Assignment-4

ELP - 720 TELECOMMUNICATION NETWORKS LABORATORY

Sudhanshu Chaudhary
2019JTM2207
2019-2021

A report presented for the assignment on
ESP32 with Arduino IDE

Bharti School Of
Telecommunication Technology and Management
IIT Delhi
India
Januray 28, 2020

Contents

List of Figures

1 Problem Statement-1

1.1 Problem Statement

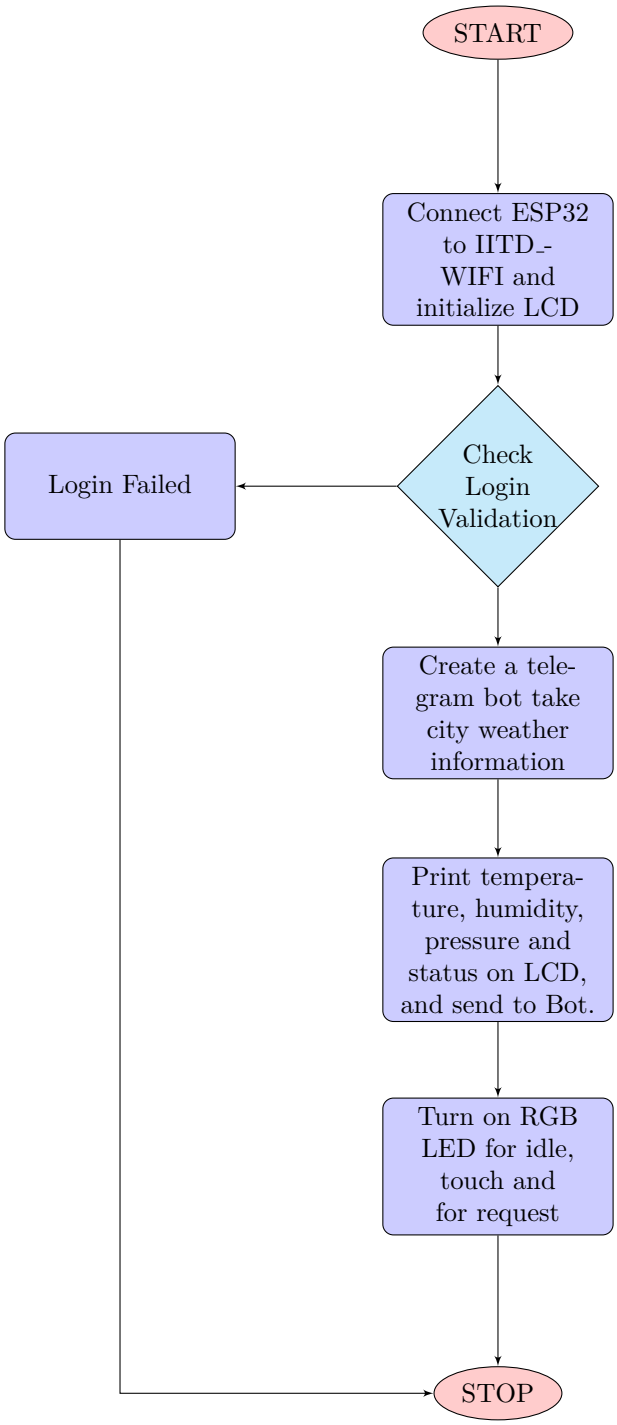
A weather station is a device that collects data related to the weather and environment using many different sensors. Weather stations are also called weather centres, personal weather stations, professional weather stations, home weather station, weather forecaster and forecasters
Let us design our own weather station!

