Polytechnic University of Puerto Rico
San Juan Campus

Electrical and Computer Engineering Faculty

Microprocessor Interfacing Laboratory

Final Project: Geographic location using Arduino

Kelvin Figueroa Figueroa - 108946
Rafael Gonzalez Cartagena - 90352

Professor: Dr. Roman Lopez

May 18, 2022

Contents

List of Tables

List of Figures

1 Design Summary

The purpose of this project is to obtain the location of our device, through the use of a GPS module. To take the project further, an application called Blynk is used to display the data obtained on a map by using internet connection and a NodeMCU module.

This design is quite simple, it only consists of an expanded NodeMCU called ESP8266 in connection with a GPS module (GT-U7) connected directly. Figure 1 shows the Schematic used to design the project. In table 1 reference is made to the connection between pins, but there is a detail to mention, the RXD and TXD pins must be inverted after uploading the code to the ESP8266.

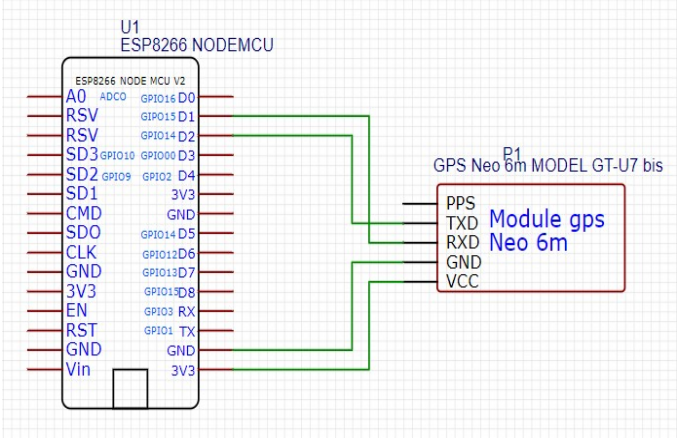

Figure 1: Schematic Design

Table 1: Design Pinout

ESP8266 (NodeMCU)	**GPS (GT-U7) Module**
3.3V	Vcc
GND	GND
D2	RXD
D1	TXD
None	PPS

The following figure represents the real connection between the ESP8266 and the GPS module, where it can be seen the details of table 1 with wires,

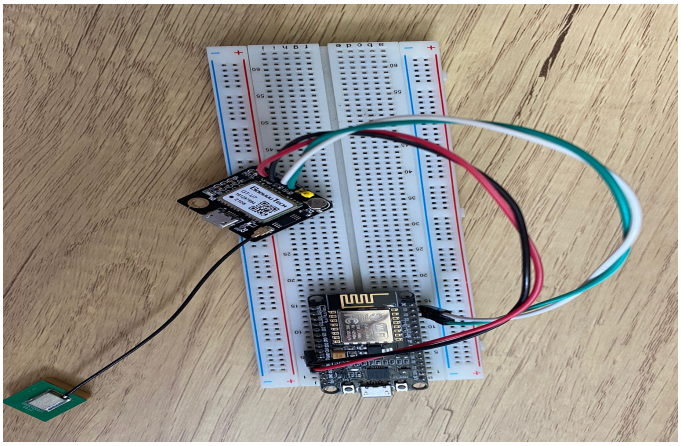

Figure 2: Real connection

2 System Details

In the previous section we saw a brief explanation of how our design works, but in this section we will be showing how our Arduino works and how it connects to the Iot Blynk application.

