/\*

1. Name of the Project: IoT Temp Humidity Logger

2. Developer: Vikrant Gupta Responsible for R&D, Firmware Design, Ideation, Schematic Design, Product Design

Pranamya Responsible for PCB Design & Fabrication, Schematic Layout Design

3. Final Version After Total R & D For Firmware Name is:- \*\*| "IoT\_Temp/Humi\_Logger\_Firmware\_V\_1.0" |\*\*

4. PCB Design name for the same is : \*\*| "IoT\_Temp/Humi\_Logger\_PCB\_DESIGN\_V\_1.0" |\*\*

\*| This Entire Project is copyright to \*|"Vertical Farming Technologies Pvt Ltd."|\*

#Features:

1. Logging Temperature and Humidity to Blynk App with Graph as well ass showing Battery Percentage

2. Showing the same at OLED Display over the Device

3. Auto connecting to the saved network once Configured.

4. Turing to access Point mode if Saved network is not Available with Network as "City\_Greens" No password

5. Once Turned On and connected to any browser based device configuiration page can be opened with address

"192.168.4.1"

6. It can configure SSID, Password, and Blynk Auth Token

7. Once saved then device will auto connected to that network and start sending data.

\*/

/\*|-----------------------------------------------------------------------------|

Including Library Files

|-----------------------------------------------------------------------------|\*/

#include <DNSServer.h> // Library for DNS Server

#include <WiFiManager.h> // Library for Wi-Fi Manager Updated By Vikrant Gupta

#include <ESP8266WebServer.h> // ESP Web Server Hosting

#include <BlynkSimpleEsp8266.h> // Blynk integration with ESP

#include <SimpleTimer.h> // Timer

#include <ArduinoJson.h> // JSON Usage

#include "DHT.h" // DHT

#include <SPI.h> // SPI The protocol Library

#include <Wire.h> // Independent Library

Adafruit\_SSD1306 display(-1); //OLED Initialisation

/\*|-----------------------------------------------------------------------------|

Display Off Intervals

|-----------------------------------------------------------------------------|\*/

const unsigned long DisoffInterval = 4000; // Display off after 8 Seconds

unsigned long previousTime = 0; // Using Milis the previous time

/\*|-----------------------------------------------------------------------------|

Defining DHT22 Pins and Blynk details

|-----------------------------------------------------------------------------|\*/

#define BLYNK\_PRINT Serial // Blynk Define over Serial

#define SET\_PIN 0 // For Blynk

#define DHTPIN 2 // DHT22 Connected to Physical Pin D4

#define DHTTYPE DHT22 // DHT22 Type Selection

DHT dht(DHTPIN, DHTTYPE); // Declaration of DHT22 PIN and Its Type

SimpleTimer timer; // Timer Initialisation

char blynk\_token[40] = ""; // Blynk Token Defination

bool shouldSaveConfig = false; // Boolian value Selection for Saving Configuration

/\*|-----------------------------------------------------------------------------|

Set Up Starts Here with Connecting to wife and Access Point mode Functions

|-----------------------------------------------------------------------------|\*/

void setup()

{

pinMode(15, OUTPUT);

dht.begin();

Serial.begin(9600); // Initialising Serial

display.begin(SSD1306\_SWITCHCAPVCC, 0x3C); // Initialize with the I2C addr 0x3D (for the 128x64)

pinMode(SET\_PIN, INPUT\_PULLUP); // Pul Up Pin Declaration

/\*|-----------------------------------------------------------------------------|

Splash Screen Once Device Starts

|-----------------------------------------------------------------------------|\*/

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(28, 2);

display.println("City Greens");

display.setCursor(2, 14);

display.println("IoT Temp/Humi Logger");

display.display();

delay(1200);

// Serial.println();

pinMode(SET\_PIN, INPUT\_PULLUP);

delay(500);

/\*|-----------------------------------------------------------------------------|

Connecting to Exixting Netwrok or starting Access Point mode

|-----------------------------------------------------------------------------|\*/

Serial.println("mounting FS..."); // read configuration from FS json

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(3, 3);

display.println("Conneecting... to" );

display.setCursor(14, 12);

display.println(" Saved Network ");

display.display();

display.clearDisplay(); // Showing Details for Tring to Connect

if (SPIFFS.begin())

{

Serial.println("mounted file system");

if (SPIFFS.exists("/config.json")) //file exists, reading and loading

{

Serial.println("reading config file");

File configFile = SPIFFS.open("/config.json", "r");

if (configFile)

{

Serial.println("opened config file");

size\_t size = configFile.size();

std::unique\_ptr<char[]> buf(new char[size]); // Allocate a buffer to store contents of the file.

configFile.readBytes(buf.get(), size);

DynamicJsonBuffer jsonBuffer;

JsonObject& json = jsonBuffer.parseObject(buf.get());

json.printTo(Serial);

if (json.success())

