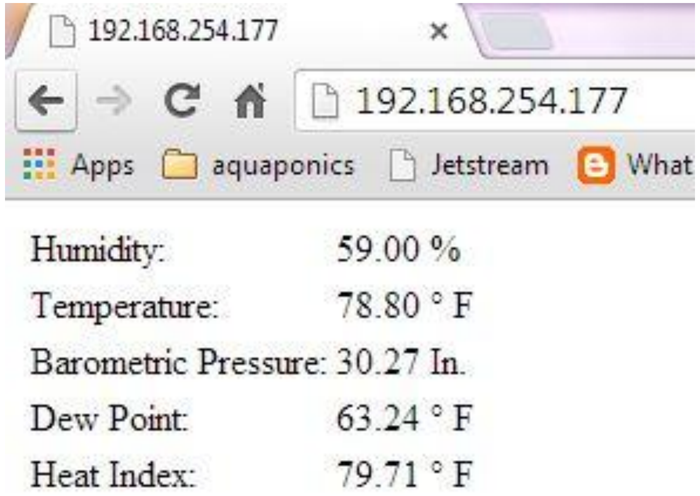


LED Control:

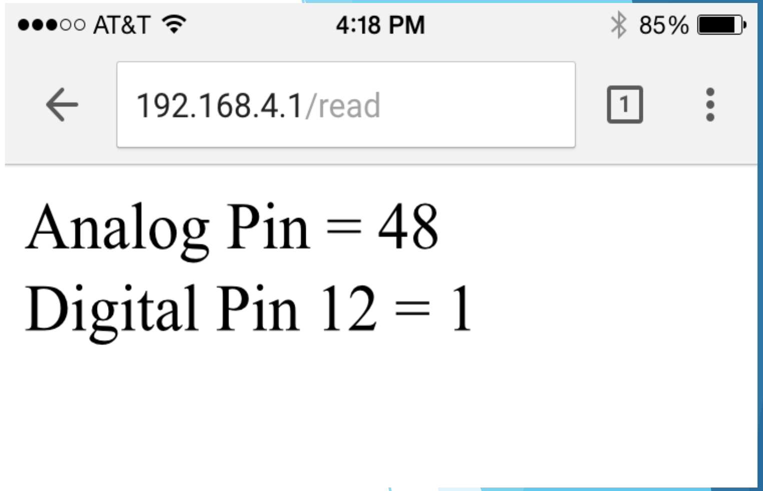
R:

G:

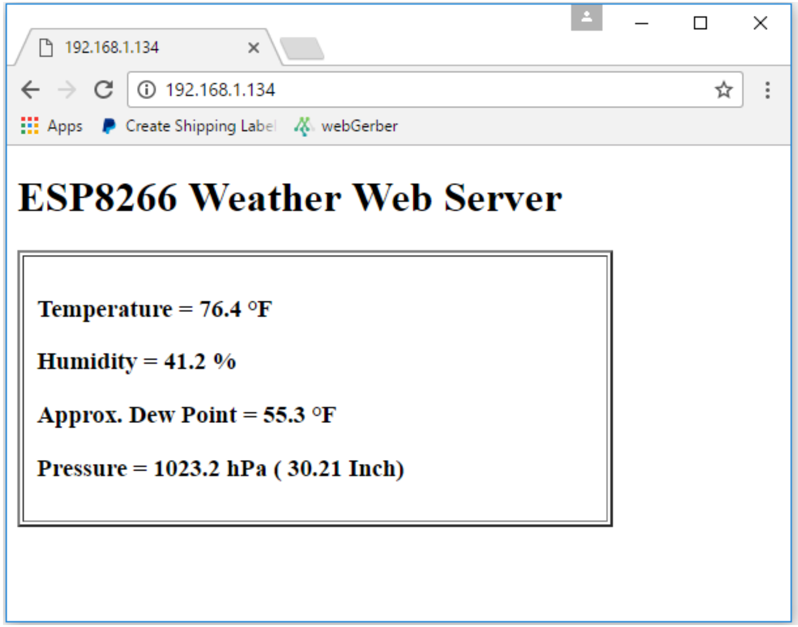
B:



Humidity:	59.00 %
Temperature:	78.80 ° F
Barometric Pressure:	30.27 In.
Dew Point:	63.24 ° F
Heat Index:	79.71 ° F