Blynk is an application designed for Internet of Things (IoT) projects, where the hardware can be controlled from a dashboard on your website portal or from the phone using an application. This platform allows us to connect devices, in this case to the internet, using a WiFi module and a phone, refer to figure 5. The three main components of the platform are:

1. Blynk App, which allows to create interfaces for projects using widgets.

2. Blynk Server, which is responsible for the communication between hardware and the smartphone.

3. Blynk Libraries, which enables the communication with the server and process all the incoming and out-coming commands. This library needed to be added to the Arduino IDE, see the following figures:

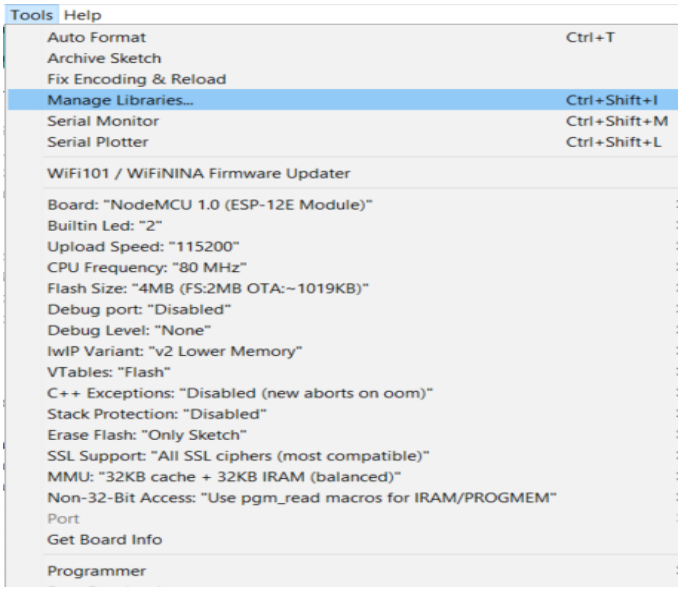

Figure 3: Downloading Blynk Library

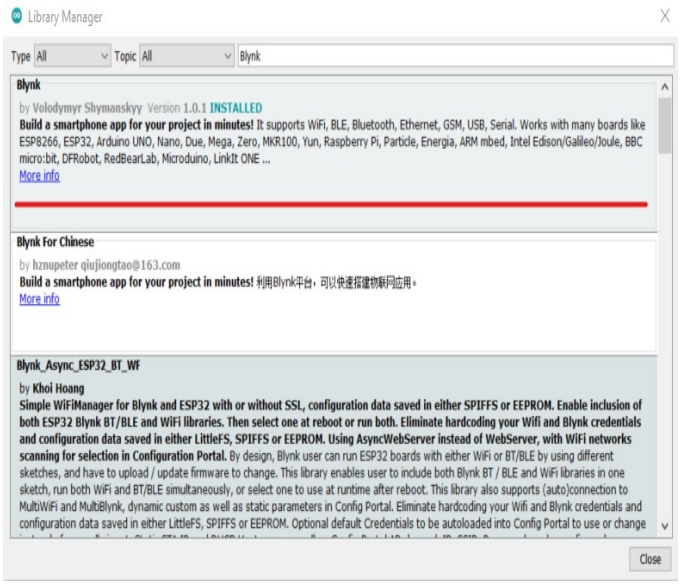