{

Serial.println("\nparsed json");

strcpy(blynk\_token, json["blynk\_token"]);

}

else

{

Serial.println("failed to load json config");

}

configFile.close();

}

}

}

else

{

Serial.println("failed to mount FS");

}

WiFiManagerParameter custom\_blynk\_token("blynk", "blynk token", blynk\_token, 32);

WiFiManager wifiManager;

wifiManager.setSaveConfigCallback(saveConfigCallback);

wifiManager.addParameter(&custom\_blynk\_token);

if (digitalRead(SET\_PIN) == LOW)

{

wifiManager.resetSettings();

}

if (!wifiManager.autoConnect("CityGreens"))

{

display.clearDisplay();

display.setCursor(10, 0);

display.println(F("Failed To connect"));

display.setCursor(11, 10);

display.println(F("Hot-Spot Started"));

display.setCursor(10, 20);

display.println(F("New Configuration"));

display.display();

delay(1000);

wifiManager.startConfigPortal("City\_Greens");

}

Serial.println("Wi-Fi Connected");

strcpy(blynk\_token, custom\_blynk\_token.getValue());

if (shouldSaveConfig) // save the custom parameters to FS

{

Serial.println("saving config");

DynamicJsonBuffer jsonBuffer;

JsonObject& json = jsonBuffer.createObject();

json["blynk\_token"] = blynk\_token;

File configFile = SPIFFS.open("/config.json", "w");

if (!configFile)

{

Serial.println("failed to open config file for writing");

}

json.printTo(Serial);

json.printTo(configFile);

configFile.close();

}

Serial.println();

Serial.print("local ip : ");

Serial.println(WiFi.localIP());

Serial.print("Blynk Token : ");

Serial.println(blynk\_token);

Blynk.config(blynk\_token);

display.clearDisplay();

display.setCursor(0, 4);

display.println(F("Connected to Network"));

display.setCursor(20, 14);

display.println(WiFi.localIP());

display.display();

delay(400);

display.clearDisplay();

}

/\*|-----------------------------------------------------------------------------|

^|^|^ Ends Here connecting to Wi-Fi and starting Hotspot part on condition

|-----------------------------------------------------------------------------|\*/

/\*|-----------------------------------------------------------------------------|

Main Loop Starts Here

|-----------------------------------------------------------------------------|\*/

void loop()

{

SendSensor();

Blynk.run();

timer.run();

}

/\*|-----------------------------------------------------------------------------|

Function for "SendSensor" Used in Loop

|-----------------------------------------------------------------------------|\*/

void SendSensor()

{

digitalWrite(15, LOW);

delay(1000);

digitalWrite(15, HIGH);

delay(1000);

float h = dht.readHumidity();

float t = dht.readTemperature();

float q = map(analogRead(A0), 4200.0f, 2900.0f, 20, 52);

if (isnan(h) || isnan(t) )

{

//Serial.println(F("Failed to read from DHT sensor!"));

display.clearDisplay();

display.setCursor(26, 10);

display.println(F("SENSOR FAULT"));

display.display();

return;

}

// Serial.print(F("Humidity: "));

// Serial.print(h);

// Serial.print(F("% Temperature: "));

// Serial.print(t);

// Serial.print(F("°C "));

// Serial.print(f);

// Serial.print(" Batt = ");

// Serial.println(q);

/\*|-----------------------------------------------------------------------------|

Displaying Data over OLED Display

|-----------------------------------------------------------------------------|\*/

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(SSD1306\_WHITE);

display.setCursor(18, 13);

display.print(F("Humi = "));

display.print(h);

display.println(" %");

display.setCursor(18, 3);

display.print(F("Temp = "));

display.print(t);

display.println(" °C");

display.setCursor(18, 23);

display.print(F("Batt = "));

display.print(q);

display.println(" %");

display.display();

/\*|-----------------------------------------------------------------------------|

Transmitting data to Blynk App

|-----------------------------------------------------------------------------|\*/

Blynk.virtualWrite(V6, h);

Blynk.virtualWrite(V5, t);

Blynk.virtualWrite(V0, q);

delay(1000);

/\*|-----------------------------------------------------------------------------|

Going to Sleep

|-----------------------------------------------------------------------------|\*/

unsigned long currentTime = millis();

if (currentTime - previousTime >= DisoffInterval)

{

display.clearDisplay();

display.setCursor(26, 10);

display.println(F("Going to Sleep"));

display.display();

delay(500);

display.ssd1306\_command(SSD1306\_DISPLAYOFF); // To switch display off

ESP.deepSleep(30e6); // To sleep Mode for 5 Minutes

}

}

/\*|-----------------------------------------------------------------------------|

Configuration call Back function to use saved network to connect

|-----------------------------------------------------------------------------|\*/

void saveConfigCallback ()

{

Serial.println("Should save config");

shouldSaveConfig = true;

}