- Connect your ESP32 to IITD_WIFI
- Create your own Telegram bot which can communicate with ESP32
- From this bot, you will send a city name
- This name will be received by your weather station (i.e. ESP32)
- And the weather information for this city is obtained by the station from **OpenWeatherMap API** server
- As a reply to the query of the bot, weather station should send the weather conditions like Temperature in degree Celsius, Humidity, Pressure and Status (Haze, Smog, Rain, Clear, etc.) of that city
- Display the same weather information on the LCD connected to ESP32
- If the station has no requests to respond to **(idle state)**, it is indicated with the help of **Blue** colour of RGB LED
- As soon as the station receives any request, RGB LED should turn **Green** till the request gets served
- Whenever someone tries to touch your server (esp32) you should get notified on Telegram bot that someone is trying to alter the server and a **separate LED** must be turned on as an indication alarm! **Bonus:** Now, you want to do some repair work in your station and you have to allow a technician to access your weather station. You should be able to turn off this feature remotely so that you won't get a notification when that technician touches the server until you turn it on again

1.2 Algorithm and Implementation

- Connect ESP32 to IITD_WIFI using institute login credentials.
- Initialize LCD on the ESP32.
- Create a telegram bot on Telegram app on android device for interfacing.
- Telegram bot communicate with ESP32.
- Take city name as input by the user on telegram bot.
- By API, take all information of weather of the city.
- Print all weather information on the serial monitor.
- Collect temperature, humidity, pressure and status from API and store them in variables.
- Print temperature, humidity, pressure and status on LCD, and send to Bot.
- Turn on Blue LED for Idle state.
- If any request comes, glow Green LED.
- if touch is sensed by the ESP, message is displayed on the telegram bot and Red LED must glow.