Figure 4: Downloading Blynk Library

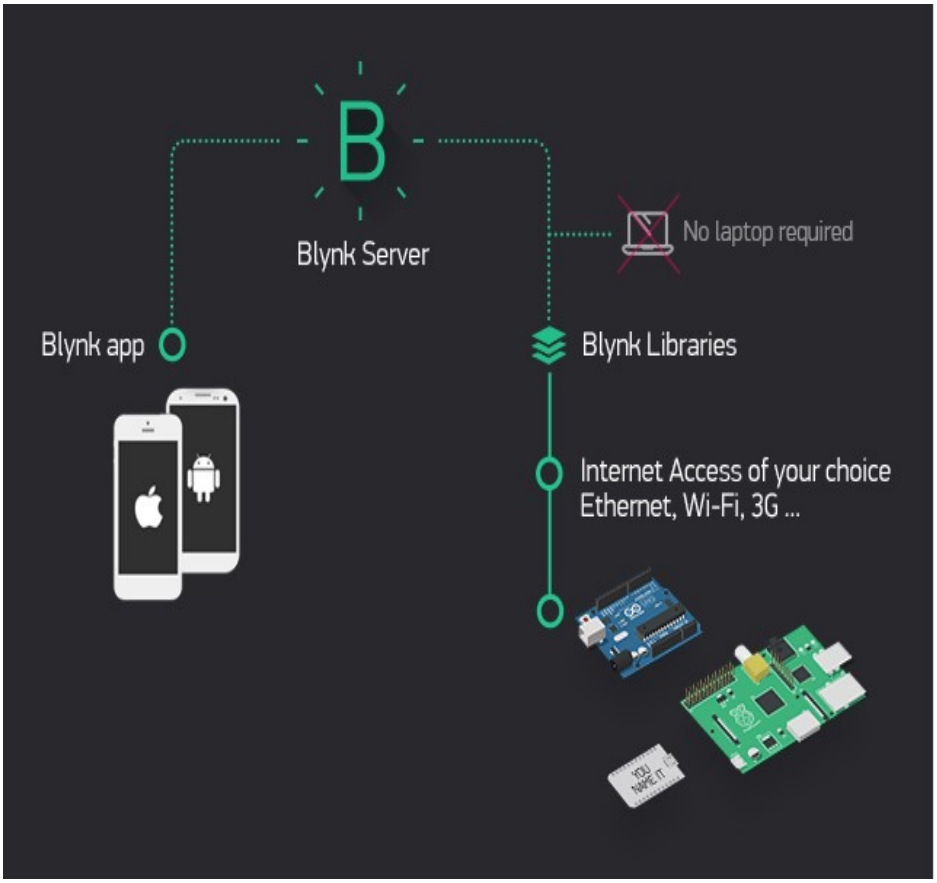

Figure 5: How Blynk Works

As mentioned in the first part, this project uses the ESP8266 module (refer to figure 6), which is able to connect to the internet giving way to the data collected from our GPS module can be used in the Blynk application. Using the WiFi module, a GPS module is connected (refer to figure 7), which works with a small antenna that locates satellites in orbit, and then detects location coordinates. To program this module we need a library called "TinyGPSPlus", which needed to be downloaded to Arduino's IDE.

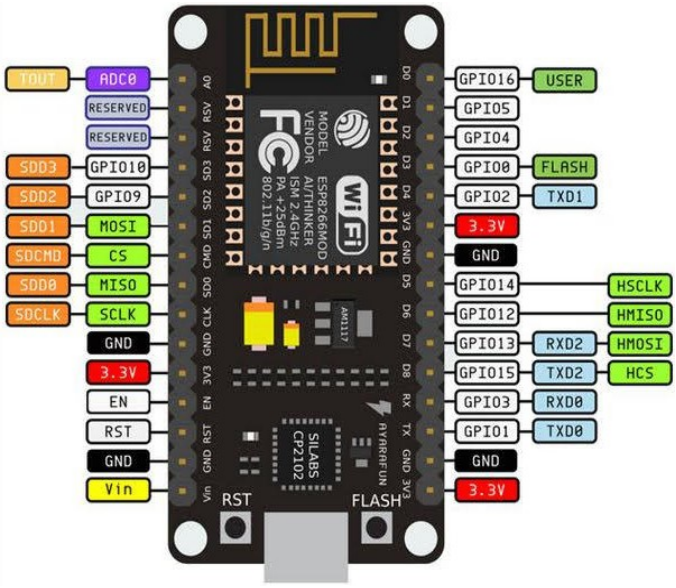

Figure 6: NodeMCU module

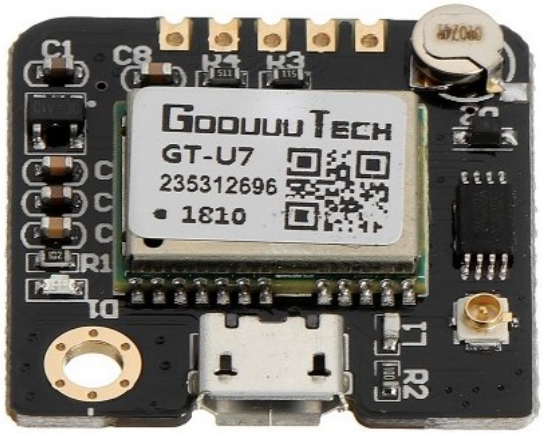

Figure 7: GT-U7 GPS module

3 Procedures

In this section we will see the process to implement the connection between the ESP8266 and the Blynk application.