Analog Pin = 48
Digital Pin 12 = 1



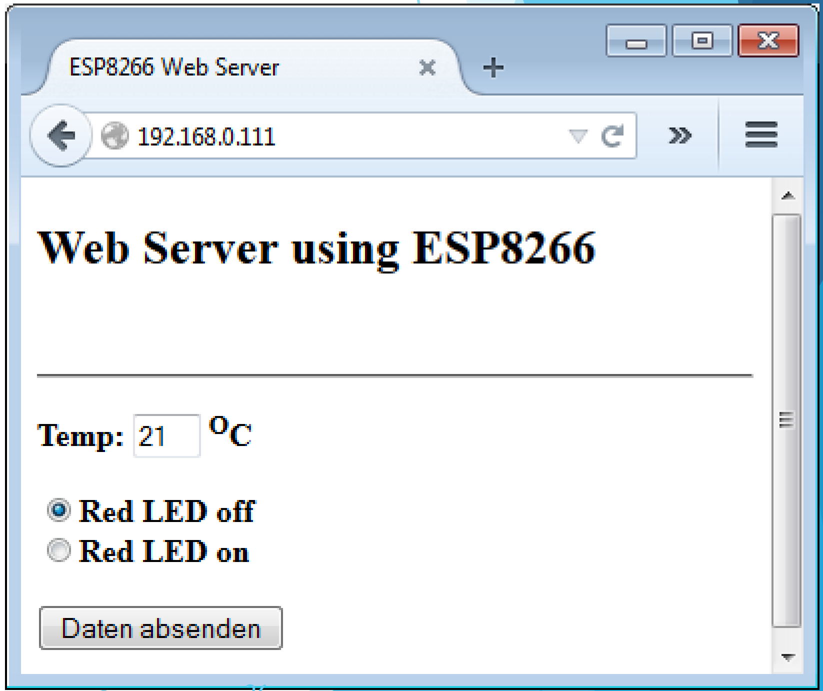
ESP8266 Weather Web Server

Temperature = 76.4 °F

Humidity = 41.2 %

Approx. Dew Point = 55.3 °F

Pressure = 1023.2 hPa (30.21 Inch)