1.3 Program Structure



1.4 Screenshots

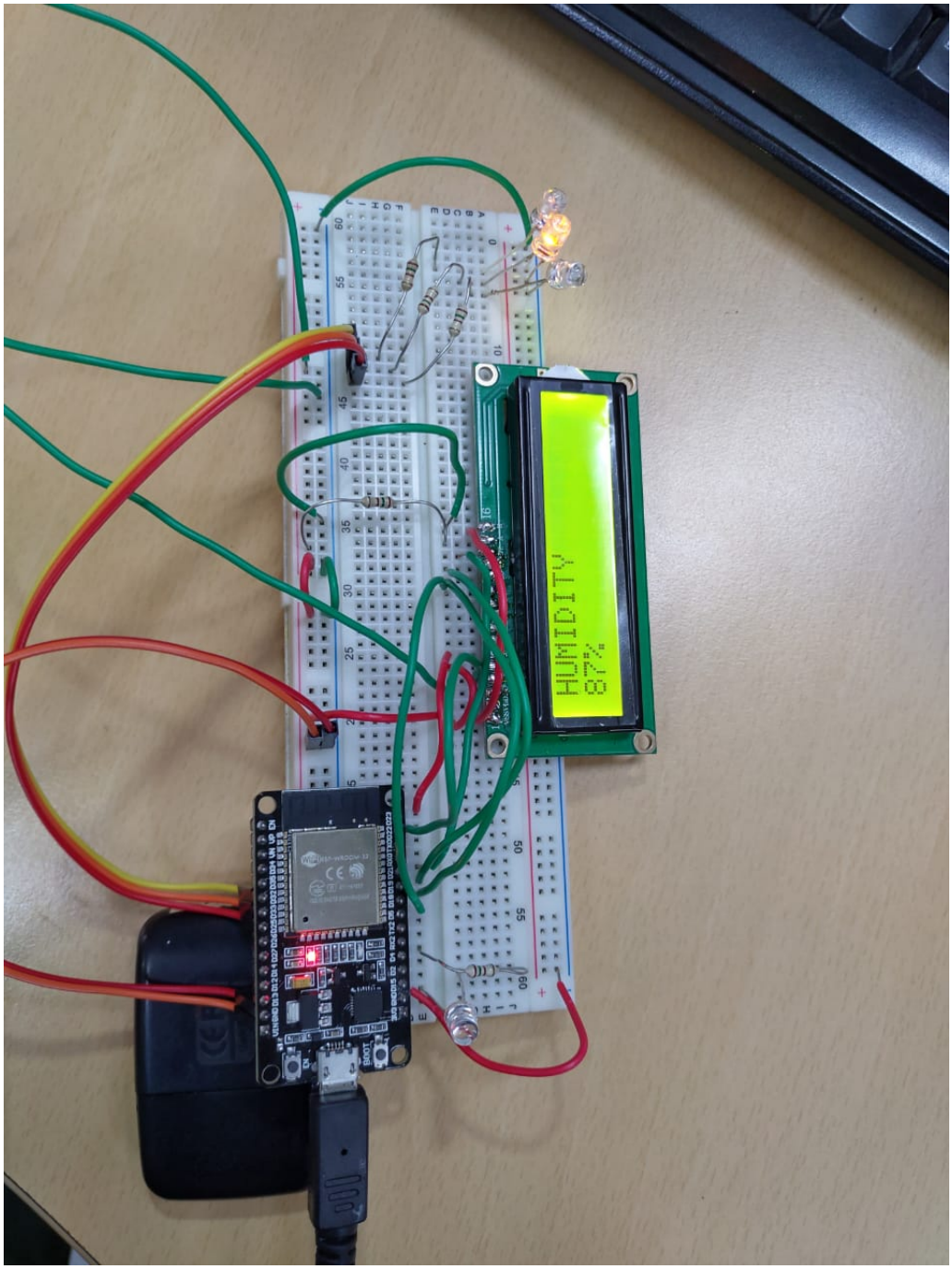


Figure 1: Hardware

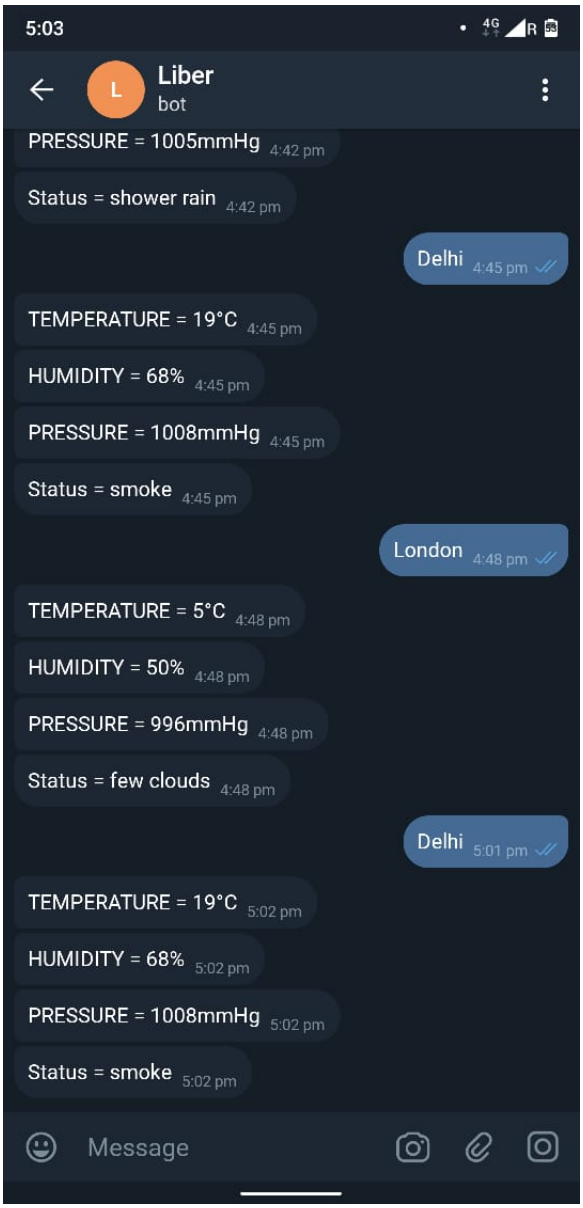


Figure 2: Telegram Bot

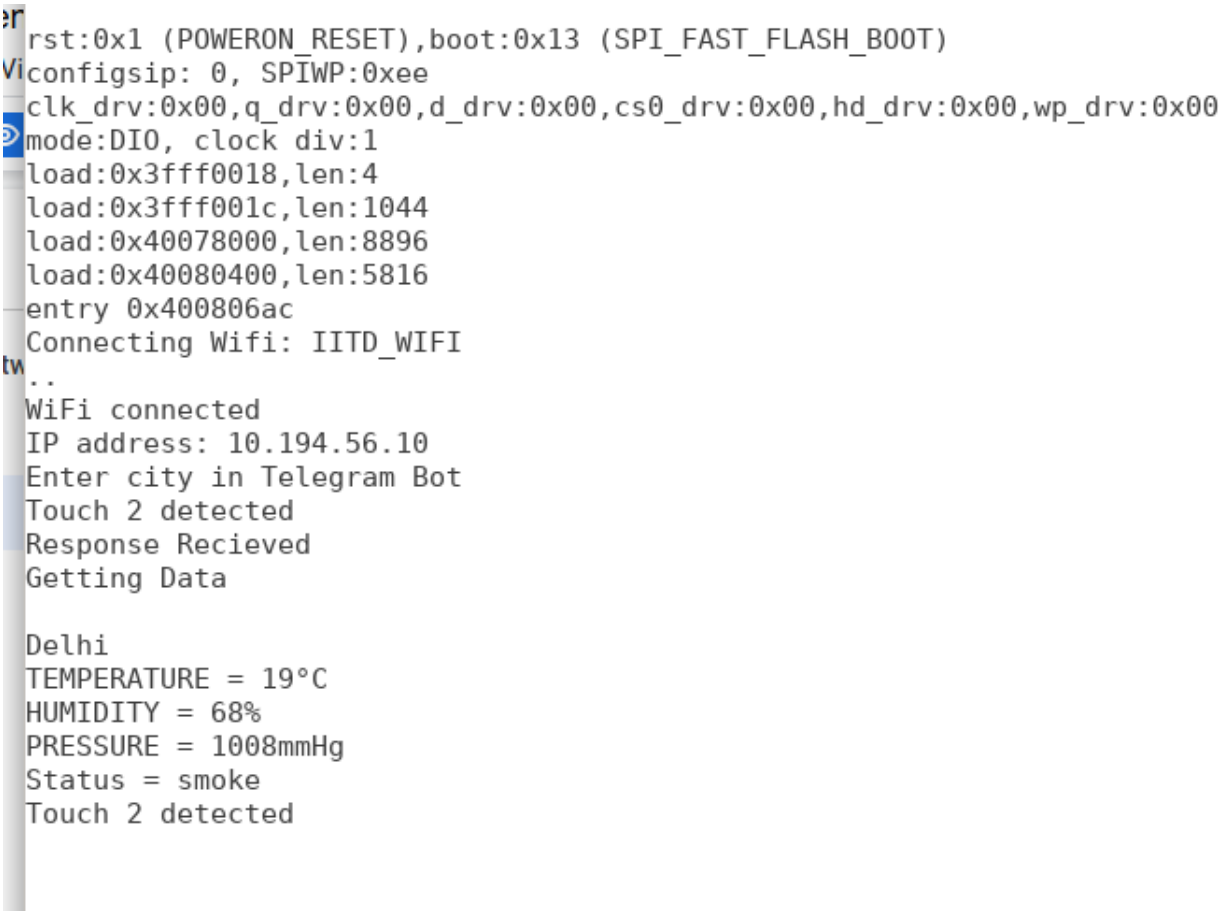


Figure 3: Serial Monitor

A Appendix

A.1 Code for ps1

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(22, 23, 5, 18, 19, 21);

#include <WiFi.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>
#include <HTTPClient.h>
#include "esp_wpa2.h" //wpa2 library for connections to Enterprise networks

#define EAP_IDENTITY "jtm192207" //if connecting from another corporation, use identity@organisation.dom
#define EAP_PASSWORD "jtm22072908" //your Eduroam password

const char* ssid = "IITD_WIFI"; // Eduroam SSID
const char* host = "https://proxy62.iitd.ac.in/cgi-bin/proxy.cgi"; //external server domain for HTTP co
int counter = 0;

StaticJsonBuffer<1024> jb;

// Initialize Telegram BOT
#define BOTtoken "1003941905:AAHSLltbvQjrSyjVqDTfAx8AY-Cc_q98sUs"  // your Bot Token (Get from Botfathe

WiFiClientSecure client;
UniversalTelegramBot bot(BOTtoken, client);

int Bot_mtbs = 1000; //mean time between scan messages
long Bot_lasttime;   //last time messages' scan has been done
String city;
String payload;
char payload1[500];
String temperature,humidity,pressure,state;
int temp;

////////////////////////////////////////////////////////////////////////////////////////////////////
int threshold = 40;
bool touch2detected = false;
void gotTouch2(){
 touch2detected = true;
}