1. LogIn into Blynk console and add new template, refer to the following link for explanation [1].

2. Open Blynk application on your phone and select the developer mode at the right upper corner.

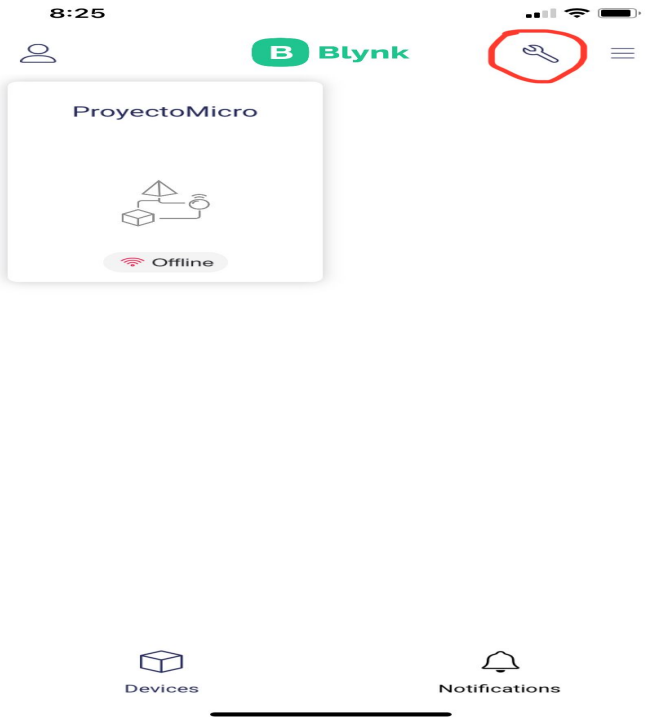

Figure 8: Blynk Developer Mode

3. Select the template to be use for the project.

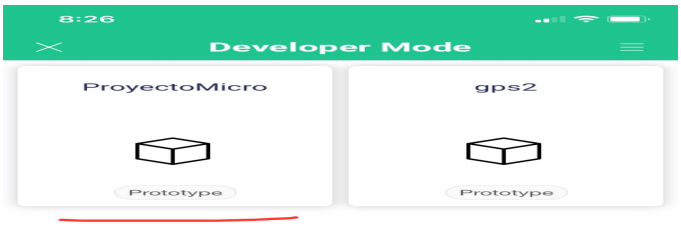

Figure 9: Template Selection

4. Add widgets to the template.

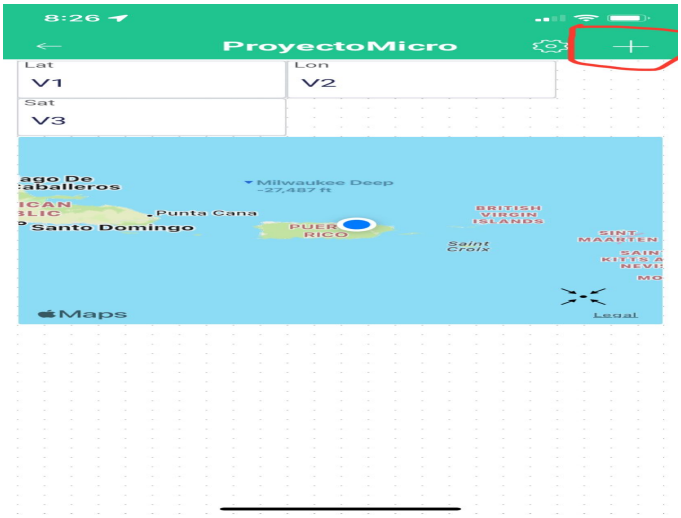

Figure 10: Adding Widgets

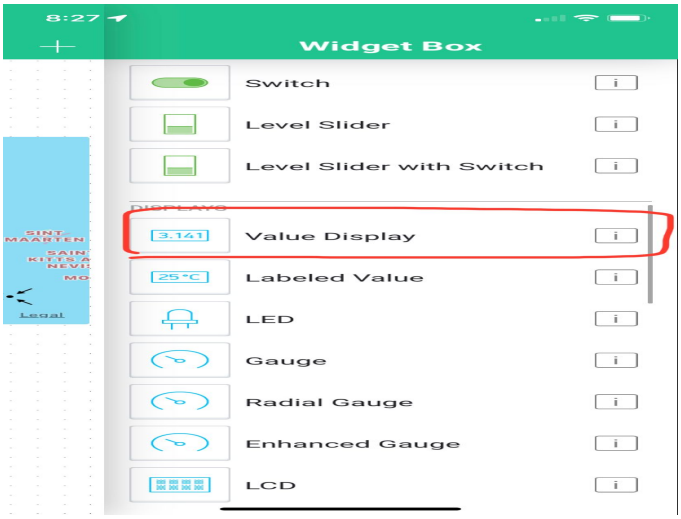

Figure 11: Adding Value Display for the coordinates

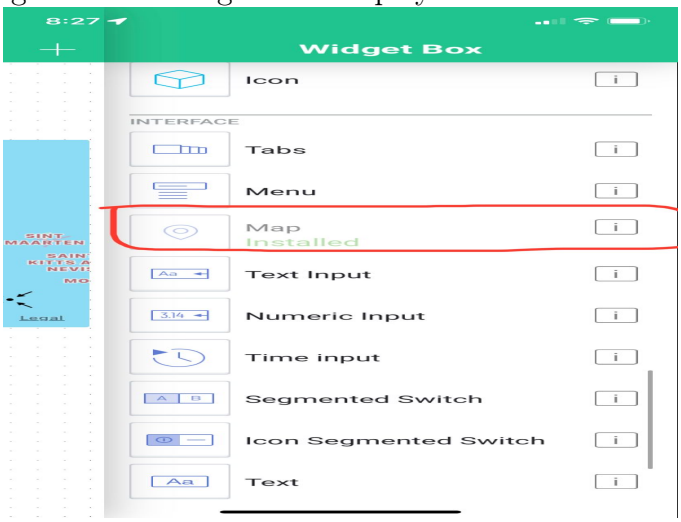

Figure 12: Adding MAP Widget

After creating the application, we proceed to create the code with the Arduino IDE, refer to section 7. Recall that when connecting the GPS module with