#ifdef ESP32

#include <WiFi.h>

#else

#include <ESP8266WiFi.h>

#endif

#include <WiFiClientSecure.h>

#include <DHT.h>

#include <UniversalTelegramBot.h> // Universal Telegram Bot Library written by Brian Lough: https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot

#include <ArduinoJson.h>

#define DHTPIN D4

#define motor D2

#define ldrPin A0

#define soilmoisture D1

#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

// Replace with your network credentials

const char\* ssid = "Galaxy M30sFFEF";

const char\* password = "balajiwifi";

// Use @myidbot to find out the chat ID of an individual or a group

// Also note that you need to click "start" on a bot before it can

// message you

#define CHAT\_ID "1237659840"

// Initialize Telegram BOT

#define BOTtoken "1741075007:AAHO1XTisyA-LLwp\_ZGF6mDKuwSu3zoUaRs" // your Bot Token (Get from Botfather)

#ifdef ESP8266

X509List cert(TELEGRAM\_CERTIFICATE\_ROOT);

#endif

WiFiClientSecure client;

UniversalTelegramBot bot(BOTtoken, client);

int botRequestDelay = 1000;

unsigned long lastTimeBotRan;

float h,t,l,s;

// Get sensor readings and return them as a String variable

String getReadings() {

h = dht.readHumidity();

// Read temperature as Celsius (the default)

t = dht.readTemperature();

l = analogRead(A0);

s = digitalRead(D1);

Serial.println(s);

String message = "Temperature: " + String(t) + " ºC \n";

message += "Humidity: " + String (h) + " % \n";

message += "Light Intensity: " + String (l) + " % \n";

message += "Soil Moisture: " + String (s) + " % \n";

return message;

}

//Handle what happens when you receive new messages

void handleNewMessages(int numNewMessages) {

Serial.println("handleNewMessages");

Serial.println(String(numNewMessages));

for (int i = 0; i < numNewMessages; i++) {

// Chat id of the requester

String chat\_id = String(bot.messages[i].chat\_id);

if (chat\_id != CHAT\_ID) {

bot.sendMessage(chat\_id, "Unauthorized user", "");

continue;

}

// Print the received message

String text = bot.messages[i].text;

Serial.println(text);

String from\_name = bot.messages[i].from\_name;

if (text == "/start") {

String welcome = "Welcome, " + from\_name + ".\n";

welcome += "Use the /readings command to get current readings.\n";

welcome += "Use /motoron and /motoroff \n";

welcome += "to control Motor \n";

bot.sendMessage(chat\_id, welcome, "");

}

if (text == "/readings") {

String readings = getReadings();

bot.sendMessage(chat\_id, readings, "");

}

if (text == "/motoron") {

String cmd = "Motor is switched on";

digitalWrite(motor, HIGH);

bot.sendMessage(chat\_id, cmd, "");

}

if (text == "/motoroff") {

String cmd = "Motor is switched off";

digitalWrite(motor, LOW);

bot.sendMessage(chat\_id, cmd, "");

}

}

}

void setup() {

Serial.begin(115200);

#ifdef ESP8266

configTime(0, 0, "pool.ntp.org"); // get UTC time via NTP

client.setTrustAnchors(&cert); // Add root certificate for api.telegram.org

#endif

dht.begin();

pinMode(D1,INPUT);

pinMode(motor, OUTPUT);

// Connect to Wi-Fi

// Connect to Wi-Fi

WiFi.mode(WIFI\_STA);

WiFi.begin(ssid, password);

#ifdef ESP32

client.setCACert(TELEGRAM\_CERTIFICATE\_ROOT); // Add root certificate for api.telegram.org

#endif

while (WiFi.status() != WL\_CONNECTED) {

delay(1000);

Serial.println("Connecting to WiFi..");

}

// Print ESP32 Local IP Address

Serial.println(WiFi.localIP());

bot.sendMessage(CHAT\_ID, "Bot Started", "");

}

void loop() {

if (millis() > lastTimeBotRan + botRequestDelay) {

int numNewMessages = bot.getUpdates(bot.last\_message\_received + 1);

while (numNewMessages) {

Serial.println("got response");

handleNewMessages(numNewMessages);

numNewMessages = bot.getUpdates(bot.last\_message\_received + 1);

}

lastTimeBotRan = millis();

}

s = digitalRead(D1);

if(s==1){

Serial.println(s);

bot.sendMessage(CHAT\_ID, "Ur plant needs water!!", "");

Serial.println("Ur plant needs water!!");

}

}