////////////////////////////////////////////////////////////////////////////////////////////////////

void setup() {

  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  lcd.clear();
  lcd.print("----WELCOME----");
```

```
  lcd.setCursor(0,1);
  lcd.print("-WEATHER STATION-");
  delay(1000);
  pinMode(4, OUTPUT);
  pinMode(25, OUTPUT);
  pinMode(32, OUTPUT);
  pinMode(33, OUTPUT);


  Serial.begin(115200);
  delay(1000); // give me time to bring up serial monitor
  // Attempt to connect to Wifi network:
  Serial.print("Connecting Wifi: ");
  Serial.println(ssid);
  lcd.clear();
  lcd.print("Connecting Wifi");

  touchAttachInterrupt(T3, gotTouch2, threshold);

////////////////////////////////////////////////////////////////////////////////////////////////
  // Set WiFi to station mode and disconnect from an AP if it was Previously
  // connected
  WiFi.mode(WIFI_STA);
  esp_wifi_sta_wpa2_ent_set_identity((uint8_t *)EAP_IDENTITY, strlen(EAP_IDENTITY)); //provide identity
  esp_wifi_sta_wpa2_ent_set_username((uint8_t *)EAP_IDENTITY, strlen(EAP_IDENTITY)); //provide username
  esp_wifi_sta_wpa2_ent_set_password((uint8_t *)EAP_PASSWORD, strlen(EAP_PASSWORD)); //provide password
  esp_wpa2_config_t config = WPA2_CONFIG_INIT_DEFAULT(); //set config settings to default
  esp_wifi_sta_wpa2_ent_enable(&config); //set config settings to enable function
  WiFi.begin(ssid);

  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
    if(counter>=60){ //after 30 seconds timeout - reset board
      ESP.restart();
    }
  }

  Serial.println("");
  Serial.println("WiFi connected");
   lcd.clear();
   lcd.print("WiFi connected");
   delay(3000);
   lcd.clear();
   lcd.print("Enter city in ");
   lcd.setCursor(0,1);
   lcd.print("Telegram Bot");
   digitalWrite(4, LOW);
   digitalWrite(25, LOW);
   digitalWrite(32, LOW);
   digitalWrite(33, LOW);
```

```
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
  Serial.println("Enter city in Telegram Bot");

}

//////////////////////////////////////////////////////////////////////////////////////////

void handleNewMessages(int numNewMessages) {

  Serial.println("Getting Data");
  //Serial.println(String(numNewMessages));

  for (int i=0; i<numNewMessages; i++) {
    String chat_id = String(bot.messages[i].chat_id);
    String text = bot.messages[i].text;
    city = bot.messages[i].text;

    if (WiFi.status() == WL_CONNECTED)
  {
  HTTPClient http;
  http.begin("http://api.openweathermap.org/data/2.5/weather?q=" + city + "&APPID=da76b2b69eb1d79fa2cd4
  int httpCode=http.GET();
  if (httpCode > 0)
 {
        //Obtaining entire information
        payload = http.getString();
        //Serial.println("\n"+String(httpCode));
        //Serial.println(payload);
        Serial.println("");
        Serial.println(city);
        //Starting to print in LCD
        lcd.begin(16, 2);
        lcd.setCursor(0,0);
        lcd.print ("----IIT Delhi----");
        lcd.setCursor(0,1);
        lcd.print ("-WEATHER STATION-");
        delay(5000);
        lcd.clear();
//////////////////////////////////////////////////////////////////////////////////////////////////

        //Splitting temperature from info received
        temperature=payload.substring(payload.indexOf("temp")+6,payload.indexOf("feels_like")-2);
        temp=temperature.toInt();
        temp=temp-273;
        temperature=String(temp);
        //Send temperature to telegram
        bot.sendMessage(bot.messages[i].chat_id,"TEMPERATURE = "+ temperature + "C" , "");
        Serial.print("TEMPERATURE = ");
        Serial.print(temperature);
```

```
        Serial.println("C");

        //Printing temperature in LCD
        lcd.setCursor(0,0);
        lcd.print ("TEMPERATURE");
        lcd.setCursor(0,1);
        lcd.print (temperature);
        lcd.setCursor(2,1);
        lcd.print((char)223);
        lcd.setCursor(3,1);
        lcd.print("C");
        delay(5000);
        lcd.clear();
////////////////////////////////////////////////////////////////////////////////////////////////////