the ESP the RXD and TXD pins need to be reversed after the code is uploaded to the card. Once the code is run our application will display the coordinates in the map widget as follows:

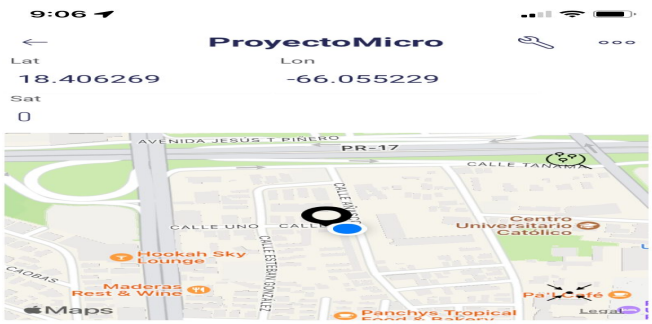

Figure 13: Application Running

4 Design Evaluation

The design shown in this document is for use with internet connection. It was made this way to be able to send the data to a map in which you can see the coordinates, maybe not exact but with more than 70 percent (%) of accuracy. For this project we opted for a design that was not too expensive, but could be effective in real life. Since most people have a smartphone, this design can be used on a daily basis to keep track of the location of some objects, such as wallets or bags, as long as they are within range of a WiFi network.