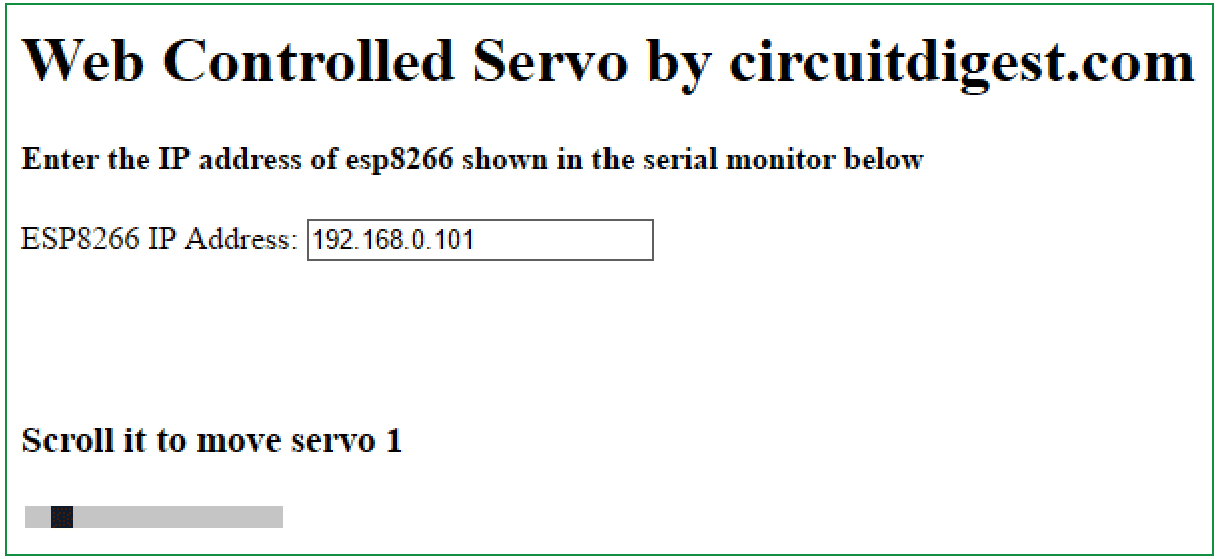


Web Server using ESP8266

Temp: 21 °C

● Red LED off
○ Red LED on

Daten absenden

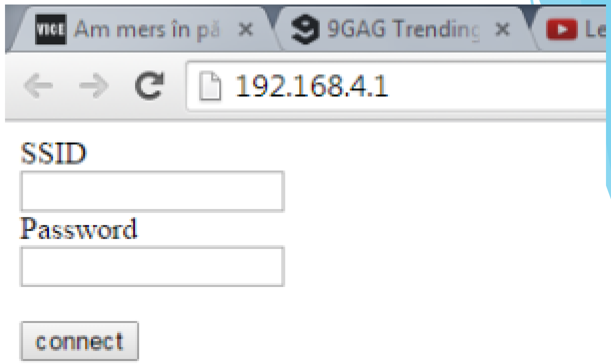


Web Controlled Servo by circuitdigest.com

Enter the IP address of esp8266 shown in the serial monitor below

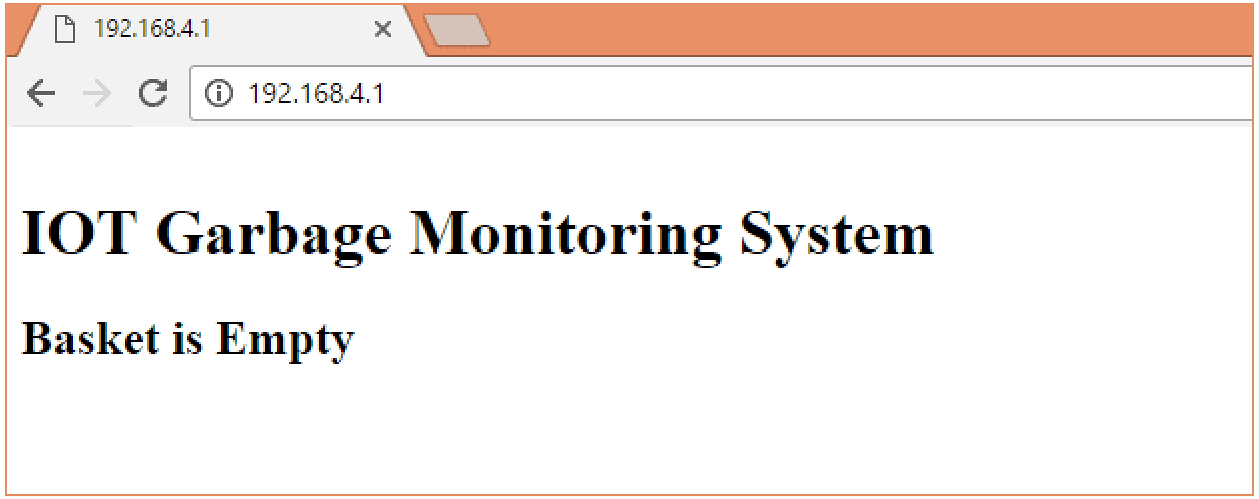
ESP8266 IP Address: 192.168.0.101

Scroll it to move servo 1



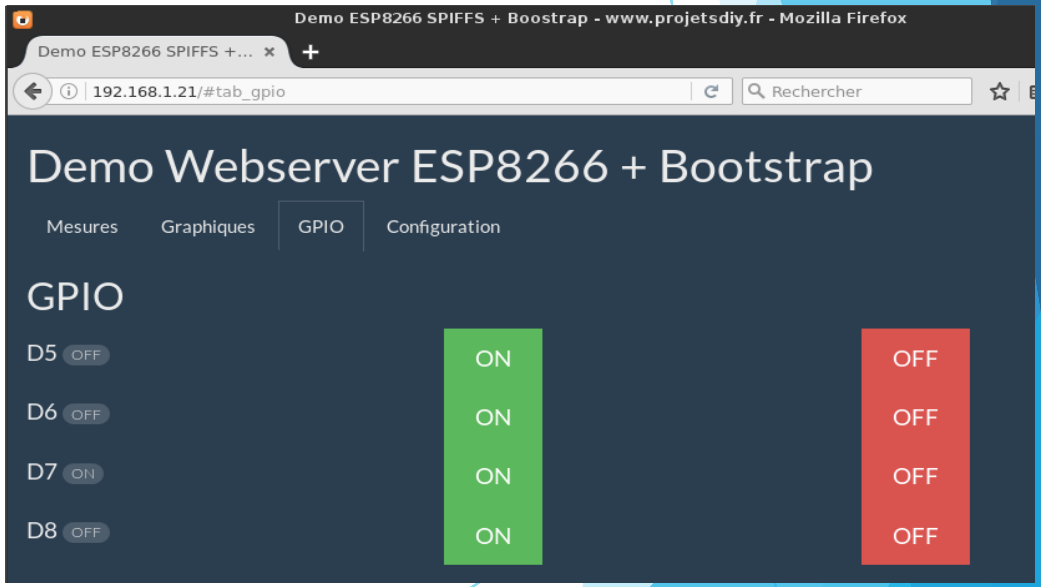
SSID

Password



IOT Garbage Monitoring System

Basket is Empty



Demo ESP8266 SPIFFS + Boostrap - www.projetsdiy.fr - Mozilla Firefox
Demo ESP8266 SPIFFS +...
192.168.1.21/#tab_gpio
Rechercher

Demo Webserver ESP8266 + Bootstrap

GPIO

D5 OFF ON OFF

D6 OFF ON OFF

D7 ON ON OFF

D8 OFF ON OFF

Thank you