        //Splitting humidity from info received
        humidity=payload.substring(payload.indexOf("humidity")+10,payload.indexOf("visibility")-3);
        //Send humidity to telegram
        bot.sendMessage(bot.messages[i].chat_id, "HUMIDITY = "+ humidity + "%" , "");
        Serial.print("HUMIDITY = ");
        Serial.print(humidity);
        Serial.println("%");

        //Printing humidity in LCD
        lcd.setCursor(0,0);
        lcd.print ("HUMIDITY");
        lcd.setCursor(0,1);
        lcd.print (humidity);
        lcd.setCursor(2,1);
        lcd.print("%");
        delay(5000);
        lcd.clear();
////////////////////////////////////////////////////////////////////////////////////////////////////

        //Splitting pressure from info received
        pressure=payload.substring(payload.indexOf("pressure")+10,payload.indexOf("humidity")-2);
        //Send pressure to telegram
        bot.sendMessage(bot.messages[i].chat_id,"PRESSURE = "+ pressure + "mmHg" , "");
        Serial.print("PRESSURE = ");
        Serial.print(pressure);
        Serial.println("mmHg");

        //Printing pressure in LCD
        lcd.setCursor(0,0);
        lcd.print ("PRESSURE");
        lcd.setCursor(0,1);
        lcd.print (pressure);
        lcd.setCursor(4,1);
        lcd.print("mmHg");
        delay(5000);
        lcd.clear();
```

```
////////////////////////////////////////////////////////////////////////////////////////////////

         //Splitting status from info received
        state=payload.substring(payload.indexOf("description")+14,payload.indexOf("icon")-3);
        //Send status to telegram
        bot.sendMessage(bot.messages[i].chat_id,"Status = " + state, "");
        Serial.print("Status = ");
        Serial.print(state);
        Serial.println("");

        //Printing pressure in LCD
        lcd.setCursor(0,0);
        lcd.print ("STATUS");
        lcd.setCursor(0,1);
        lcd.print (state);
        delay(5000);
        lcd.clear();
        digitalWrite(33, LOW);
        lcd.print("Enter city in ");
        lcd.setCursor(0,1);
        lcd.print("Telegram Bot");

  }
    else {
      Serial.println("Error on HTTP request");
    }

  }
}
}


////////////////////////////////////////////////////////////////////////////////////////////

void loop() {
  if (millis() > Bot_lasttime + Bot_mtbs)  {

    digitalWrite(25, HIGH);
    // digitalWrite(33, LOW);
    int numNewMessages = bot.getUpdates(bot.last_message_received + 1);

    while(numNewMessages) {
      digitalWrite(25, LOW);
      digitalWrite(33, HIGH);
      Serial.println("Response Recieved");
      handleNewMessages(numNewMessages);
      numNewMessages = bot.getUpdates(bot.last_message_received + 1);
    }

    Bot_lasttime = millis();
  }
```

```
if(touch2detected){
    touch2detected = false;
    Serial.println("Touch 2 detected");
    digitalWrite(32, HIGH);
    delay(4000);
     digitalWrite(32, LOW);
  }
}

////////////////////////////////////////////////////////////////////////////////////////////
```

References

[1] *ESP32 Series datasheet*. https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf.

[2] *Getting Started with the ESP32 Development Board*. https://randomnerdtutorials.com/getting-started-with-esp32/.

[3] *Installing ESP32 Board in Arduino IDE on Ubuntu Linux*. https://circuits4you.com/2018/02/02/installing-esp32-board-in-arduino-ide-on-ubuntu-linux/.

[4] *Installing the ESP32 Board in Arduino IDE (Mac OS X and Linux instructions)*. https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-mac-and-linux-instructions/.

[5] *Quickly getting started with ESP32 / ESP32S in 5 - 10 minutes! Beginner Friendly! Arduino!* https://www.youtube.com/watch?v=mJcxnaR08Dg.

[6] *RGB LED*. https://randomnerdtutorials.com/electronics-basics-how-do-rgb-leds-work/.

[7] *Weather API*. https://openweathermap.org/api.