We can say that although our design works, it depends a lot on the internet connection, so an ideal model can be a connection through Bluetooth so that these devices stay connected to receive the location and to transfer it to the Blynk application.

5 Partial Part List

This section mentions some details of the devices used for the project.

5.1 ESP8266

The NodeMCU is a Node Micro Controller Unit that is an open source software and hardware development environment built around an inexpensive System-on-a-Chip called the ESP8266. It contains the crucial elements of a computer such as:

1. RAM
2. CPU
3. Wifi

Table 2: ESP8266 Module

ESP8266 (NodeMCU)	
Model	ESP8266MOD (EX)
Vendor	`https://amzn.to/3MrMRMe`
Price	$6.99

5.2 GPS

The GPS module GT-U7 is a stand alone GPS receiver with a high performance positioning engine. It has high sensitivity IPEX antenna for satellite positioning navigation. It only uses 3.3V with a baud rate of 9600, which can be modified.

Table 3: GPS Module

GPS (GT-U7 Neo-6M)	
Model	GT-U7 Neo-6M
Vendor	`https://amzn.to/3wAtyJT`
Price	$11.99

5.3 Blynk Application

Blynk is a full suite of software required to prototype, deploy, and remotely manage connected electronic devices at any scale: from personal IoT projects to millions of commercial connected products.

Table 4: Blynk Application

ESP8266 (NodeMCU)	
Vendor	`https://docs.blynk.io/en/`
Price	$6.99/month

6 Lessons Learned

For this project we saw how the GPS modules communicate through a serial communication of an ESP8266. The GPS module attracts all the data with a high sensitivity IPEX antenna for satellite positioning navigation, so our device needs to have a clear view of the sky. It is important to mention that to obtain the location data with the GPS modules, the RXD and TXD connections must be inverted, which some modules already have by default, but in the case of the NodeMCU this is not the case. It was also shown how different web applications can be used for IoT designs, such as Blynk, so that the data can be visualized in different ways such as on a map.

7 Arduino Code

This section illustrates the program that was developed for this project:

```
/*
 The following information can be found in the Blynk web dashboard
    template,
 The information is necessary for our device to connect to the cloud
    application through the internet.
   1. #define BLYNK_TEMPLATE_ID
   2. #define BLYNK_DEVICE_NAME
   3. #define BLYNK_AUTH_TOKEN
*/
#define BLYNK_TEMPLATE_ID "TMPLAwe9YV8O"
```

```
#define BLYNK_DEVICE_NAME "ProyectoMicro"
#define BLYNK_AUTH_TOKEN "fAfLPtEOgSQd-IpkQm6HFw5SolsQZGnd"
#define BLYNK_FIRMWARE_VERSION    "0.1.0"
#define BLYNK_PRINT Serial
#define APP_DEBUG

#include <TinyGPS++.h>
#include <SoftwareSerial.h>
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

BlynkTimer timer;
TinyGPSPlus my_gps;
WidgetMap my_map(V0);
SoftwareSerial ss(4,5);

char auth[] = BLYNK_AUTH_TOKEN;

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "Kelvin";
char pass[] = "Figueroa1998";

void setup()
{
  /* Baud of the device (ESP8266) is 115200 and Serial comunication
      is 9600
     Function Blynk.begin() connecs the NodeMCU to the Blynk Cloud,
        using credentials such as:
      1. Authentication token from device
      2. SSID, which is the name of the network to be connected
      3. Password of the network if any

     Timer function (timer.setInterval(secondsL, function to run)),
        runs a funtion X seconds
  */
  Serial.begin(115200);
  ss.begin(9600);
  delay(100);
  Blynk.begin(auth, ssid, pass);
```

```
46
47    //Set function setup to be called every 10seconds
48    timer.setInterval(10000L, CheckGPS);
49  }
50
51  void CheckGPS(){
52    /*
53     Function that verifies if the GPS is working by receiving the
          total of characters received by the object
54     IF the total of characters are less than 10, then the function
          sends an error message to the Serial Monitor.
55    */
56    if (my_gps.charsProcessed() < 10){
57      Serial.print("No GPS detected, invert Rx and Tx on the
            NodeMCU.");
58      Blynk.virtualWrite(V3, "GPS ERROR");
59    }
60  }
61
62  void loop() {
63    /*
64     Function that runs the program while the Serial Software is
          available,
65     by calling the display function.
66
67     Blynk.run():
          https://docs.blynk.cc/#blynk-firmware-connection-management-blynkrun
68     timer.run(): https://docs.blynk.cc/#blynk-firmware-blynktimer
69    */
70    while (ss.available() > 0)
71    {
72      // sketch displays information every time a new sentence is
            correctly encoded.
73      if (my_gps.encode(ss.read())){
74        displayInfo();
75      }
76    }
77    Blynk.run();
78    timer.run();
79  }
```

```
80
81  void getData(float &latt, float &longg, int &sat){
82    /*
83     Function to get the data from the GPS module
84    */
85    latt = (my_gps.location.lat()); //Storing the Latitude value from
          gps
86    longg = (my_gps.location.lng()); //Storing Longitude value from gps
87    sat = my_gps.satellites.value();   //get number of satellites
88  }
89  void displayInfo()
90  {
91    /*
92     IF the GPS module is available, then:
93       Function to display the information obtained by the gps module,
             variables are:
94        1. lt -> Latitude
95        2. ln -> Longitude
96       These variables are set to be float with 6 decimal points
             (##.######)
97
98     References:
99      1. MapWidget:
           https://docs.blynk.io/en/blynk.console/widgets-console/map
100     2. Virtual Pins:
           https://docs.blynk.cc/#blynk-main-operations-virtual-pins
101     3. Getting Started:
           https://docs.blynk.cc/#getting-started-getting-started-with-the-blynk-app
102   */
103   float lt, ln;
104   int sat;
105   if (my_gps.location.isValid() and Blynk.connected())
106   {
107     getData(lt,ln,sat);
108     Serial.print("SATS: ");
109     Serial.println(sat); // float to x decimal places
110     Serial.print("LATITUDE: ");
111     Serial.println(lt, 6); // float to x decimal places
112     Serial.print("LONGITUDE: ");
113     Serial.println(ln, 6);
```

```
114
115     Blynk.virtualWrite(V1, String(lt,6));
116     Blynk.virtualWrite(V2, String(ln,6));
117     Blynk.virtualWrite(V3, sat);
118     Blynk.virtualWrite(V0, ln, lt);
119   }
120   else if(Blynk.connected()==false){
121     Serial.print("LAST LOCATION IS: " + String(lt,6)+ ';' +
          String(ln,6) + '\n');
122   }
123   //Serial.println();
124 }
```

It should be noted that if further information about the GPS portion of the code is needed, i.e., that of **TinyGPS**++, all of it can be found in [2].

8 Conclusion

In conclusion, it was possible to demonstrate that the design used for this project is effective when using an ESP8266 and a GT-U7 Neo-6M GPS. It is worth mentioning that for a more efficient design, a connection through Bluetooth can be used, in this way the dependency would not be only through the internet.

References

The following *References* related with this document are in the IEEE Format, as shown below:
[1] MIT, Blynk Documentation. [Online]. Available: `https://docs.blynk.cc/#getting-started-getting-started-with-the-blynk-app.` [Accessed: 18-May-2022].
[2] Mikalhart, "Mikalhart/tinygpsplus: A new, customizable Arduino NMEA Parsing Library," GitHub, 2022. [Online]. Available: https://github.com/mikalhart/TinyGPSPlus. [Accessed: 11-May-2022].